P-3606

SYBIL GUARD - AN OPTIMAL NETWORK DEFENSE AGAINST SYBIL ATTACKS

PROJECT REPORT

Submitted by

SENTHIL NATHAN.K **Reg.No: 0710108048**

NITHIN RAJ.P **Reg.No: 0710108032**

In partial fulfillment for the award of the degree

Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE – 641 049

APRIL 2011

KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University of Technology, Coimbatore)

COIMBATORE - 641049

Department of Computer Science and Engineering

PROJECT WORK, April 2011

BONAFIDE CERTIFICATE

This is to certify that the project entitled "**SYBIL GUARD - AN OPTIMAL NETWORK DEFENSE AGAINST SYBIL ATTACKS**", the bonafide work of **SENTHIL NATHAN.K, NITHIN RAJ.P** who carried out the project work under our supervision.

[Mrs.S.Rajini, M.S.,]
Project Guide

[Mrs.P.Devaki, M.E.,(Ph.D),]
Head of the Department/CSE

The candidates with **university Register Nos. 0710108048 & 0710108032** was examined by us in project viva-voce examination held on 20/4.

Internal Examiner

External Examiner

DECLARATION

We,

SENTHIL NATHAN.K **Reg.No: 0710108048**
NITHIN RAJ.P **Reg.No: 0710108032**

Hereby declare that the project entitled **"SYBIL GUARD - AN OPTIMAL NETWORK DEFENSE AGAINST SYBIL ATTACKS"**, submitted in partial fulfillment to Anna University of Technology as the project work of Bachelor of Engineering (Computer Science and Engineering) degree, is record of original work done by us under the supervision and guidance of Department of Computer Science and Engineering, Kumaraguru College of Technology, Coimbatore.

Place: Coimbatore
Date:

[SENTHIL NATHAN.K] [NITHIN RAJ.P]

Project Guided by,

[Mrs.S.Rajini,M.S.,]
Associate Professor
Department of Computer Science and Engineering,
Kumaraguru College of Technology,
(An Autonomous Institution)
Coimbatore-641 049.

ACKNOWLEDGEMENT

First and foremost, We would like to thank the Lord Almighty for enabling us to complete this project.

We express our profound gratitude to our Chairman **Padmabhusan Arutselvar Dr.N.Mahalingam, B.Sc., F.I.E.,** for giving this opportunity to pursue this course.

We would like to thank **Dr.J.Shanmugam, Ph.D.,** *Director* and **Dr.S.Ramachandran, Ph.D.,** *Principal* for providing the necessary facilities to complete our project.

We take this opportunity to thank **Dr.S.Thangasamy Ph.D.,** *Dean, Research and Development,* for his precious suggestions. We also thank **Mrs.P.Devaki M.E. (PhD),** *HOD,* Department of Computer Science and Engineering, for her support and timely motivation.

We register our hearty appreciation to the Guide **Mrs.S.Rajini, M.S.,** *Associate Professor,* Department of Computer Science and Engineering, our Project advisor. We thank her for her support, encouragement and ideas.

We would like to convey our honest thanks to all **Teaching** staff members and **Non Teaching** staffs of the department for their support. We would like to thank all our classmates who gave us proper light moments and study breaks apart from extending some technical support whenever we needed them most.

ABSTRACT

Open-access distributed systems such as peer-to-peer systems are particularly vulnerable to *Sybil attacks*, where a malicious user creates multiple fake identities (called *Sybil nodes*). Without a trusted central authority that can tie identities to real human beings, defending against Sybil attacks is quite challenging. Among the small number of decentralized approaches, our recent Sybil Guard protocol leverages a key insight on social networks to bound the number of sybil nodes accepted. Despite its promising direction, SybilGuard can allow a large number of Sybil nodes to be accepted.

Furthermore, SybilGuard assumes that social networks are fast-mixing, which has never been confirmed in the real world. This project presents a novel SybilLimit protocol that leverages the same insight as SybilGuard, but offers dramatically improved and near-optimal guarantees. The number of Sybil nodes accepted is reduced by a factor of around 200 times in our experiments for a million-node system.

We further prove that Sybil Limit's guarantee is at most a factor away from optimal when considering approaches based on fast-mixing social networks. Finally, based on three large-scale real-world social networks, we provide the first evidence that real-world social networks are indeed fast-mixing. This validates the fundamental assumption behind SybilLimit's and SybilGuard's approach.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

1. INTRODUCTION

As the scale of a decentralized distributed system increases, the presence of malicious behavior (e.g., Byzantine failures) becomes the norm rather than the exception. Most designs against such malicious behavior rely on the assumption that a certain fraction of the nodes in the system are honest. For example, virtually all protocols for tolerating Byzantine failures assume that at least 2/3 of the nodes are honest. This makes these protocols vulnerable to *Sybil attacks,* Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

In which a malicious user takes on multiple identities and pretends to be multiple, distinct nodes (called *Sybil nodes* or *Sybil identities*) in the system. With Sybil nodes comprising a large fraction (e.g., more than 1/3) of the nodes in the system, the malicious user is able to "out vote" the honest users, effectively breaking previous defenses against malicious behaviors. Thus, an effective defense against Sybil attacks would remove a primary practical obstacle to collaborative tasks on peer-to-peer (p2p) and other decentralized systems. Such tasks include not only Byzantine failure defenses, but also voting schemes in file sharing, DHT routing, and identifying worm signatures or spam.

A trusted central authority that issues and verifies credentials unique to an actual human being can control sybil attacks easily. For example, if the system requires users to register with government-issued social security numbers or

driver's license numbers, then the barrier for launching a sybil attack becomes much higher. The central authority may also instead require a payment for each identity. Unfortunately, there are many scenarios where such designs are not desirable. For example, it may be difficult to select/establish a single entity that every user worldwide is willing to trust. Furthermore, the central authority can easily be a single point of failure, a single target for denial-of-service attacks, and also a bottleneck for performance, unless its functionality is it widely distributed. Finally, requiring sensitive information or payment in order to use a system may scare away many potential users.

Defending against Sybil attacks without a trusted central authority is very hard. Many decentralized systems today try to combat sybil attacks by binding an identity to an IP address. However, malicious users can readily harvest (steal) IP addresses. Note that these IP addresses may have little similarity to each other, thereby thwarting attempts to filter based on simple characterizations such as common IP prefix. Spammers, for example, are known to harvest a wide variety of IP addresses to hide the source of their messages, by advertising BGP routes for unused blocks of IP addresses. Beyond just IP harvesting, a malicious user can co-opt a large number of end-user machines, creating a botnet of thousands of compromised machines spread throughout the Internet. Botnets are particularly hard to defend against because nodes in botnets are indeed distributed end users' computers.

1.1 PROJECT DESCRIPTION

This section formalizes the desirable properties and functions of a defense system against sybil attacks. We begin by defining our system model. The system has n honest human beings as honest users, and one or more malicious human beings as malicious users. By definition, a user is distinct. Each honest user has a single (honest) identity, while each malicious user has one or more (malicious) identities. To unify terminology, we simply refer to all the identities created by the malicious users as sybil identities. Identities are alsocalled nodes, and we will from now on use "identity" and "node" interchangeably. All malicious users may collude, and we say that they are all under the control of an adversary.

Nodes participate in the system to receive and provide service (e.g., file backup service) as peers. Because the nodes in the system may be honest or sybil, a defense system against sybil attacks aims to provide a mechanism for a node V to decide whether or not to accept or reject another node S. Accepting S means that V is willing to receive service from and provide service to S. Ideally, the defense system should guarantee that V accepts only honest nodes. Because such an idealized guarantee is challenging to achieve, we aim at providing the following guarantees that, while weaker, are still sufficiently strong to be useful.

CHAPTER – 2

SYSTEM DEVELOPMENT

2.1 SYSTEM ANALYSIS

System analyses can be defined as a reduction of an entire system by studying various operation performed and their relationship within system and examination of business activity with a view to identify the problem areas.

2.1.1 EXISTING SYSTEM

Sybil attack means use others IP address for our communication. It create major problem in the network. In Existing system Packets are filter by using Distributed Packet filtering method used here. Denial of service (DOS) attack is major issue in the network, this attack create confuse while detecting attacker. Inter domain ip spoofing can't detecting here. Because source router do the detecting work based on the source router connection, if attacker use address of some other system which is using the same source router for reaching destination which becomes a problem for finding the attacker.

2.1.2 PROPOSED SYSTEM

In the Proposed system, route-based packet filtering method is used. Each node has individual path for destination. It is detected using the Sybil Guard System. It determines the path of the node . This path appends with the packet header. Packet filter takes decision to either pass the packet or discard the packets. So inter domain Sybil attack is avoided by using this method.

2.2 DEVELOPMENT ENVIRONMENT

2.2.1 HARDWARE CONFIGURATION

Processor Name	: Pentium IV
Processor Speed	: 1.7 GHz
Memory (RAM)	: 256 MB
Hard Disk	: 10 GB

2.2.2 SOFTWARE CONFIGURATION

Operating System	-	Windows XP and above
Software Tools	-	Microsoft Visual studio 2010 (C#.Net)
Application Server	-	Internet Information Server

2.2.3 SOFTWARE DESCRIPTION

Microsoft Visual studio 2008 (C#.Net):

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It can be used to develop console and graphical user interface applications along with Windows Forms applications, web sites, web applications, and web services in both native codes together with managed code for all platforms supported by Microsoft Windows.

C# is a multi-paradigm programming language encompassing imperative, declarative, functional, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft within the .NET initiative C# is one of the programming languages designed for the Common Language Infrastructure. C# is intended to be a simple, modern, general-purpose, object-oriented programming language.

CHAPTER – 3

PROJECT PLAN AND FEASIBILITY STUDY

3.1 TEAM STRATERGY AND WORK

The team consists of 2 members all are qualified for the system to be developed. The system was first designed, coded and implemented in the local host using in-built visual studio's server and testing process was done.

3.2 DEVELOPMENT SCHEDULE

3.2.1 Milestones

Milestones are being established for each and every module to improve the product visibility. It enhances the development process to become more tangible. It exposes errors, which help in improving the product quality and increase project communication. In our application it has been done sub-module wise.

3.2.2 Reviews

Review issues lists, are prepared to identify problem area within the product. As a programmer we do the following reviews.

- Critical Design Review
- Source code Review
- Acceptance Test Review

3.3 FEASIBILITY ANALYSIS

The feasibility study is very rough analysis of the viability of a project. It is however a highly desirable checkpoint that should be completed before committing to more resources. Feasibility study is conducted to obtain an overview of the problem and to roughly whether feasible solutions exist prior to committing substantial resources to a project.

The primary objective of a feasibility study is to assess three types of feasibility.

- Operational feasibility
- Technical feasibility
- Economical feasibility

3.3.1 Operational feasibility

Operational feasibility study is must, because it ensures that that the project implements in the organization work the feasibility should be high.

The Operational feasibility is high in this project as it allows to register for mark updating, result viewing and provides good interface, which is easy and friendly for the user to use it.

3.3.2 Technical feasibility

Technical feasibility analysis makes a comparison of the level of technology available and the same is required for the development of the product. The level of technology accounts for factors such as the programming language, the machine environment, the programming practices and the software tools.

Resource availability such as Pentium 3 processor with 128MB RAM, software and tools required for the project are available at the organization. Hence it is technical feasible.

3.3.3 Economical feasibility

This is the most important aspects that have to be critically evaluated. This includes the feasibility study of cost - benefit analysis. This is an assessment of the economic justification for a computer based system project. Most of the software is available in the web. Hence the threat of financial non-feasibility does not exist. It is determined that benefits out beat the cost of implementation and the system is considered to be economically feasible.

CHAPTER – 4

MODULE DESCRIPTION

4.1 Introduction

Our system consists of 3 modules which are listed as follows:

- Server side
- Client side
- Sybil guard

4.1.1 SERVER SIDE MODULE

Server side consists of the multimedia data which is accessed by the client using its unique ID provided by the server. The unique ID is created manually by the admin and stored in the server. If hacker hacks the system by IP spoofing the unique ID would terminate the network connection.

4.1.2 CLIENT SIDE MODULE

The client side module consist of the users which has been registered in the network using the unique ID, the client could access the server only till the unique ID is registered in the Sybil guard.

4.1.3 SYBIL GUARD

The Sybil guard consists of the unique ID numbers which has been registered. The connection of the network would be established only if the code in the Sybil guard and the unique ID in the client are same.

CHAPTER 5

SYSTEM DESIGN AND IMPLEMENTATION

5.1 SYSTEM DESIGN

System Design is a solution, a "how to" approach to the creation of new system It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. A Design goes through the logical and physical stages of development. Design is a creative process that involves working with the unknown new system, rather than analyzing the existing system. Thus, in analysis it is possible to produce the correct model of existing system.

5.1.1 INPUT DESIGN

Input screens form the primary interfaces between the user and software. It clearly describes the users what type of data should be given. The input screens are designed in such a way that it has a simple and user-friendly layout.

The screens are designed to capture all the necessary and sufficient data to the system. It includes the validations that are to be done during data entry. The data is the basis for information systems. Without data, there is no system, but data must be provided in the format acceptable to the user. A screen is actually a display station that has a buffer for storing data. The main objective of the screen design is for simplicity, accurate and quick data capture or entry.

5.1.2 OUTPUT DESIGN

Computer output is the most important and direct source of information to the user. The efficient and intelligible output design improves the system's relationship with the user and help in decision making.

The output should be designed around the requirements of the user. Outputs from the computers are required primarily to communicate the result of processing to the users and to provide a permanent copy of the results for later communication. The entire system appears fine if the produced output is well qualified, otherwise it causes the entire system to fail.

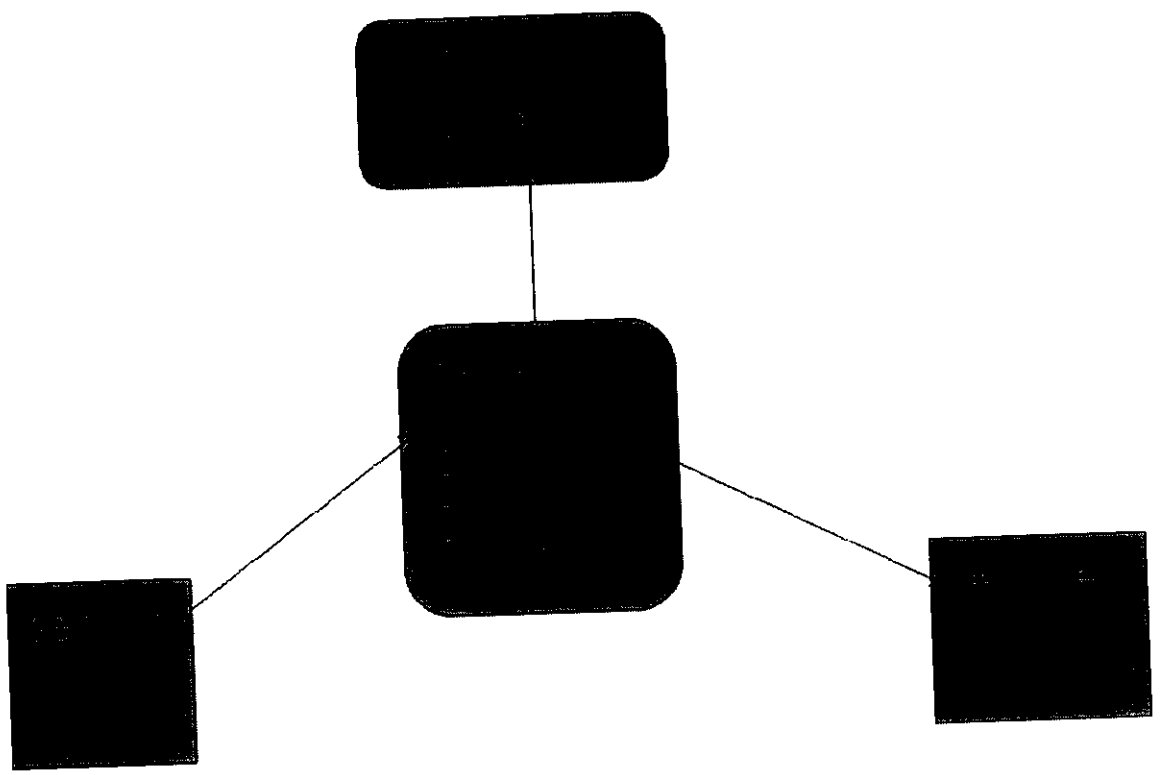
Designing computer output should proceed in an organized, will thought out manner. The right input element ensures that the developed output is qualified one. It makes the users to training themselves easily and effectively. The term output applies to any information produced by an information system, whether printed or displayed.

5.1.3 DATA BASE DESIGN

A database is a collection of interrelated data with minimum redundancy to serve the user quickly and efficiently. The data are stored in tables. It contains the individual records, the sequence in which the records are held on the storage medium and the order in which they may be accessed.

Without data there is no system, but the data must be provided in the right form for input and the information produced must be in a format acceptable to the user.

D 1.1 ARCHITECTURE DIAGRAM



D 1.2 SEQUENCE DIAGRAM

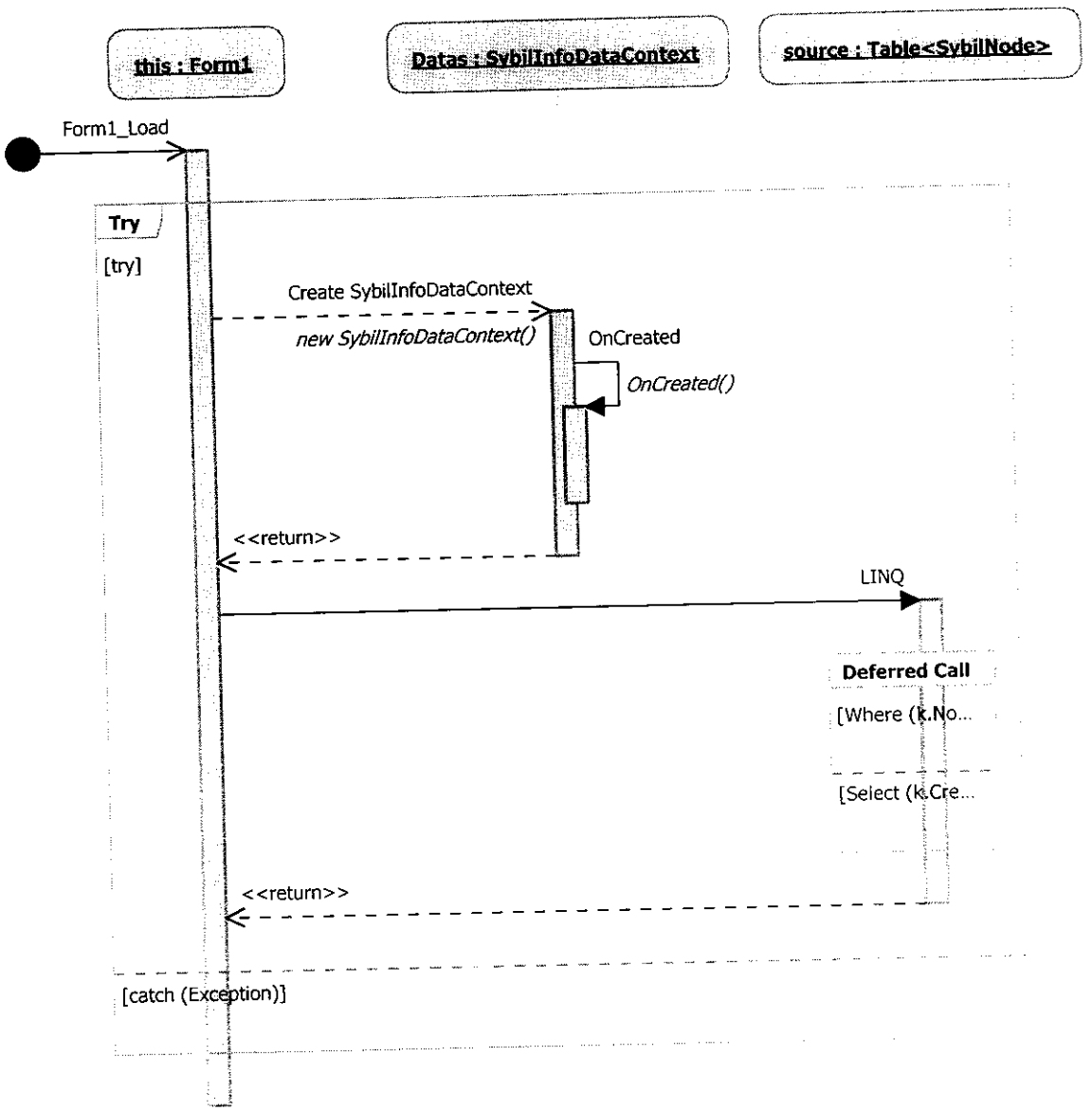
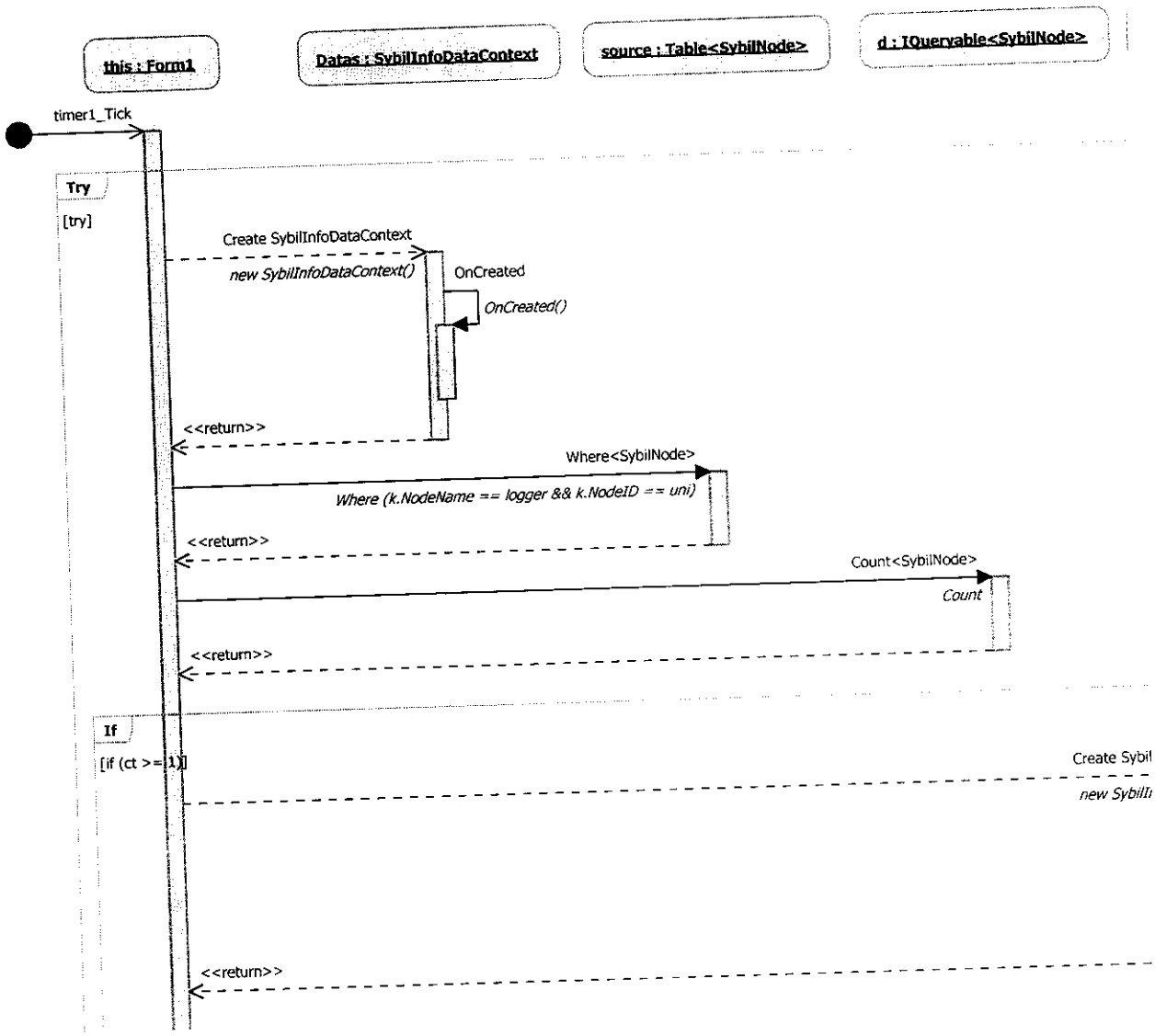
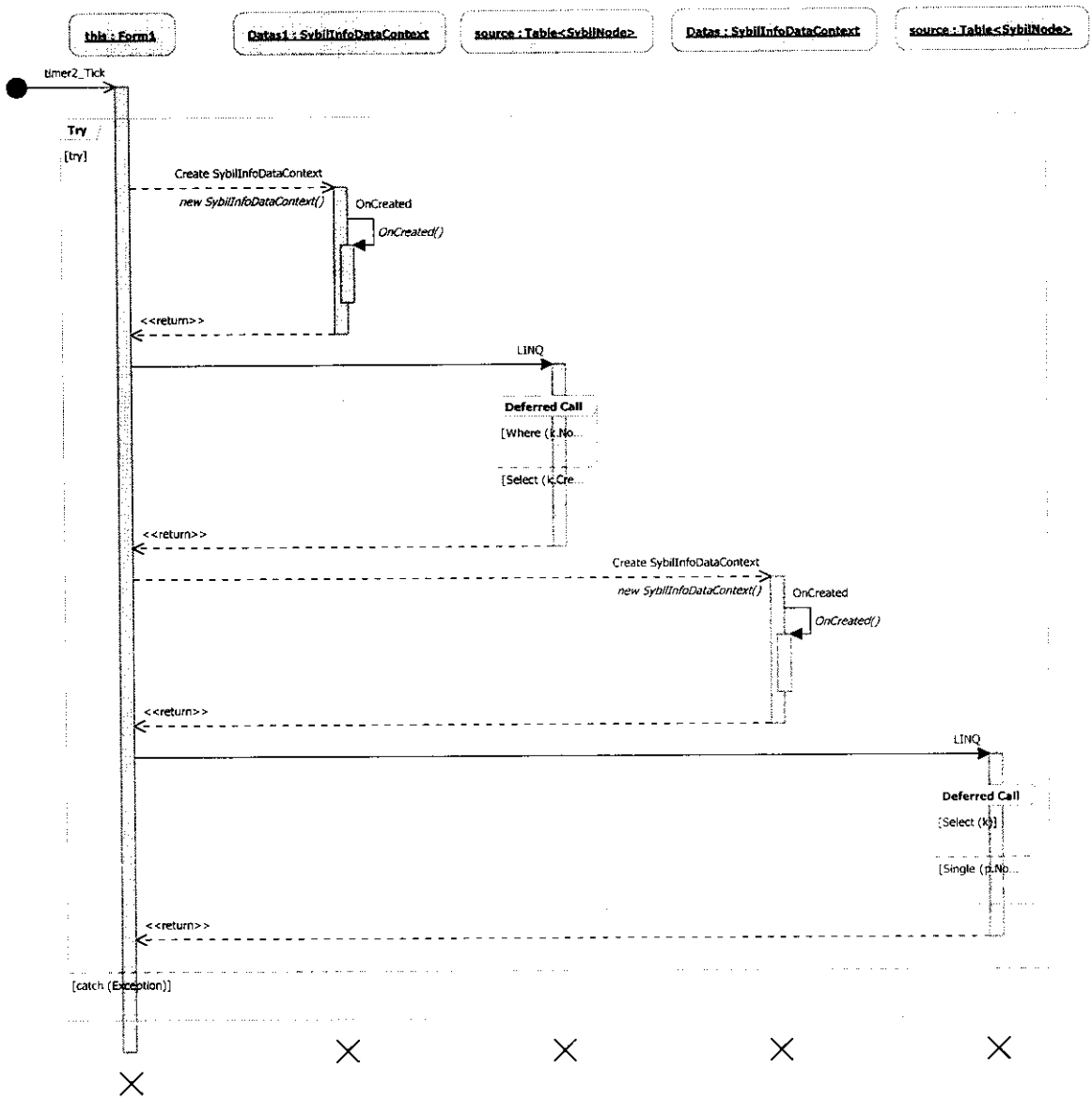
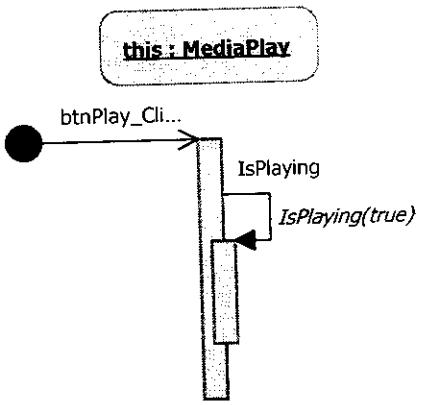
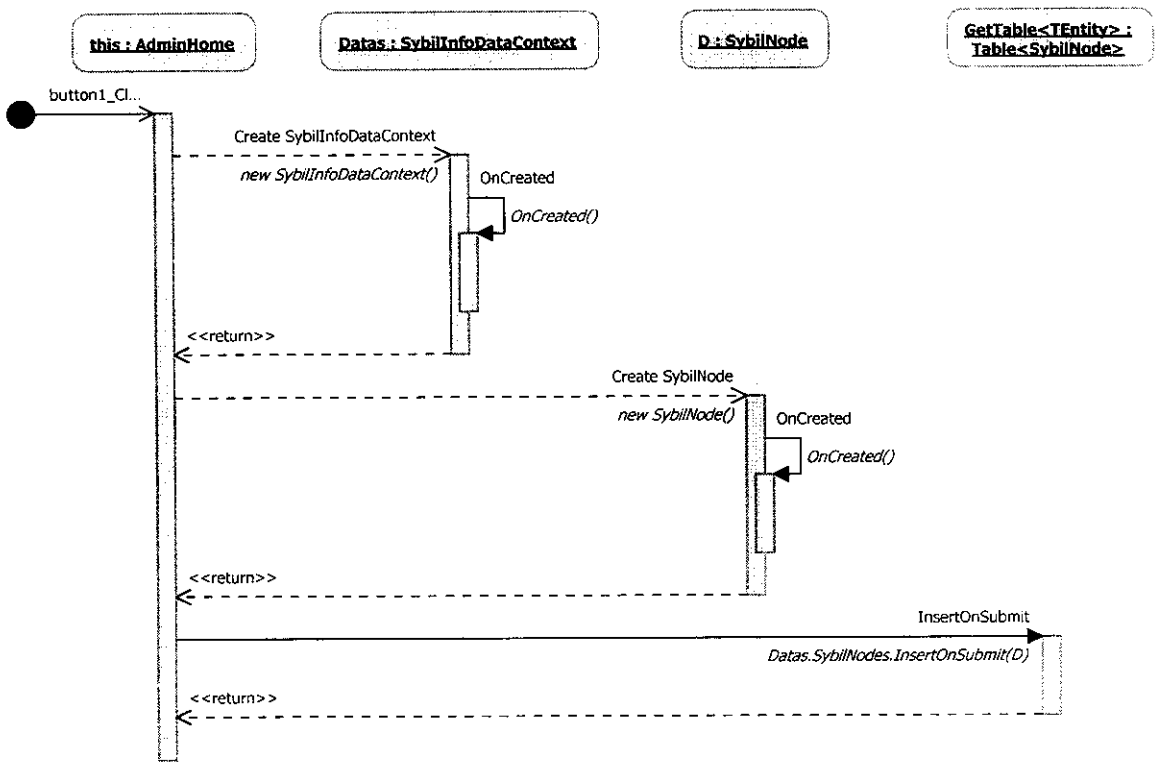


FIG no:1


this : Form1
Datas : SybilInfoDataContext
source : Table<SybilNode>
d : IQueryable<SybilNode>
timer1_Tick
Try
[try]
Create SybilInfoDataContext
new SybilInfoDataContext()
OnCreated
OnCreated()
<<return>>
Where<SybilNode>
Where (k.NodeName == logger && k.NodeID == uni)
<<return>>
Count<SybilNode>
Count
<<return>>
If
[if (ct >= 1)]
Create Sybil
new SybilI
<<return>>


this : Form1
Datas1 : SybilInfoDataContext
source : Table<SybilNode>
Datas : SybilInfoDataContext
source : Table<SybilNode>
timer2_Tick
Try
[try]
Create SybilInfoDataContext
new SybilInfoDataContext()
OnCreated
OnCreated()
<<return>>
LINQ
Deferred Call
[Where (i.No...
[Select (i.Cre...
<<return>>
Create SybilInfoDataContext
new SybilInfoDataContext()
OnCreated
OnCreated()
<<return>>
LINQ
Deferred Call
[Select (k)]
[Single (n.No...
<<return>>
[catch (Exception)]


this : MediaPlay
btnPlay_Cli...
IsPlaying
IsPlaying(true)


this : AdminHome
Datas : SybilInfoDataContext
D : SybilNode
GetTable<TEntity> : Table<SybilNode>
button1_Cl...
Create SybilInfoDataContext
new SybilInfoDataContext()
OnCreated
OnCreated()
<<return>>
Create SybilNode
new SybilNode()
OnCreated
OnCreated()
<<return>>
InsertOnSubmit
Datas.SybilNodes.InsertOnSubmit(D)
<<return>>

5.2 SYSTEM TESTING

The philosophy behind testing is to find errors. The common view of testing is to bring the program without errors. Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and code generation. Once the source code has been generated, software must be tested to uncover as many errors as possible before delivery to the customer. In order to find the highest possible number of errors, tests must be conducted systematically and test cases must be designed using disciplined techniques.

5.2.1 Verification and Validation

Verification refers to the set of activities that ensures that system correctly implements a specific function. Validation refers to different set of activities that ensure that the system has been built is traceable to customer requirements. Verification and validation encompass a wide array of software quality assurance (SQA) activities that include formal reviews, quality, performance monitoring, documentation review, database review.

5.2.2 Unit Testing

Unit testing focuses verification effort on the smallest unit of software unit of software design, the module. Using the procedural design description as a guide, important control paths are tested to uncover errors within the boundary of the module.

5.2.3 Integration Testing

Once the modules are tested individually under the testing strategy, it is necessary to put all these modules together-interfacing. It is here that the data can be lost across the interface, one module can have an inadvertent, adverse effect on another.

Integration testing is a systematic technique for constructing the program structure while at the same time conducting test to uncover errors associated with interfacing. The objective is to take unit testing modules and build a program structure that has been dictated by design.

5.3 IMPLEMENTATION

Implementation means the process of converting a new or revised system design into an operational one. It is the most crucial stage in achieving a new successful system and in giving a confidence on the new system for the users that it will work efficiently and effectively. In this phase, we build the components either from scratch or by composition. Given the architecture document from the design phase and requirement document from the analysis phase, we can build exactly what has been requested.

This phase deals with issues of quality, performance, baselines, libraries and debugging. The end deliverable is the product itself. There are three types of implementation:

1. Implementation of a computer system to replace a manual system.
2. Implementation of new computer system to replace an existing one.
3. Implementation of a modified application to replace an existing one, using the same computer.

Implementation of our System comes under third category. At the end of the specific period, the system performance and the reliability are tested. Implementation is the key stage in achieving a successful new system.

CHAPTER 6

CONCLUSION

This project implements the SybilGuard, a novel decentralized protocol for limiting the corruptive influences of sybil attacks, by bounding both the number and size of sybil groups. SybilGuard relies on properties of the users' underlying social network, namely that (i) the honest region of the network is fast mixing, and (ii) malicious users may create many nodes but relatively few attack edges. In all our implementation experiments with multiple nodes, SybilGuard ensured that (i) the number and size of sybil groups are properly bounded for 99.8% of the honest users, and (ii) an honest node can accept, and be accepted by, 99.8% of all other honest nodes. Currently we are working on obtaining real social media to further validate SybilGuard.

REFERENCES

Papers:

[1] Center for Computational Analysis of Social and Organizational Systems (CASOS), 2006.
http://www.casos.cs.cmu.edu/computational tools/data.php.
[2] International Network for Social Network Analysis, 2006.
http://www.insna.org/INSNA/data inf.htm.
[3] I. Abraham and D. Malkhi. Probabilistic quorums for dynamic systems. In DISC, 2003.
[4] R. Bazzi and G. Konjevod. On the establishment of distinct identities in overlay networks. In ACM PODC, 2005.
[5] W. J. Bolosky, J. R. Douceur, D. Ely, and M. Theimer. Feasibility of a serverless distributed file system deployed on an existing set of desktop PCs. In ACM SIGMETRICS, 2000.
[6] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Gossip algorithms: Design, analysis and applications. In IEEE INFOCOM, 2005.

Books Referred:

1. Alex Homer , "**Professional C#.NET 1.1**", 2004, Wrox Publications
2. Steven Holzner, "**C#.NET Black Book**", 2003, Dreamtech Publications

3. Roger S Pressman, "**Software Engineering**", 2000, Dreamtech Publications

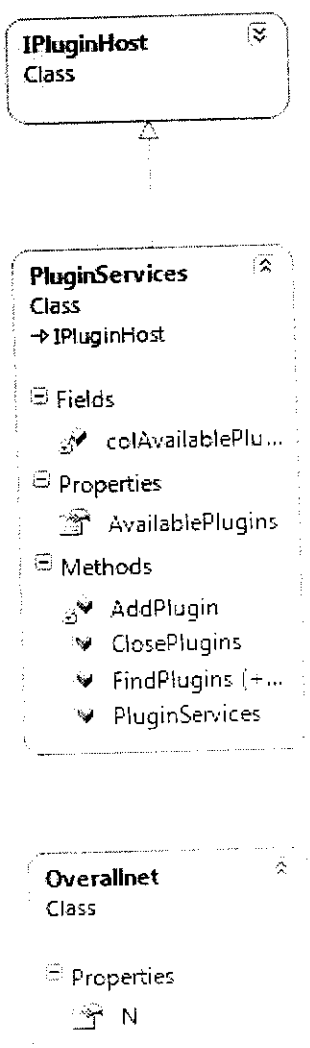
4. Karli Watson, Richard Anderson , "**Professional ASP.NET 1.1**" , 2004, Wrox Publications

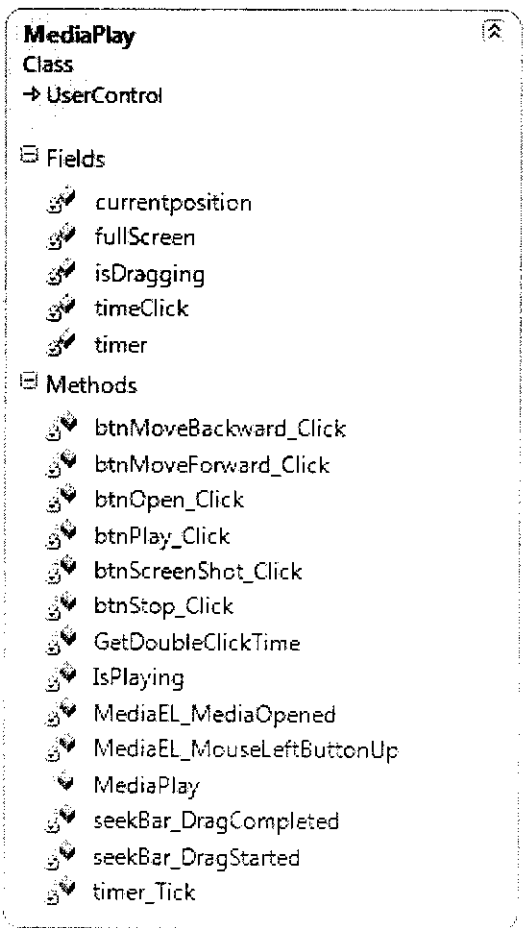
Websites:

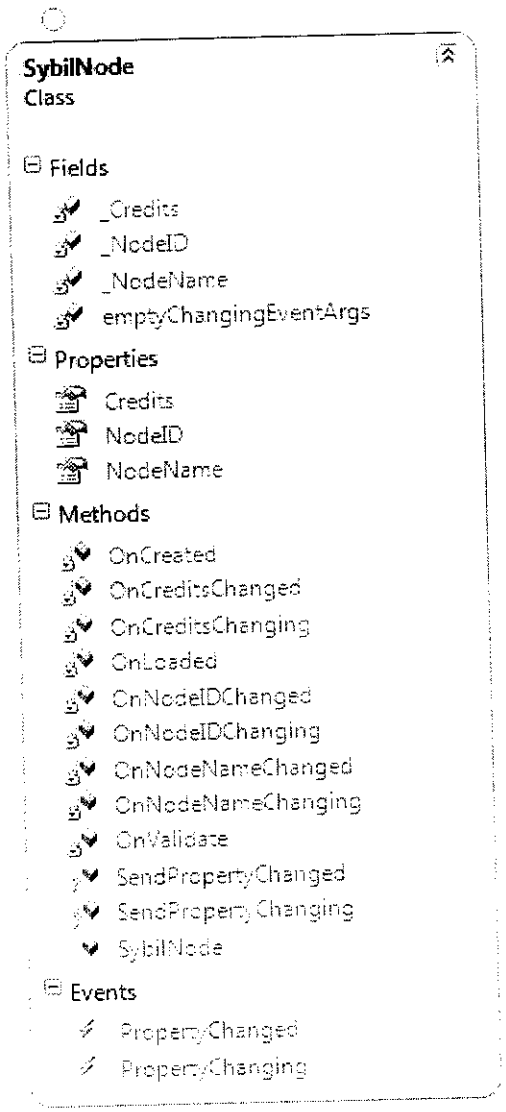
1. www.msdn.microsoft.com
2. www.vbcity.com
3. www.vbdotnetheaven.com
4. www.codeguru.com

APPENDIX 1

CLASSES AND PROPERTIES



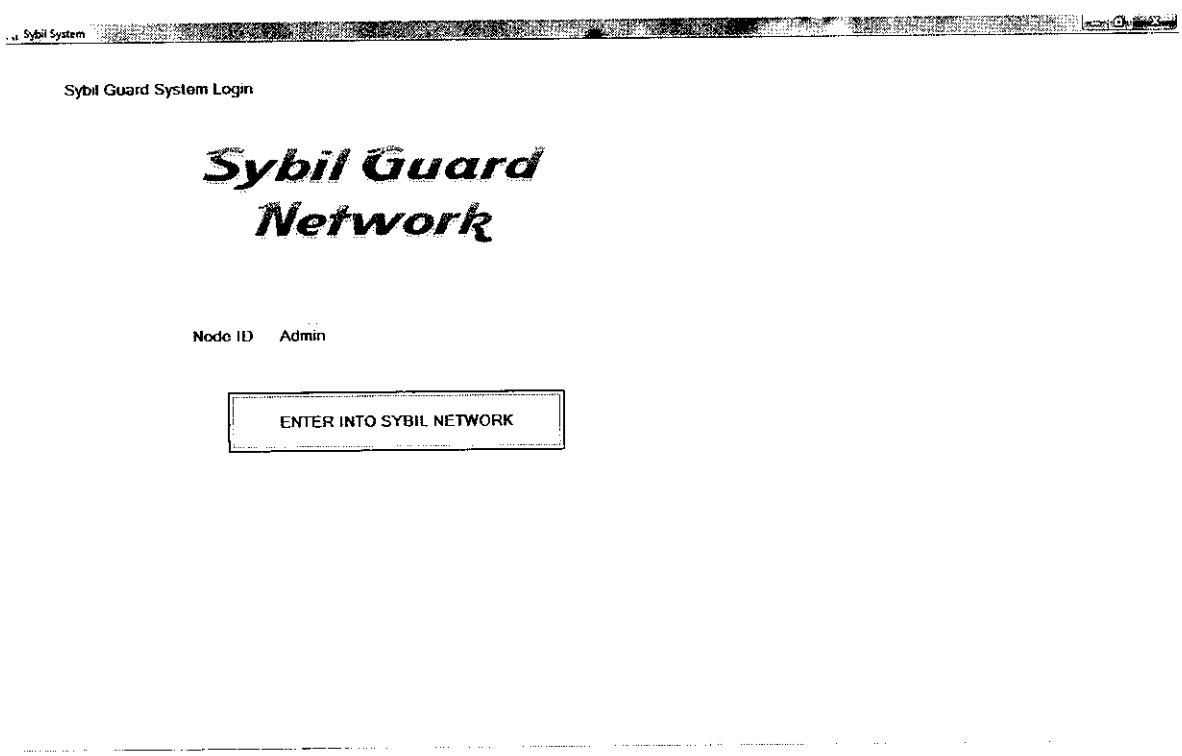

MediaPlay
Class
→ UserControl
Fields
currentposition
fullScreen
isDragging
timeClick
timer
Methods
btnMoveBackward_Click
btnMoveForward_Click
btnOpen_Click
btnPlay_Click
btnScreenShot_Click
btnStop_Click
GetDoubleClickTime
IsPlaying
MediaEL_MediaOpened
MediaEL_MouseLeftButtonUp
MediaPlay
seekBar_DragCompleted
seekBar_DragStarted
timer_Tick


SybilNode
Class
Fields
_Credits
_NodeID
_NodeName
emptyChangingEventArgs
Properties
Credits
NodeID
NodeName
Methods
OnCreated
OnCreditsChanged
OnCreditsChanging
OnLoaded
OnNodeIDChanged
OnNodeIDChanging
OnNodeNameChanged
OnNodeNameChanging
OnValidate
SendPropertyChanged
SendPropertyChanging
SybilNode
Events
PropertyChanged
PropertyChanging

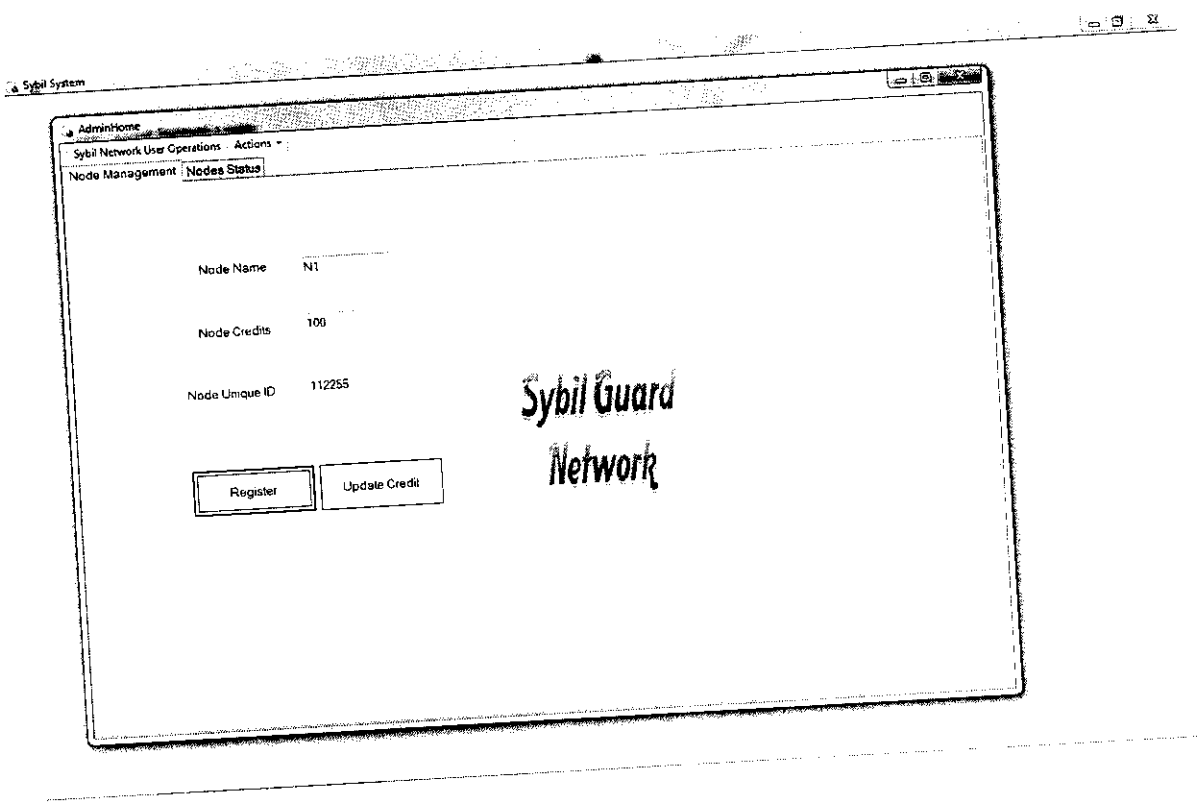
APPENDIX 2

SCREEN SHOTS

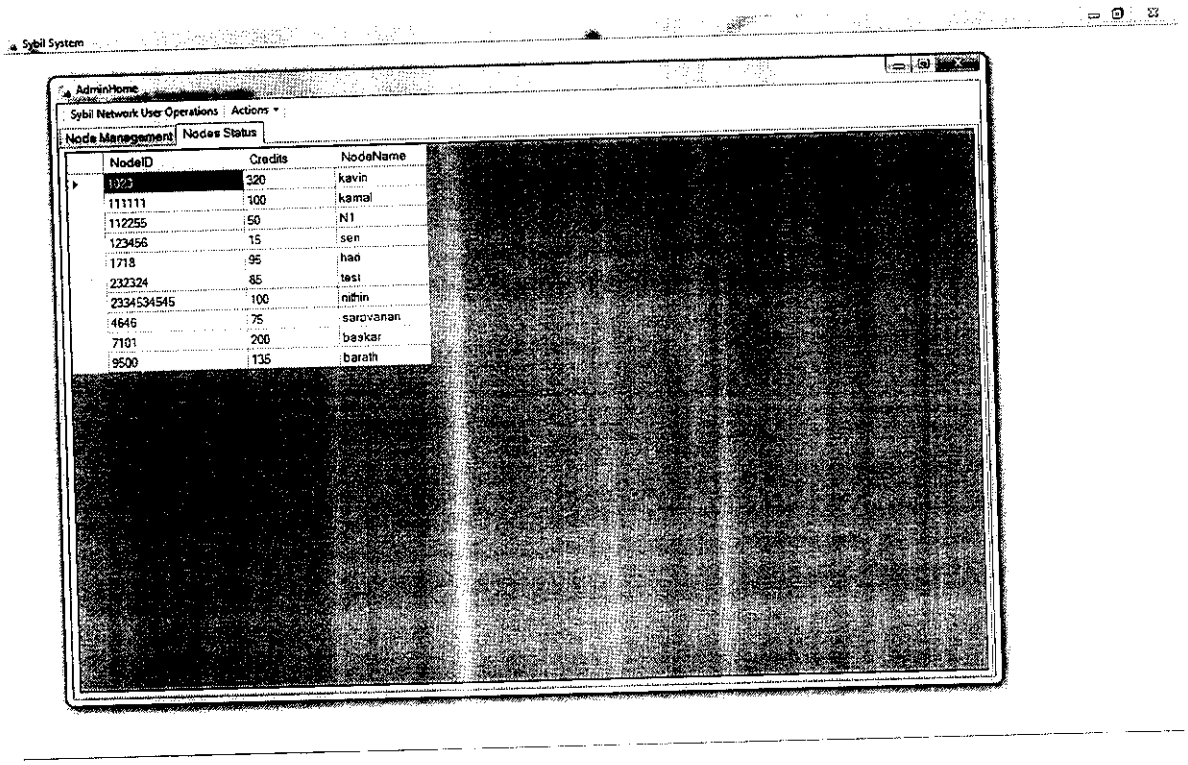
Home Screen



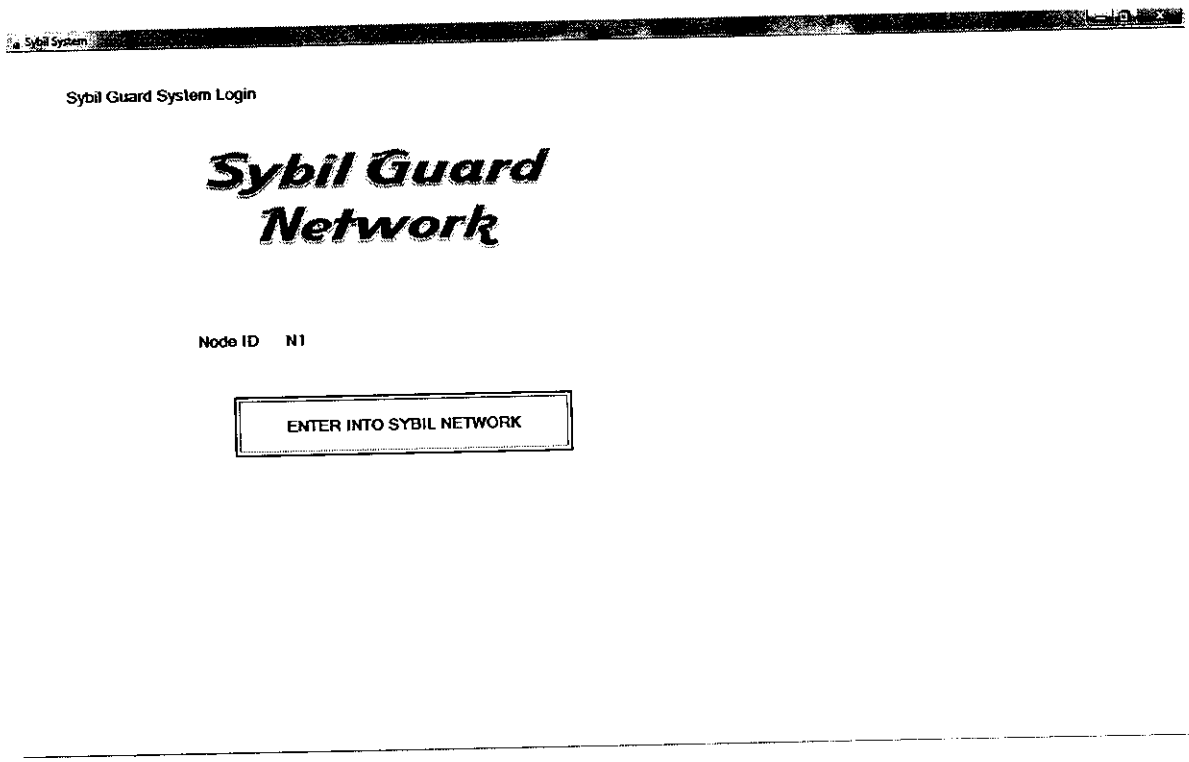
Admin Home


Sybil System
AdminHome
Sybil Network User Operations
Actions
Node Management
Nodes Status
Node Name
N1
Node Credits
100
Node Unique ID
112255
Register
Update Credit
Sybil Guard
Network

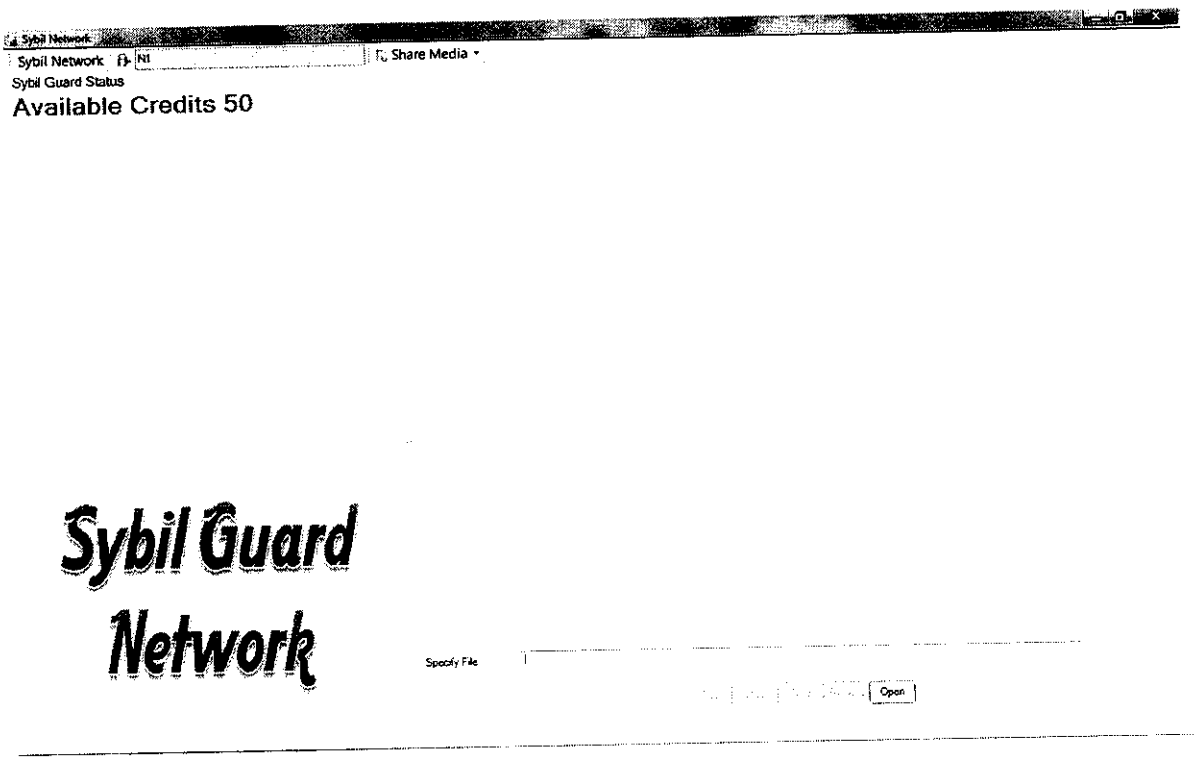
Node List


Sybil System
AdminHome
Sybil Network User Operations
Actions
Node Management
Nodes Status
NodeID
Credits
NodeName
1023
320
kavin
111111
100
kamal
112255
50
N1
123456
15
sen
1718
95
had
232324
85
test
2334534545
100
nithin
4646
75
sarpvanan
7101
200
baskar
9500
135
barath

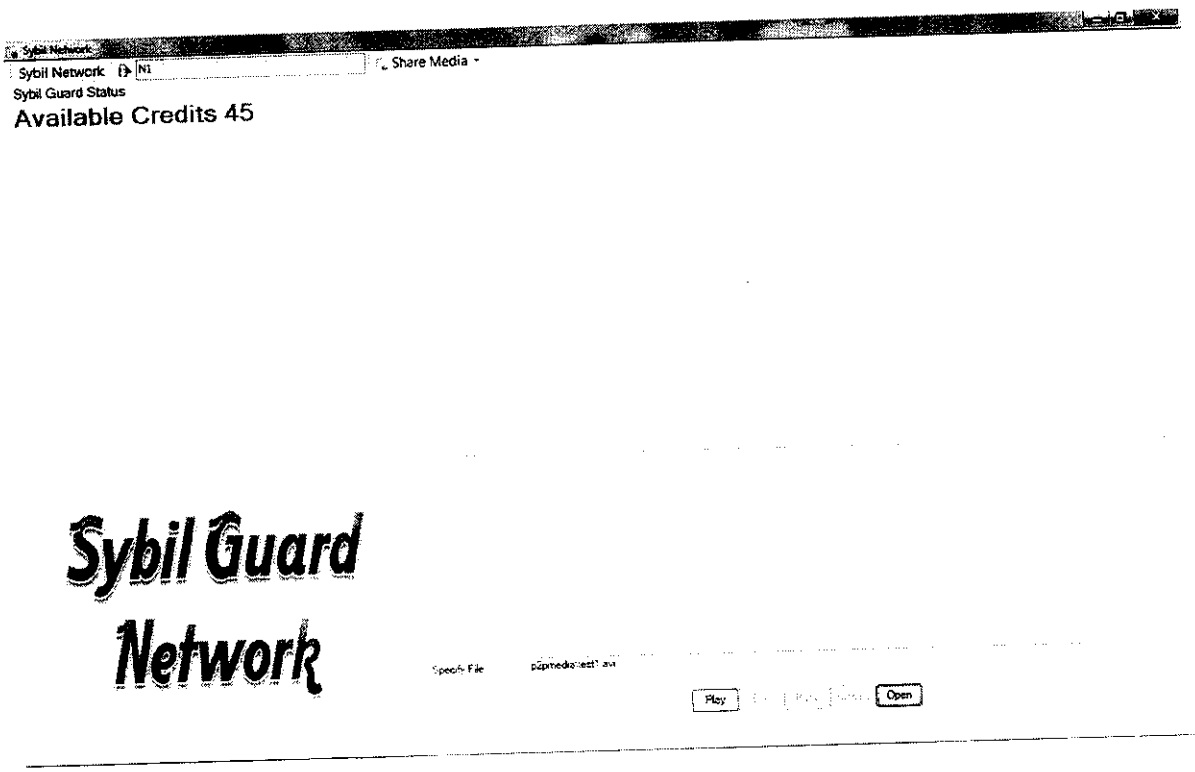
Node Login


Sybil System
Sybil Guard System Login
Sybil Guard Network
Node ID N1
ENTER INTO SYBIL NETWORK

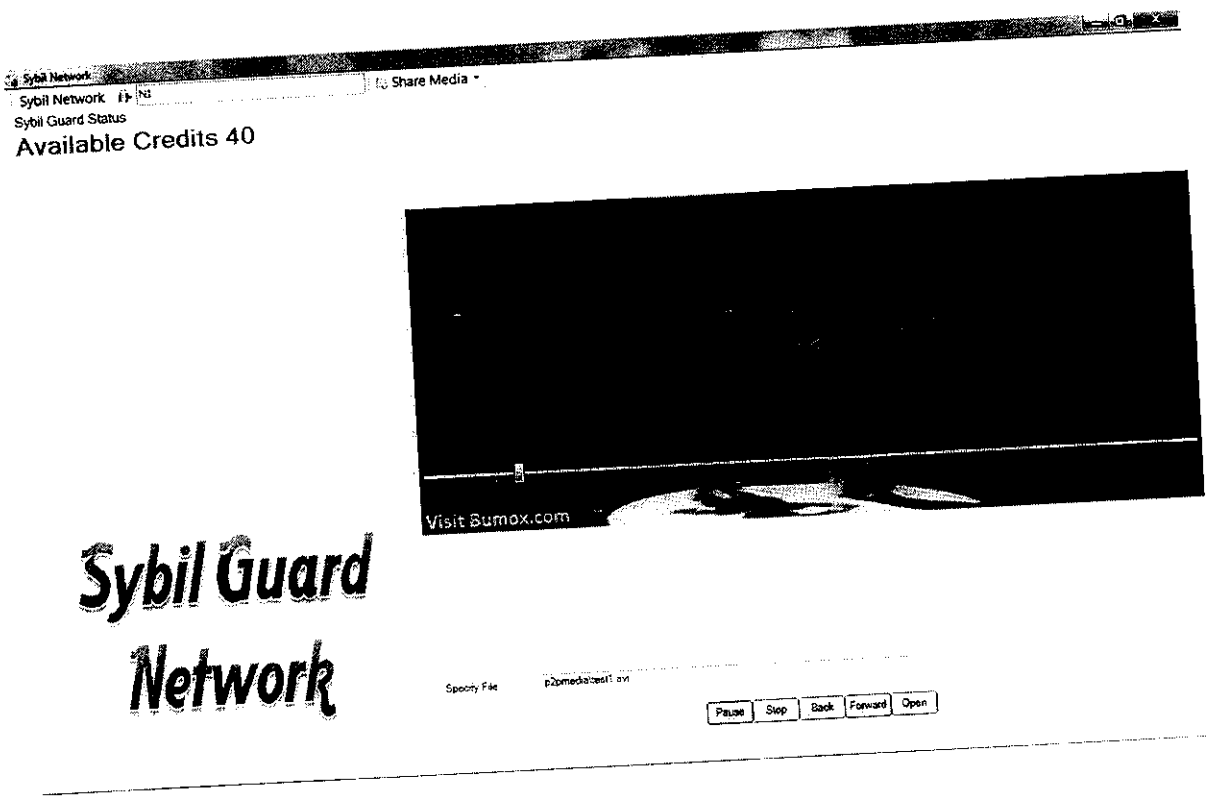
Node Home


Sybil Network
Sybil Network
NI
Share Media
Sybil Guard Status
Available Credits 50
Sybil Guard Network
Specify File
Open

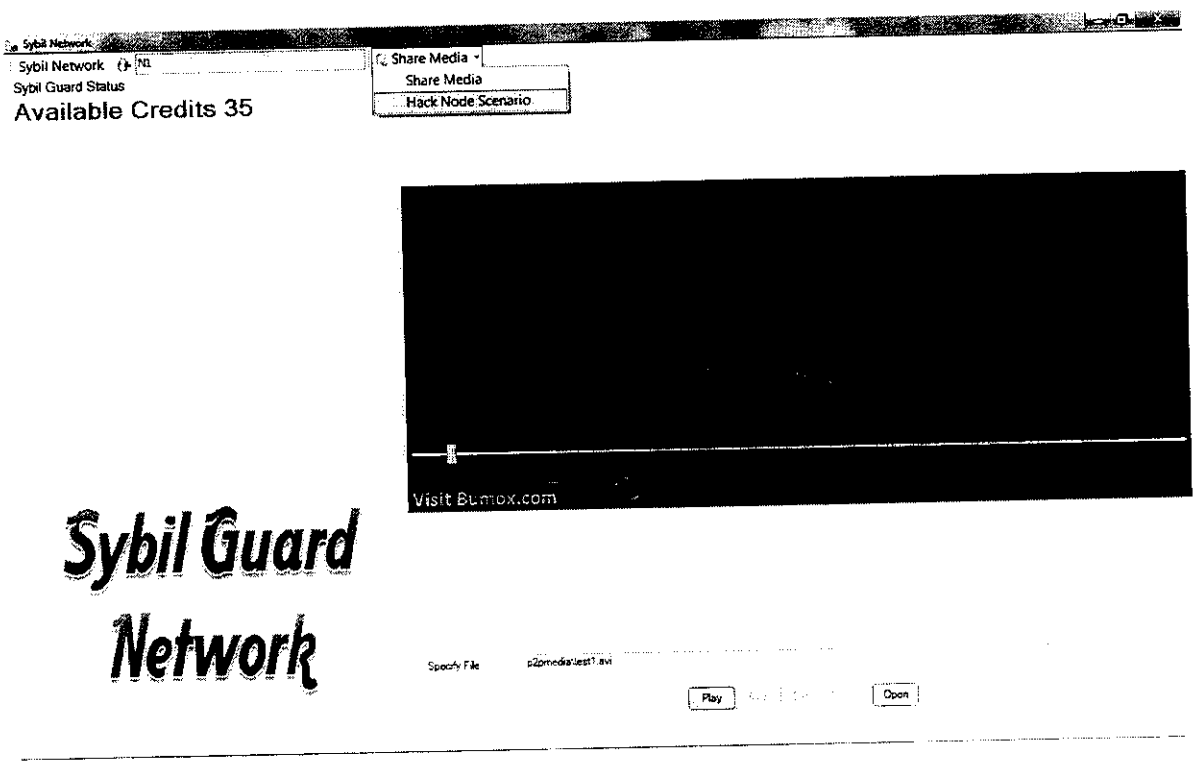
Specifying Access File



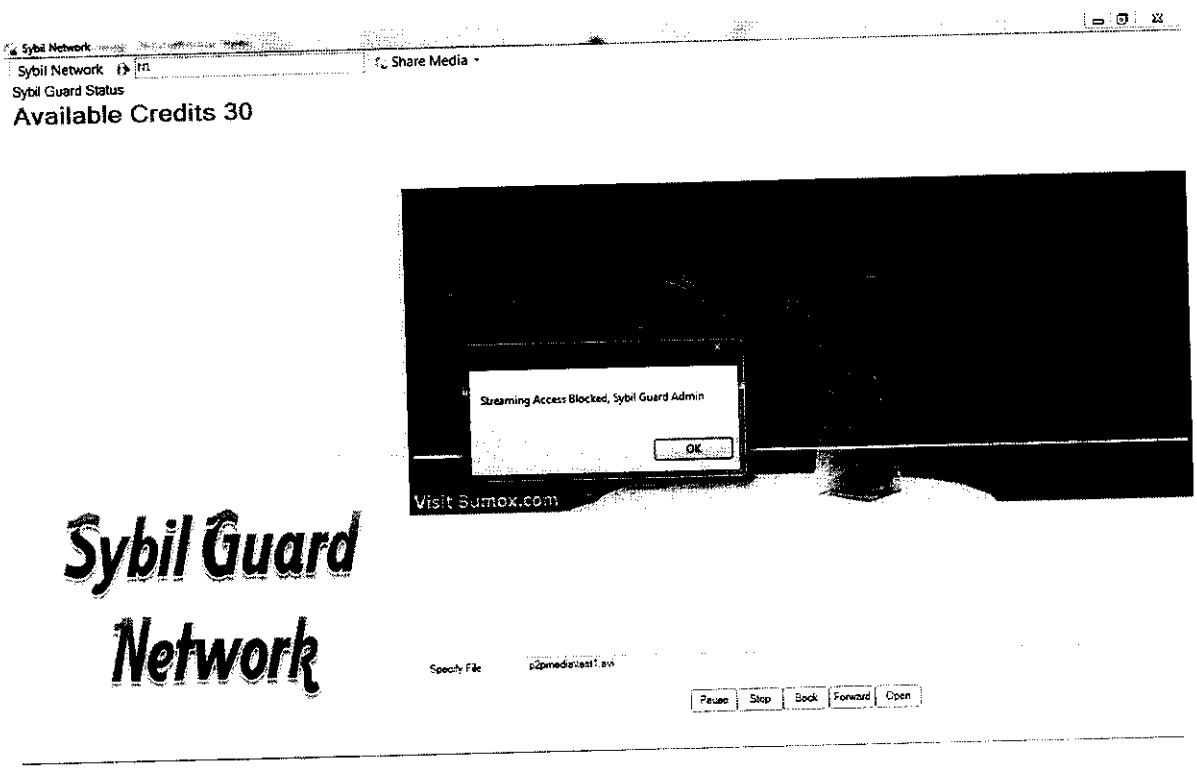
Streaming File



Sybil Network Attack



Sybil Guard


Sybil Network
Sybil Network
Share Media
Sybil Guard Status
Available Credits 30
Streaming Access Blocked, Sybil Guard Admin
OK
Visit Sumox.com
Sybil Guard Network
Specify File
Pause
Stop
Back
Forward
Open

CODING

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Diagnostics;
using System.IO;

namespace Reputation
{
    public partial class Form1 : Form
    {
        string uni = "";
        string logger = "";
        public Form1(string s)
        {
            InitializeComponent();
            logger = s;
            uni = "112255";
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            toolStripTextBox1.Text = logger;
            timer1.Enabled = true;
            timer2.Enabled = true;
            try
            {
                SybilInfoDataContext Datas = new SybilInfoDataContext();
                var d = (from k in Datas.SybilNodes
                         where k.NodeName == logger
                         select k.Credits).First();
                label1.Text = "Available Credits " + d.ToString();
            }
            catch (Exception ex)
            {

            }
        }

        private void toolStripButton1_Click(object sender, EventArgs e)
        {

        }

        private void viewCapacityToolStripMenuItem_Click(object sender, EventArgs e)
        {



        }

        private void shareFileToolStripMenuItem_Click(object sender, EventArgs e)
        {
```

```
        }

        private void chooseDownloadpathToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            OpenFileDialog D = new OpenFileDialog();
            D.ShowDialog();
            string file = D.FileName;
            //foreach (var file in System.IO.Directory.GetFiles(@"D:\keym"))
            {
                Upload("ftp://www.media.com/", "testuser", "test123", file);
            }

            MessageBox.Show("Media Shared ..");
        }

        private static void Upload(string ftpServer, string userName, string password,
string filename)
        {
            using (System.Net.WebClient client = new System.Net.WebClient())
            {
                client.Credentials = new System.Net.NetworkCredential(userName,
password);
                client.UploadFile(ftpServer + "/" + new FileInfo(filename).Name,
"STOR", filename);
            }
        }

        private void timer1_Tick(object sender, EventArgs e)
        {

            try
            {
                SybilInfoDataContext Datas = new SybilInfoDataContext();
                var d = from k in Datas.SybilNodes
                        where k.NodeName == logger && k.NodeID == uni
                        select k;
                int ct = d.Count();
                if (ct >= 1)
                {
                    SybilInfoDataContext Datas1 = new SybilInfoDataContext();
                    var d1 = (from k in Datas1.SybilNodes
                              where k.NodeName == logger && k.NodeID == uni
                              select k.Credits).First();
                    int crdt = Int32.Parse(d1.ToString());
                    if (crdt <= 10)
                    {
                        if (MessageBox.Show("Low Credit Level Streaming Access Blocked,
 Sybil Guard Admin") == System.Windows.Forms.DialogResult.OK)
                        {
                            Application.Exit();
                        }
                    }
                    else
                    {

                    }
                }
```

```
                else
                {
                    if (MessageBox.Show("Streaming Access Blocked, Sybil Guard Admin")
== System.Windows.Forms.DialogResult.OK)
                    {
                        Application.Exit();
                    }
                }
            }
            catch (Exception ex)
            {
            }
        }

        private void timer2_Tick(object sender, EventArgs e)
        {
            try
            {
                SybilInfoDataContext Datas1 = new SybilInfoDataContext();
                var d1 = (from k in Datas1.SybilNodes
                          where k.NodeName == logger && k.NodeID == uni
                          select k.Credits).First();
                int crdt = Int32.Parse(d1.ToString());

                crdt = crdt - 5;

                SybilInfoDataContext Datas = new SybilInfoDataContext();
                var D = (from k in Datas.SybilNodes select k).Single(p => p.NodeID ==
uni);
                D.Credits = crdt.ToString();
                Datas.SubmitChanges();
                label1.Text = "Available Credits " + crdt.ToString();

            }
            catch (Exception ex)
            {
            }
        }

        private void hackNodeScenarioToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            uni = "2334534545";
        }
    }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
```

```
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using System.IO;
using System.Runtime.InteropServices;
using System.Windows.Controls.Primitives;
using System.Windows.Threading;

namespace Reputation
{
    /// <summary>
    /// Interaction logic for MediaPlay.xaml
    /// </summary>
    public partial class MediaPlay : UserControl
    {
        DispatcherTimer timer;
        public MediaPlay()
        {
            InitializeComponent();
            IsPlaying(false);
            timer = new DispatcherTimer();
            timer.Interval = TimeSpan.FromMilliseconds(200);
            timer.Tick += new EventHandler(timer_Tick);
            MediaEL.LoadedBehavior = MediaState.Manual;
        }


        private void btnOpen_Click(object sender, RoutedEventArgs e)
        {
            if (textBox1.Text.Contains(".mp3"))
            {
                image1.Visibility = System.Windows.Visibility.Visible;
            }
            else
            {
                image1.Visibility = System.Windows.Visibility.Hidden;

            }
            MediaEL.Source = new Uri("http://www.media.com/" + textBox1.Text);
            btnPlay.IsEnabled = true;
        }

        #region IsPlaying(bool)
        private void IsPlaying(bool bValue)
        {
            btnStop.IsEnabled = bValue;
            btnMoveBackward.IsEnabled = bValue;
            btnMoveForward.IsEnabled = bValue;
            btnPlay.IsEnabled = bValue;
            btnScreenShot.IsEnabled = bValue;
            seekBar.IsEnabled = bValue;
        }
        #endregion

        #region Play and Pause
        private void btnPlay_Click(object sender, RoutedEventArgs e)
        {
            IsPlaying(true);
            if (btnPlay.Content.ToString() == "Play")
```

```
            {
                MediaEL.Play();
                btnPlay.Content = "Pause";
            }
            else
            {
                MediaEL.Pause();
                btnPlay.Content = "Play";
            }
        }
        #endregion

        #region Stop
        private void btnStop_Click(object sender, RoutedEventArgs e)
        {
            MediaEL.Stop();
            btnPlay.Content = "Play";
            IsPlaying(false);
            btnPlay.IsEnabled = true;
        }
        #endregion

        #region Back and Forward
        private void btnMoveForward_Click(object sender, RoutedEventArgs e)
        {
            MediaEL.Position = MediaEL.Position + TimeSpan.FromSeconds(10);
        }

        private void btnMoveBackward_Click(object sender, RoutedEventArgs e)
        {
            MediaEL.Position = MediaEL.Position - TimeSpan.FromSeconds(10);
        }
        #endregion

        #region Open Media
        //private void btnOpen_Click(object sender, RoutedEventArgs e)
        //{
        //    System.Windows.Forms.OpenFileDialog ofd = new
System.Windows.Forms.OpenFileDialog();
        //    ofd.Filter = "Video Files (*.wmv)|*.wmv";
        //    if (ofd.ShowDialog() == System.Windows.Forms.DialogResult.OK)
        //    {
        //        MediaEL.Source = new Uri(ofd.FileName);
        //        btnPlay.IsEnabled = true;
        //    }
        //}
        #endregion

        #region Capture Screenshot
        private void btnScreenShot_Click(object sender, RoutedEventArgs e)
        {
            byte[] screenshot = MediaEL.GetScreenShot(1, 90);
            FileStream fileStream = new FileStream(@"Capture.jpg", FileMode.Create,
FileAccess.ReadWrite);
            BinaryWriter binaryWriter = new BinaryWriter(fileStream);
            binaryWriter.Write(screenshot);
            binaryWriter.Close();
        }
        #endregion
```

```
        #region Seek Bar
        private void MediaEL_MediaOpened(object sender, RoutedEventArgs e)
        {
            if (MediaEL.NaturalDuration.HasTimeSpan)
            {
                TimeSpan ts = MediaEL.NaturalDuration.TimeSpan;
                seekBar.Maximum = ts.TotalSeconds;
                seekBar.SmallChange = 1;
                seekBar.LargeChange = Math.Min(10, ts.Seconds / 10);
            }
            timer.Start();
        }

        bool isDragging = false;

        void timer_Tick(object sender, EventArgs e)
        {
            if (!isDragging)
            {
                seekBar.Value = MediaEL.Position.TotalSeconds;
                currentposition = seekBar.Value;
            }
        }

        private void seekBar_DragStarted(object sender, DragStartedEventArgs e)
        {
            isDragging = true;
        }

        private void seekBar_DragCompleted(object sender, DragCompletedEventArgs e)
        {
            isDragging = false;
            MediaEL.Position = TimeSpan.FromSeconds(seekBar.Value);
        }
        #endregion

        #region FullScreen
        [DllImport("user32.dll")]
        static extern uint GetDoubleClickTime();

        System.Timers.Timer timeClick = new
System.Timers.Timer((int)GetDoubleClickTime())
        {
            AutoReset = false
        };

        bool fullScreen = false;
        double currentposition = 0;

        private void MediaEL_MouseLeftButtonUp(object sender, MouseButtonEventArgs e)
        {
            if (!timeClick.Enabled)
            {
                timeClick.Enabled = true;
                return;
            }

            if (timeClick.Enabled)
```

```
            {
                if (!fullScreen)
                {
                    LayoutRoot.Children.Remove(MediaEL);
                    this.Background = new SolidColorBrush(Colors.Black);
                    this.Content = MediaEL;

                    MediaEL.Position = TimeSpan.FromSeconds(currentposition);
                }
                else
                {
                    this.Content = LayoutRoot;
                    LayoutRoot.Children.Add(MediaEL);
                    this.Background = new SolidColorBrush(Colors.White);

                    MediaEL.Position = TimeSpan.FromSeconds(currentposition);
                }
                fullScreen = !fullScreen;
            }
        }
        #endregion
    }

    #region Extension Methods
    public static class ScreenShot
    {
        public static byte[] GetScreenShot(this UIElement source, double scale, int
quality)
        {
            double actualHeight = source.RenderSize.Height;
            double actualWidth = source.RenderSize.Width;
            double renderHeight = actualHeight * scale;
            double renderWidth = actualWidth * scale;

            RenderTargetBitmap renderTarget = new RenderTargetBitmap((int)renderWidth,
                (int)renderHeight, 96, 96, PixelFormats.Pbgra32);
            VisualBrush sourceBrush = new VisualBrush(source);

            DrawingVisual drawingVisual = new DrawingVisual();
            DrawingContext drawingContext = drawingVisual.RenderOpen();

            using (drawingContext)
            {
                drawingContext.PushTransform(new ScaleTransform(scale, scale));
                drawingContext.DrawRectangle(sourceBrush, null, new Rect(new Point(0,
0),
                    new Point(actualWidth, actualHeight)));
            }
            renderTarget.Render(drawingVisual);

            JpegBitmapEncoder jpgEncoder = new JpegBitmapEncoder();
            jpgEncoder.QualityLevel = quality;
            jpgEncoder.Frames.Add(BitmapFrame.Create(renderTarget));

            Byte[] imageArray;

            using (MemoryStream outputStream = new MemoryStream())
            {
                jpgEncoder.Save(outputStream);
```

```
                imageArray = outputStream.ToArray();
            }
            return imageArray;
        }
    }
    #endregion
}
```