Modelling Virus Propagation in P2P Networks

Muhammad Faheem Rasheed (MS-CS)
Department of Computer Science & Information Technology
The University of Lahore, Pakistan.

Abstract:

Peer-to-peer (P2P) networking is currently an emerging technology that has gained dramatic increase in popularity both for network traffic and in the research domain. The growing popularity and use of P2P networks means that P2P has also attracted the attention of virus creators. A number of viruses have recently appeared that are especially designed to propagate through P2P networks. The purpose of this paper aims to provide a study on existing P2P simulators and selects an optimum simulator to implement a virus model that determines the spread of viruses via pure P2P networks. The defined model is a customized form of SEI (Susceptible-Exposed-Infected) model based on the study of epidemiology. The virtual behavior of fizzer virus is used in modeling virus propagation. Derivation of several differential equations is given that predicts the expected behavior of a P2P network. The derived equations are verified with the implementation of virus propagation model in a P2P simulator.

Keywords: Virus Propagation, Peer-to-Peer, fizzer Virus, PeerThing.

Introduction:

Peer-to-peer (P2P) technology is not a new theory for computer networks industry. P2P technology is based on the phenomena to share computer resources and services by direct communication. The idea of direct communication has become the major source of popularity of P2P networks. P2P is currently used in wide variety of applications such as file sharing, games, instant messaging and content distribution. P2P networks are mainly used for sharing files containing data, audio and video in digital format [1]. There are several types of P2P networks that include Gnutella [2], Napster [3], Kazaa [4], BitTorrent [5] and many more. The growing popularity and use of P2P networks means that P2P has also attracted the attention of virus creators. P2P networks always have to deal with serious security issues, as they are prone to malicious codes such as viruses and worms. While P2P networks can be prone to variety of traditional viruses, there are many viruses that specifically target the performance of a P2P network. Some of those viruses and worms include Swen [4], Fizzer worm [4], LirvaA [4] and LirvaB [4]. Security concerns are always associated to any computer network irrespective of being a local network or a network connected to the Internet.

Commercially used P2P networks consist of millions of users that are connected with millions of computer systems. In such a huge network, large number of file transfers takes place in a fraction of a second. In such conditions, a file containing a malicious code is likely to propagate to thousands of computer systems in no time. This leads to the need of a system that can predict the spread of viruses in P2P networks. The designed system must possess the capability of modeling different virus behaviors in different types of networks. Attempts have been made in the past for the development of systems that can be used to predict virus propagation, but most of the designed systems are based on numerical calculations and experiments. Nowadays, development of a system such as a computer network requires pre-testing being performed on a network simulator to make an assessment on the requirements and performance of that network. Similarly, a system developed in a P2P network simulator is required to analyze virus propagation in P2P networks.

Security Threats in P2P networks:

Viruses and worms are always major security threats to all kinds of computer networks, especially P2P networks. Worms are special type of viruses that have the capability of replicating itself and finding its way to spread around a network. A worm in a P2P network is generally an independent program that propagates throughout the network from one node to another. The main aim of a P2P worm is to affect the hosts in P2P networks. There are many viruses and worms that affect P2P networks below is a brief description of major known P2P viruses as stated in [4,5].

Swen:

Swen [4] virus is one of the famous P2P viruses that mainly affect P2P networks such as Kazaa. Is also known as Win32.Swen.A@mm. Swen virus is considered as a very dangerous virus for the following reasons:

1· Propagates through email, Instant Messaging (IM), Internet chat rooms.

2· Propagates through P2P applications by making multiple copies of itself and placing them in the shared folder chosen from a list of filenames

3· Generally comes in a hidden form of Microsoft Patch

4· Mails itself to email addresses selected from the victim's email address book and sent folders

5· Introduces false errors in windows Messaging API (MAPI) and asks users for confidential information such as password, SMTP server, account, etc.

6· Aliases: W32/Gibe@mm

Fizzer Worm:

Fizzer [4] worm is similar to Swen virus and is also one of the most popular viruses in P2P networks. It is also known as Win32.Fizzer.A@mm. The basic behaviour of fizzer includes the following:

1· Propagates through email, Instant Messaging (IM), Internet chat rooms

2· Propagates through P2P applications by making multiple copies of itself and placing them in the shared folder chosen from a list of filenames
3· Contains backdoor and key logger abilities
4· Backdoor component uses Mirc and AOL Instant Messenger (AIM) to allow the instigator of virus to issue commands on victim's computer
5· Keylogger stores data being input by the victim and hence allowing the virus author to view confidential data such as password, account details, etc
6· Attempts to shutdown anti-virus programs
7·Aliases:W32/Fizzer-A,I-Worm.Fizzer,W32.HLLW.Fizzer @ mm,W32/Fizzer@MM

Lirva A and Lirva B:

Lirva A and Lirva B [4] viruses mostly affect Kazaa P2P networks. These viruses are much dangerous as compared to the ones mentioned above. The main behaviour of this virus includes:
1· Attempts to de-activate firewall and anti-virus applications.
2· Scans Outlook address books and sent folders for addresses to mail itself on to
3· Virus can be distributed through the following files: Resume.exe, Download.exe, MSO-Patch-0071.exe, MSO-Patch-0035.exe, Two-Up-Secretly.exe, Transcripts.exe, Readme.exe, AvrilSmiles.exe, AvrilLavigne.exe, Complicated.exe, Singles.exe, Sophos.exe, Cogito_Ergo_Sum.exe, CERT-Vuln-Info.exe, Sk8erBoi.exe, IAmWiThYoU.exe.

Magic Eightball:

Magic Eightball [4] is known to be a trojan virus that infects P2P networks such as Kazaa. Magic Eightball comes in a packaged Zip file known as eightball2.zip. Its functionalities are as follows:
1• Attempts to delete files on C: directory
2• Once executed, trojan creates a batch file that contains instructions to delete files from the root directory
3• Displays a series of dialogue boxes and when OK is pressed on the last dialogue box then the virus activates.

Benjamin.A and Benjamin.B:

The Benjamin [4] worms are a strong security threat to P2P networks. These worms are occasionally found to attack Kazaa networks without their authorization or awareness.
The behavior of this virus includes:
1· Display of fake error messages such as "Access error #03A:94574: Invalid pointer operation File possibly corrupted."
2· Once Worms then creates hundreds of copies of its files and places them in the shared folder of all the users in the network
3· Worms also display the web site benjamin.xww.de.

Lolol.a:

Lolol [5] is another worm that spreads rapidly in a P2P network. This worm is similar to fizzer worm in behavior and has reported to infect many Kazaa file-sharing networks. Basic information about Lolol is as follows:
1· Consists of a powerful backdoor routine
2· Backdoor routine connects to IRC channel where it executes commands send by the author of the worm.
3· Worm is a 60 KB Windows PE .exe file and written in Microsoft Visual C++

Related Work on P2P Virus Propagation:

Study of virus propagation in P2P networks takes its initiation from a biological field of epidemiology. Epidemiology as defined in [6] is the study of how diseases spread in human population. There have been several papers [6, 7, 8, 9] published on virus propagation in P2P networks. In [6], how an infected file propagates through a P2P network is discussed. For the prediction of propagation they have used a modified version of a three state S-E-I (Susceptible-Exposed-Infected) model. The analysis is based on numerical calculations of the proposed model equations with some assumptions. Such analysis cannot be proved to be 100% correct unless performed on a P2P network simulator. The research paper in [8] is the modified version of the work mentioned in [6] by the same group of people. In this paper, the additional work includes the study of dissemination of polluted files and the release of P2P viruses.
Extension in modeling includes the modeling of online/offline behavior of peers and the modeling of peers that remain infected. Data for the evaluation of this modified model is taken from a practical P2P network. The analysis is again made on the numerical calculations and also from simulation results but there isn't any particular P2P simulator mentioned. The research paper in [9] presents the threats caused by P2P worms and possible mechanisms to avoid such threats. They formed a basic model in order to perform virus propagation and presented a theoretical analysis. The theoretical analyses are also supported by experiments performed on P2P graphs generated by a P2P simulator but the name of the simulator is not mentioned. They further developed an epidemic simulator that takes the P2P topology graph and the probability of node being a guardian node and generates the corresponding output representing the rate of change of infected nodes along time. There isn't sufficient information about the functionalities provided by the epidemic simulator and the basis on which the output is generated. There is also lack of information on the performance of guardian nodes, if they themselves get infected.

P2P Virus Propagation Model:

The first step in analyzing the behavior of a virus requires a model. The objective of the defined model is to predict the expected spread of a virus via P2P networks and the changes it makes to the state of peers. Virus Propagation model described in [6] is used in this modeling technique with some amendments.

Let's assume a P2P network comprising of total number of N peers. Each peer in the P2P network can either be in one of the following three basic states:

• **Vulnerable (V):** Peers vulnerable to download a file

infected with virus
• **Exposed (E):** Peers that have downloaded one or more infected files
• **Infected (I):** Peers that have executed an infected file.
An infected peer in the P2P network can be further classified as:

- **Peer State (PS):** State of the infected peers in terms of online or offline

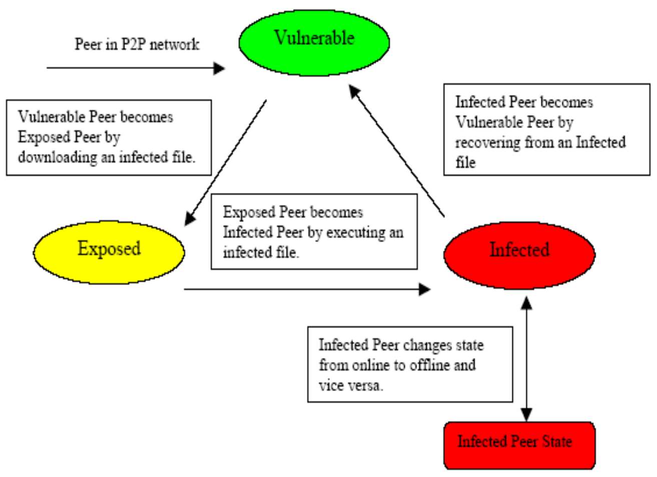
When the above-mentioned variables are measured in terms of time t, then the following assumptions can be made:

• **V (t)** represents number of vulnerable peers in time t
• **E (t)** represents number of exposed peers in time t
• **I (t)** represent number of infected peers in time t
• **PS (t)** represents number of infected peers in online/offline states in time t

Since the total number of peers in the P2P network is fixed at N, then for all values of t, N can be expressed by N = V (t) + E (t) + I (t). The above-mentioned time-varying variables are affected by several individual events that include

• A peer downloading a file from another peer,
• A peer executing a shared file,
• An infected peer changing its state and
• An infected peer recovering,
• ωV = Mean rate, in files per minute, at which each peer downloads new files.
• ωE = Mean rate, in files per minute, at which each peer executes shared files.
• ωR = Mean rate, in recoveries per minute, at which infected peers recover.

The mean rate at which these events occur are represented by the following parameters:


Peer in P2P network
Vulnerable
Vulnerable Peer becomes Exposed Peer by downloading an infected file.
Infected Peer becomes Vulnerable Peer by recovering from an Infected file
Exposed
Exposed Peer becomes Infected Peer by executing an infected file.
Infected
Infected Peer changes state from online to offline and vice versa.
Infected Peer State

P2P Simulators:

Survey has been made on several existing P2P simulators. These simulators are specially designed for evaluating P2P network performances. The list of simulators that were surveyed includes: P2Psim, PeerSim, 3LS, Neurogrid, GPS, Query Cycle Simulator, RealPeer and PeerThing. In order to evaluate the performance of each simulator and choose the optimum to implement there is some evaluation criteria in which simulator's architecture, usability, documentation, scalability, extensibility, statistics and traffic modelling has been measured.

Comparison:

The comparison between all the surveyed simulators. The assessment has been presented in way to analyze a simulator at a glance. The key aspects of each criterion stated are summarized. The summary highlights the main feature of each principle to facilitate a reader to grasp a bird's eye view of each simulator. Table illustrates the summary of all simulators in terms of their evaluation performed on the criteria mentioned.

Simulators	Architecture	Usability & Documentation	Scalability	Extensibility	Statistics	Traffic Modelling
P2Psim	Discrete event	Command Prompt / Poor documentation	Up to 3000 nodes for Chord protocol	Limited protocol extensibility but complex to handle	Limited set of statistics	Low
PeerSim	Discrete event	Command prompt / Detailed API documentation but lacks detail in user manual	Highly Scalable up to 10^6 nodes supported	Designed to be extensible	Works required to gather statistics	None
3LS	Discrete event	GUI / No documentation except the research paper	Low scalability of 20 nodes	Theoretically extensible but practical implementatio n is required	N/A	N/A
NeuroGrid	Discrete event for unstructured networks	GUI-based / Good API documentation and user manual but source code is not commented well	Around 300,000 nodes are claimed	Designed to be extensible	Not easy to interpret the results	Low
GPS	Discrete event for both structured and unstructured networks	GUI-based / Poor documentation	Up to 512 nodes tested	Lack of good documentation limits extensibility	Lack of information for statistical analysis	Low
Query Cycle	Discrete event	GUI-based / Poor API documentation and user manual	Highly scalable up to 10^6 nodes supported	Limited	Limited to simulator's predefined analysis	N/A
RealPeer	Discrete Event	Command Prompt / Well commented source code but poor API documentation and user manual	Around 20,000 peers can be simulated	Lack of good API documentation does not support extensibility	Limited to simulator's analysis and hard to understand	N/A
PeerThing	Discrete event driven for both unstructured and structured networks	GUI-based / Good User Manual but source code is not well commented and poor API documentation	Gnutella up to 2000 (tested with 600) nodes & Napster up to 1000	Extensible with and without changes made in source code	Set of simulator's visualisation & external visualisation also supported	High

Chosen Simulator:

All the simulators mentioned in the above table consist of several pros and cons. Now a simulator has to be chosen in order to model virus propagation in P2P networks. The simulator that can be chosen to achieve our goal is

PeerThing. Following describes the reason for choosing this simulator.

Simulator Architecture:

Simulator architecture features of PeerThing simulator that made it preferable on other simulator architectures. The architecture is primarily based on the criterion to support discrete event simulations. The main components of the architecture are divided into two major categories: "system behaviour" and "system scenario".

The system behaviour allows defining the behaviour of each peer in a P2P network and scenario allows defining the environment of the P2P network. Additional behaviours can also be added and used by calling the behaviour with CallBehaviour attribute. The resource allocation for each peer is also defined in the scenario. After a network has been setup, its corresponding code is generated in XML for both system behaviour and system scenario. The architecture of PeerThing is well defined as compared to the architectures of rest of the surveyed simulators.

Usability:

PeerThing provides a GUI interface for its users to interact with the system. A user after modelling the system behaviour and scenario can run simulations in the simulation window. Simulation takes the system behaviour and scenario details as input and generates the output in a specified log file. PeerThing has the capability to run multiple simulations for a specified number of time steps or number of messages. The results for the number of messages generated and the nodes involved in the generation of those messages are shown by default on the results window. The generated results can also be converted to .csv format and opened in Microsoft Excel.

Documentation:

The user manual for PeerThing as compared to all the surveyed simulators can be considered as the best user manual. It has detailed information on how to use the simulator. It contains a step-by-step guide to build a basic network and view its corresponding results.

The authors didn't provide any API documentation with the simulator source code. The source code is not well commented and too complex to understand as there are many things interconnected with each other. But the user manual contains sufficient information for a user to model a basic network that lacked in all the surveyed simulators.

Scalability:

The authors have claimed that this simulator can simulate up to 2000 nodes for Gnutella network and 1000 nodes for Napster model. The simulator has been tested to support up to 700 nodes for Gnutella network with its proper functionalities.

Extensibility:

The extensibility of the simulator code is an extremely hard task to do as the source code is not well commented. But extensibility can be performed in the behaviour of the provided Gnutella and Napster basic models. The default Gnutella behaviour provided in the simulator only performs searching mechanism and does not provide any functionality for download operations if the searched file is found.

Statistics:

The results generated from the simulator are always stored in a log file before another simulation is performed. The results generated in form of tables and their corresponding graphs in the simulator visualization are limited to visualization provided by the simulator. Such results are not sufficient enough for analysis on subjects not supported by the simulator. But the capability of exporting files in .csv format allows a much detailed analysis to be done on the generated results.

Traffic Modelling:

Traffic modelling is basically the flow control of events and messages being performed in a network. PeerThing allows traffic modelling in the system behaviour of a network. Traffic can be modelled with the help of state of peers, the transitions a peer uses to go from one state to another and the tasks that it performs within a state. The capability of defining different traffic models allows the creation of networks with different behaviours. This feature is one of the most critical features that are supported by PeerThing and none of the other simulators proved to be a better option in this regard.

Simulations:

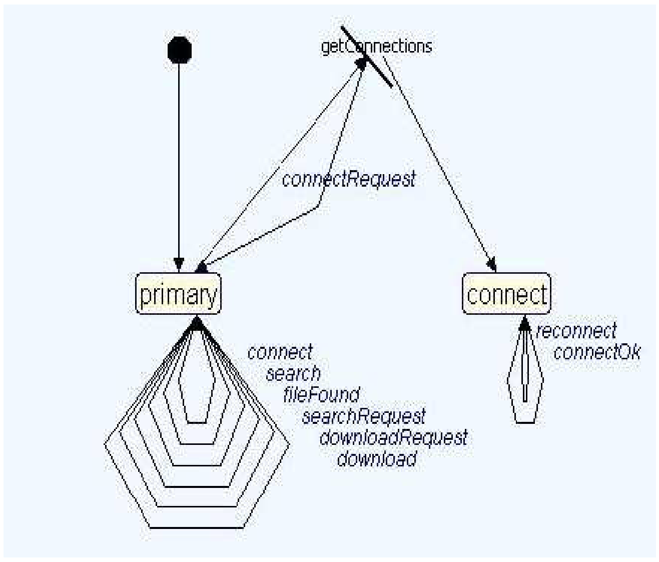
In this part of the paper implementation of virus propagation model in PeerThing simulator presents. Further presents a simulation performed on a basic Gnutella network in PeerThing simulator. Next, virus propagation model is implemented in Gnutella model then shows the results achieved from the simulator and finally analyses are presented on the achieved results.

Gnutella Simulation:

Gnutella type P2P network is a pure P2P network. Hence it is important to implement the virus model in a Gnutella network. But before the implementation of virus model, a proper implementation of Gnutella network is required.

Experimental Setup:

In this section, the experimental setup of Gnutella model in PeerThing simulator is presented. Let's assume that the total number of peers in a Gnutella P2P network is N and the simulation runs for a time period T. TTL = 7 is used for the search requests i.e. a search request will pass through a maximum number of 7 hops and if the requested file is not found then the message would be dropped with a message "filenotfound". Initially a peer is defined to have at least 2 connections that can go to a maximum of 6 connections. As discussed earlier, Gnutella model consists of servents that act as both, as peers and servers. The behaviour of servents is defined in the system behaviour editor of the simulator. Illustrates the behaviour of a single peer in a Gnutella network.



Peer Behaviour in Gnutella Model

Gnutella network developed in PeerThing makes use of states and transitions for defining the behaviour of peers. States are both source and destination of transitions. There are two distinct states in this Gnutella model namely primary state and connect state. The primary state is the starting state for all the nodes in the network. It selects a peer randomly from the available peers and stores them into a list. The selected peer becomes a host peer and a connection request is made to that peer from other peers. The connect state represents the nodes that are in connected state and are ready to perform any requested service. Transitions specify the change between states. There are several transitions in a Gnutella model that include: connectRequest, connect, reconnect, connectOk, search, fileFound, searchRequest, downloadRequest and download.

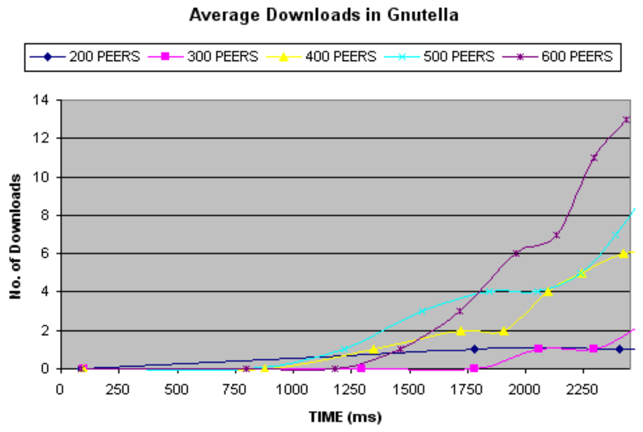
Working:

This section provides the working behind the Gnutella model in PeerThing is discussed When the simulation begins, all the peers gets their status to online and starts making connection requests to randomly selected host peers in the network. Connections are established between the host peers and the requesting peers whenever a host peer successfully replies to the requesting peer. This corresponds to the ping and pong operations of Gnutella. The connected peers then become the neighbours of the host peers. After the connections have been established, next comes the search request for a desired file. A peer randomly selects a file from the list of resources in the network and places a search request for that file to its neighbouring peers.

This operation is similar to the query operation performed in a practical Gnutella network. The neighbouring peers after receiving the search requests then search for that file in the list of resources allocated to them; if the requested peer contains the desired file then it sends a message to the requesting peer that the requested file has been found. Along with the message that the file has been found, the identification of the node containing the file is sent to the query initiator. This corresponds to the QueryHit operation of Gnutella network. If the requested peer doesn't contain the requested file then it forwards the search request to its neighbouring peers until the requested file has been found or all the connected neighbours have been searched for the file.

Results and Analysis:

Results of several sample simulations performed to analyze the basic behaviour of a Gnutella network. The simulations are performed to view the average number of downloads that takes place in a Gnutella network. In a practical network, as the number of nodes increase the average number of downloads is also expected to increase. The simulations were performed on different number of peers ranging from 200 to 600 with different time steps.



Average Downloads in Gnutella Network

The above figure shows the results obtained from the simulator and plotted on Microsoft Excel. The obtained results shows that average downloading starts after 1000 ms. If the result at time stamp 2000 ms is observed then it shows that a network comprising of 200 and 300 peers downloads only a single file. But as the size of the network increases to 400 peers then the download operations rises to 3. Similarly for 500 and 600 peers, the average number of downloads that takes place are 4 and 6 respectively. To summarise the acquired results, it can be said that as the size of a Gnutella type P2P network increases then the average number of downloads that takes place also increases.

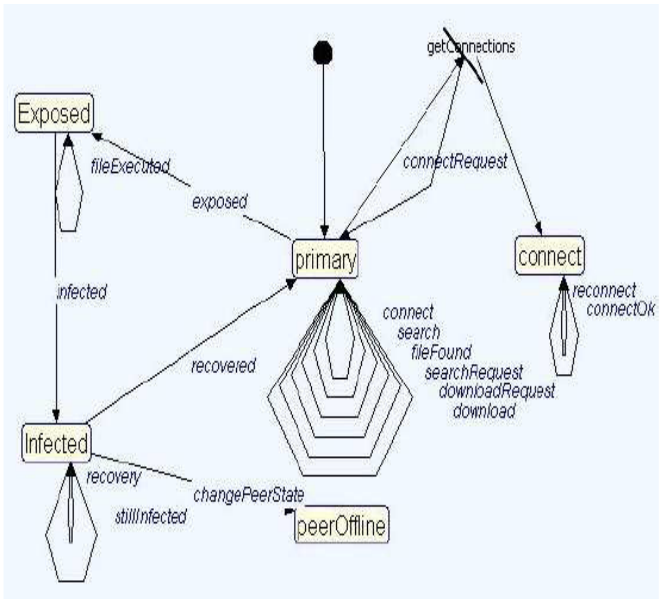
Gnutella Simulation with Virus Model:

In this section, modification can be made to the basic Gnutella network to implement virus propagation in P2P networks on PeerThing. Next section describes the experimental setup of the virus model, and then explains the working mechanism of the virus model. Furthermore illustrates the results achieved from the simulations and their corresponding analyses are presented in last.

Experimental Setup:

The principal experiment setup of Gnutella network is the same for the virus model. There is an addition of several states and transitions to implement the virus model. The new states in this model are exposed, infected, peerOffline and the primary state are considered to be as the susceptible state. Below is the diagram that displays the model to implement virus propagation. The exposed state is used to represent the exposed peers.

When a peer downloads an infected file then it goes from susceptible (primary) state to exposed state through an exposed transition. The infected state represents the infected peers. When a peer executes the exposed file then it gets infected and goes to the infected state through infected transition. The peerOffline state is used to represent the peers that are no longer online. A peer goes offline whenever it is infected with a deadly virus that kills the processes of anti-virus programs. When such virus attacks a peer then it is virtually unusable and hence goes to peerOffline state. Figure displays the behaviour of a servent for modelling virus propagation



Peer Behaviour in Gnutella Virus Propagation Model

There are several additional transitions used in this model to implement virus propagation. The list of additional transitions is given below:

- **Exposed**
- **Infected**
- **Recovery**
- **StillInfected**
- **ChangePeerState**
- **Recovered**

Working:

Basic assumptions defined for Gnutella model described above are also applicable to this model. The main supposition for this model is that all the downloaded files are considered to have virus embedded in them. This means that whenever a peer downloads a file then it automatically goes from susceptible state to exposed state as the downloaded file already contains the virus. Virus modelling in PeerThing makes of the virtual behaviour of virus analyzed from the actual behaviour of virus. The working mechanism for this model starting from the connection setup till the download procedure is the same as the basic Gnutella model.

As mentioned earlier the peer that performs download operation gets exposed. The exposed peer is supposed to have executed the downloaded file after certain time interval. When the file gets executed then the virus inside the file becomes active and starts to infect the peer. After infecting the peer that downloaded the file, the virus then looks for the neighbouring peers and infects all the connected peers. In the next step, the infected peer tries to recover itself from the virus by going through a recovery process. In case of generic virus when the recovery procedure is applied, the infected file is removed from the list of resources allocated to that peer and the peer gets recovered and comes back to vulnerable state.
On the other hand, if a peer is infected with a fizzer virus then the recovery procedure fails to recover the infected peer and the peer remains infected as the fizzer virus has the capability to shuts down the processes of anti-virus programs. When a peer remains infected with a fizzer virus for a certain time period then it is considered to be in a state where it is virtually unusable. At this moment, the infected peer goes from the infected state to an offline state that means the peer is no longer active and disconnects itself from its neighbours.

Simulator Limitations:

PeerThing simulator similar to other simulators, along with several productive features also holds several limitations. First of all, the simulator is not highly scalable as when tested with more than 600 nodes, after roughly 3000 ms of simulation time, it starts to give java heap space error i.e. the simulator does not possess enough memory to run the complete simulation successfully. When tested with lower number of nodes such as 200 then the maximum simulation time achieved was 5000 ms. In order to achieve longer simulation time, simulations have to be performed in short span of time intervals. But even this technique does not last long as the simulator runs out of memory generate the java heap space error. Due to uncertain time periods of simulation run, multiple simulations have to be performed to achieve optimum results and also to verify that the obtained results are correct.
Another limitation of PeerThing is that whenever an error is generated, a pop-up appears that states the error and also mentions that for more details view the console but there isn’t console associated with the simulator. The behaviours defined in the system scenario of PeerThing occasionally pretend as if they do not take part in the simulations at all. The user manual lacks details for the proper use of system scenario. The java heap space error can be handled with some extension made to the source code of the simulator. Since the simulator does not come with complete set of source code so it is not possible to extend the code for the simulator. Such limitations caused the most of the simulations to be performed for a limited time period of about 3000 ms.

Conclusion:

This paper illustrates a contribution to the study of virus propagation in P2P networks. As stated in the introduction, this study is of immense importance as even the best security enabled networks find it hard to guard themselves against the attack and spread of deadly malicious codes. The purpose of this paper was to model virus propagation in pure P2P networks using a P2P simulator. Different researchers have contributed in this field with models designed and implemented on numerical observations. But a lack of research material was found in the implementation of virus propagation model in an appropriate P2P simulator. The report started off with an overview of P2P systems in terms of its classifications, applications and major security threats. Next, a model was used to analyze virus propagation in P2P networks. After modelling virus propagation in P2P networks, a survey was presented on the existing P2P simulators. Virus model was then implemented on one of the surveyed simulator that proved to be the best among all the examined simulators. Finally, the expected results of virus propagation model were verified in the chosen simulator. This paper mainly provides an ideology to people responsible for network design and its security, so that they can test the level of their networks' possibility for being prone to virus propagation prior to practical implementation. It is always better to define a strategy to cope with virus propagation beforehand rather than waiting for the threat to attack.

Future Recommendations:

The study of virus propagation in P2P networks is still in its early ages and requires a lot of research to be done in this domain. This paper may further be implemented with different probabilities assigned to a file being infected and an infected file being executed in virus propagation model. Other extensions to this paper can be the implementation of the virus propagation model in other popular P2P networks such as Napster, Kazaa, BitTorrent etc, in PeerThing simulator. The simulator used in this paper can also be extended to overcome the limitations for providing a better environment for the evaluation of P2P networks and model virus propagation. Implementation of virus model can be also performed in any other simulator that can implement the model with complete functionalities.

References:

[1] David Barkai (2000, February). An Introduction to Peer-to-Peer Computing, Retrieved June 22, 2007, from Peer-to-Peer Architecture Group, Microcomputer Research Lab, Intel Corporation, Project Web site: www.intel.com/technology/magazine/systems/it02012.pdf

[2] Patrick Kirk (2003). Gnutella File Sharing And Distribution Network, Retrieved June 28, 2007, The Gnutella Homepage: http://rfc-Gnutella.sourceforge.net/

[3] Napster (2007). Retrieved June 28, 2007, The Napster Homepage: http://www.Napster.com/

[4] Kazaa (2006). Retrieved June 28, 2007, from Sharman Networks, The Kazaa Homepage: http://www.Kazaa.com

[5] BitTorrent (2006). Retrieved June 28, 2007, The BitTorrent Homepage: http://www.BitTorrent.org

[6] Yi Lin (2004). Peer-to-Peer Systems, Retrieved July 15, 2007, from McGill University, School of Computer Science http://www.cs.mcgill.ca/~cs577/lectures/577-p2p-slides.pdf

[7] OECD (Organisation for Economic Co-operation and Development (2004, October). Peer-to-Peer Networks in OECD countries, Retrieved August 5, 2007 http://www.oecd.org/dataoecd/55/57/32927686.pdf

[8] Miguel Castro, Manuel Costa and Antony Rowstron (2004). Peer-to-peer overlays: structured, unstructured, or both? Retrieved August 5, 2007, from Microsoft Research, Cambridge, UK Project Web site: research.microsoft.com/~antr/MS/Structella-tr.pdf

[9] Choon Hoong Ding, Sarana Nutanong, and Rajkumar Buyya. Peer-to-Peer Networks for Content Sharing, Retrieved June 20, 2007, from Grid Computing and Distributed Systems Laboratory, Department of Computer Science and Software Engineering, The University of Melbourne, Australia http://www.gridbus.org/papers/P2PbasedContentSharing.pdf

[10] Zupeng Li, Daoying Huang, inrang Liu, Jianhua Huang (2003). Research of Peer-to-Peer Network Architecture, Retrieved July 20, 2007, from National Digital Switching System Engineering & Technological R&D Center, Airforce Engineering University, China Project Web site: http://ieeexplore.ieee.org/iel5/8586/27215/01209091.pdf?tp=&arnumber=1209091&isnumber=27215

[11] Jian Liang, Keith W. Ross, Rakesh Kumar (2004, May). Understanding Kazaa, Retrieved June 20, 2007, from Polytechnic University, USA http://cis.poly.edu/~ross/papers/UnderstandingKazaa.pdf

[12] Paul L. Piccard, Brian Baskin, Craig Edwards, George Spillman (2006). Securing IM and P2P Applications for the Enterprise. In Marcus H.Sachs (Technical Editor), Jaime Quigley, Chapter 11 BitTorrent, (pp 285 – 312), USA and Canada ISBN: 1597490172

[13] Virus List (2007). Retrieved August 5, 2007, from All About Internet Security, Kaspersky Lab Project Web site: www.viruslist.com/en/viruses/encyclopedia

[14] R.W. Thommes, M.J. Coates (2005). Modeling Virus Propagation in Peer-to- Peer Networks, Retrieved June 22, 2007 from Department of Electrical and Computer Engineering, McGill University ieeexplore.ieee.org/iel5/11110/35625/01689197.pdf?tp=&isnumber=&arn umber=1689197

[15] Krishna Ramachandran, and Biplab Sikdar (2006). Modeling Malware Propagation in Gnutella Type Peer-to-Peer Networks, Retrieved June 22, 2007, from Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, USA http://networks.ecse.rpi.edu/~bsikdar/papers/ipdps06.pdf

[16] Richard Thommes and Mark Coates. Epidemiological Modelling of Peer-to-Peer Viruses and Pollution, Retrieved June 22, 2007, Department of Electrical and Computer Engineering McGill University, Canada ieeexplore.ieee.org/iel5/4146652/4146653/04146754.pdf?tp =&isnumber=&arnumber=4146754

[17] Lidong Zhou, Lintao Zhang, Frank McSherry, Nicole Immorlica, Manuel Costa, and Steve Chien. A First Look at Peer-to-Peer Worms: Threats and Defenses, Retrieved June 22, 2007, from Microsoft Research Cambridge and University of Cambridge research.microsoft.com/users/lidongz/p2pworm.pdf
[18] Howrey Simon Arnold & White, LLP, Lisa Jose Fales, Esq.Charles Webb, Esq., The CapAnalysis Group, LLC James C. Miller III, Ph.D. Jeffrey A. Eisenach, Ph.D. Peer-to-Peer Software Providers' Liability Under Section 5 of the FTC Act, Retrieved July 5, 2007, from Washington, DC http://ipcentral.info/blog/P2P%20White%20Paper.doc
[19] Dan Sullivan (2005, April). Threats to Enterprise Security, Retrieved July 5, 2007, from Security Docs
Project Web site: http://www.securitydocs.com/library/3211
[20] F-Secure (2007). F-Secure Virus Descriptions: Fizzer, Retrieved July 5, 2007 Project Web site:http://www.f-secure.com/v-descs/fizzer.shtml
[21] Stephen Naicken, Anirban Basu, Barnaby Livingston, Sethalat Rodhetbhai, Ian Wakeman (2006). Towards Yet Another Peer-to-Peer Simulator, Retrieved June 25, 2007, from Department of Informatics, University of Sussex, Brighton, U.K. www.comp.brad.ac.uk/het-net/tutorials/P37.pdf
[22] Stephen Naicken, Anirban Basu, Barnaby Livingston and Sethalat Rodhetbhai(2006). A Survey of Peer-to-Peer Network Simulators, Retrieved June 25, 2007, from Network Lab, Department of Informatics, University of Sussex, Brighton, U.K. www.xtremebytes.com/~abasu/d-docs/paperpgnet2006_p2psimsurvey.pdf
[23] S. Naicken B. Livingston A. Basu S. Rodhetbhai I. Wakeman D. Chalmers (2007, April). The State of Peer-to-Peer Simulators and Simulations, Retrieved June 25, 2007, Department of Informatics, University of Sussex, U.K. informatics.sussex.ac.uk/research/groups/softsys/papers/ccr07.pdf
[24] Nyik San Ting. A Generic Peer-to-Peer Network Simulator, Retrieved June 25, 2007, from Department of Computer Science, University of Saskatchewan http://bistrica.usask.ca/MADMUC/Pubs/ting880.pdf
[25] Thomer M. Gil, Frans Kaashek, Jinyang Li, Robert Morris, Jeremy Stribling (2005). P2Psim: A Simulator for Peer-to-Peer (P2P) Protocols, Retrieved June 25, 2007
Project Web Site: http://pdos.csail.mit.edu/p2psim
[26] Chord (2006, December). Retrieved June 25, 2007 The Chord Homepage: http://pdos.csail.mit.edu/chord/
[27] Jinyang Li, Jeremy Stribling, Robert Morris, and M. Frans Kaashoek (2005). Bandwidth-efficient management of DHT routing tables, Retrieved June 25, 2007, from MIT Computer Science and Artificial Intelligence Laboratory, http://pdos.csail.mit.edu/~jinyang/pub/nsdi05-accordion.pdf
[28] Indranil Gupta, Ken Birman, Prakash Linga, Al Demers, Robbert van Renesse (2003). Kelips: Building an Efficient and Stable P2P DHT Through Increased Memory and Background Overhead, Retrieved June 25, 2007, Cornell University, Ithaca, NY, USA
Project Web Site: http://iptps03.cs.berkeley.edu/final-papers/kelips.pdf
[29] Chimera (2006). Retrieved June 25, 2007 The Tapestry Homepage: http://current.cs.ucsb.edu/projects/chimera/
[30] Petar Maymounkov, David Mazieres (2002). Kademlia: A Peer-to-peer Information System Based on the XOR Metric, Retrieved June 25, 2007 http://citeseer.ist.psu.edu/529075.html
[31] Mark Jelasity, Alberto Montresor, Gian Paolo Jesi, Spyros Voulgaris (2006, November). PeerSim: A Peer-to-Peer Simulator, Retrieved June 25, 2007 Project Web Site: http://peersim.sourceforge.net/
[33] Sam Joseph (2001, August). Neurogrid, Retrieved June 25, 2007 Project Web Site: www.neurogrid.net/
[34] Weishuai Yang and Nael Abu-Ghazaleh. GPS: A General Peer-to-Peer Simulator and its Use for Modelling BitTorrent, Retrieved June 25, 2007, Department of Computer Science, Binghamton University Project Web Site: www.cs.binghamton.edu/~wyang/gps/mascots05.pdf
[35] Tyson Condie, SepandarKamvar, Mario T. Schlosser, Beverly Yang. Stanford P2P Sociology Project, Retrieved June 25, 2007 Project Web Site: http://p2p.stanford.edu
[36] Mario T. Schlosser, Tyson E. Condie, and Sepandar D. Kamvar. Simulating a File-Sharing P2P Network, Retrieved June 25, 2007, from Department of Computer Science, Stanford University, Stanford, USA http://www.stanford.edu/~sdkamvar/papers/simulator.pdf
[37] Dieter Hildebrandt (2007, February). RealPeer - A Framework for the Simulation-based Development of Peer-to-Peer Systems, Retrieved June 25, 2007
Project Web Site: http://www.p2parch.de/
[38] Dieter Hildebrandt Dieter, Ludger Bischofs and Wilhelm Hasselbring (2007,February). RealPeer - A Framework for the Simulation-based Development of Peer-to-Peer Systems, Retrieved June 25, 2007
http://ieeexplore.ieee.org/iel5/4135239/4135240/04135315.pdf?tp=&arnumber=4135315&isnumber=4135240
[39] Project Group "PeerThing" (2006, September). PeerThing, Retrieved July 5, 2007, from Software Engineering Group, Department of Computing Science University of Oldenburg, Uhlhornsweg, Germany
Project Web Site: http://se.informatik.uni-oldenburg.de/lehre/projectgroups/pgp2p/
[40] Sepandar D. Kamvar, Mario T. Schlosser, and Hector GarciaMolina (2003, May). The EigenTrust Algorithm for Reputation Management in P2P Networks, Retrieved June 25, 2007, from Stanford University Project Web Site: http://www.stanford.edu/~sdkamvar/papers/eigentrust.pdf

Faheem: is currently working as System Engineer in a DUNYA news TV a leading News Channel in Pakistan. He has done BS in Computer Engineering from Sir Syed UET Karachi, Pakistan in 2006; Later on he has done his MS in Computer Science from The University of Lahore, Pakistan in 2012. He has more than 6 years of Practical Experience in different organizations.