

DEPARTMENT OF
APPLIED PHYSICS AND ELECTRONICS
UMEÅ UNIVERSITY, SWEDEN



DIGITAL MEDIA LAB

A Study on Anonymous P2P Networks and their Vulnerabilities

Dharma Reddy Manda
e-mail: dhma0001@student.umu.se

SUPERVISOR: APOSTOLOS GEORGAKIS

Report Date: January 16, 2007

Abstract

Peer-to-Peer Networks is a newly emerging paradigm in the communication era, replacing the popular client-server model. This thesis work is a study of various existing anonymous P2P networks and also about the attacks on these networks. Initially, we discuss about various possible attacks on P2P networks and the proposed solutions to overcome these attacks. Also, we discuss about anonymous P2P networks starting with anonymity in P2P networks discussing various anonymous mechanisms like crowds, mixes, onion routing, and antnet routing. Finally, we investigate and describe briefly various existing anonymous P2P networks that are available.

Keywords

Peer-to-peer, overlay, structured, anonymity, security, privacy, attacks, vulnerabilities.

Contents

1	Introduction	1
1	Classification of Peer-to-Peer Networks	1
2	Advantages of P2P Networks	2
3	History	3
4	Attacks on P2P Networks	3
5	Anonymous P2P Systems	4
6	Technical Terms	4
7	Thesis Organization	4
2	Attacks on P2P Networks	5
1	Denial-of-Service Attack	5
1.1	Solutions	6
2	Man-in-the-Middle Attack	7
2.1	Solutions	7
3	Sybil Attack	8
3.1	Solutions	8
4	Eclipse Attack	9
4.1	Solutions	10
5	Rational Attack	10
5.1	Solutions	11
6	Attacks as nodes live or join	11
6.1	Solutions	11
7	Intersection attack	11
7.1	Solutions	11
8	Identity Attack	12
8.1	Solutions	12
9	Spamming Attack	12
9.1	Solutions	12

10	Poisoning	12
10.1	Solutions	13
11	Trickle Attack	13
11.1	Solutions	13
12	Worm Propagation	13
12.1	Solutions	14
3	Anonymity and Anonymous Mechanisms	16
1	Anonymity	16
2	Various existing Anonymous Mechanisms:	17
2.1	Mixes	17
2.2	Crowds	18
2.3	Onion Routing	19
2.4	Antnet Routing	20
4	Various Anonymous P2P Networks	23
1	MUTE	23
2	Anonymizing Peer-to-Peer Proxy	24
3	Free Haven	25
4	Freenet	26
5	Ants	26
6	Anonymous Peer-to-peer File Sharing	27
7	WASTE	28
8	GNUnet	28
9	Mantis	29
10	I2P	29
11	Entropy	30
12	Nodezilla	30
5	Acknowledgement	31
	References	31

Chapter 1

Introduction

Peer-to-Peer Networks in short known as P2P Networks is a newly emerging paradigm in the communication era. The popularity of P2P networks has been increased tremendously in recent years. The reason for this popularity lies in the services provided by these networks by utilizing the resources of their peers.

Stephenous and Diomidis, provides more elaborated definition of Peer-to-Peer Networks [53]:

Peer-to-Peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage and bandwidth, capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance without requiring the intermediation or support of a global centralized server or authority.

Wikipedia provides the general definition of P2P networks [1]:

A Peer-to-Peer computer network is a network that relies primarily on the computing power and bandwidth of the participants in the network rather than concentrating it in a relatively low number of servers.

This means that, the clients which, participate in the network will perform the operations cooperating with each other rather than relying on a single central server. *In short, it is a system with completely decentralized self-organization and resource usage.*

Steinmetz and Wehrle in [12] describes Peer-to-Peer as both a technology for file-sharing and also as a fundamental design principal for distributed systems.

1 Classification of Peer-to-Peer Networks

One possible classification is based on the degree of centralization [53, 42]:

- **Hybrid P2P:** It is also known as hybrid decentralized networks. This network has a single central server that keeps track of the entire information in the network. The peers make use of server for searching and identifying the nodes that has the existing file. Thereby, establish a direct interaction with that node for file-sharing. In this network, peers will also increase the scalability of the network by storing the information. The drawback of this network is a central-server mechanism which leads to single point of failure.
- **Pure P2P:** It is also known as purely decentralized networks. All nodes in this network will have similar responsibilities acting as both server and client. There is no central server or router in this network. The nodes of this network are termed as *servents* (Servers + Clients). This network is also referred as serverless P2P.

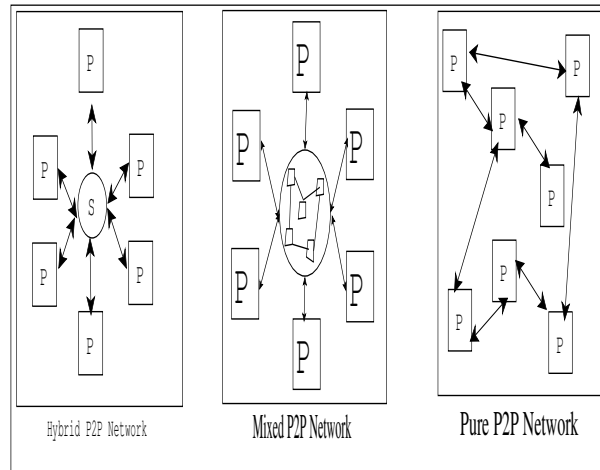


Figure 1: Classification of P2P Networks.

- **Mixed P2P:** It is also known as partially centralized networks. This network stands between hybrid and pure P2P networks. Some nodes are treated as superficial by allocating with responsibilities like maintaining central index of files shared by peers, helping a peer in establishing a relationship with another peer for existing file-sharing, etc. These superficial nodes are termed as *Super Nodes*.

Classification of P2P networks based on their structure [1, 53]:

- **Unstructured P2P Networks:** The storage of content files or the connection between nodes is done arbitrarily without following any rules or the overlay network topology. The nodes which are searching for a file will just flood the network with queries, in a way to get the file. But the chances of failure are more in this network even after we flood the network with queries because of no correlation between a peer and content managed by it. Moreover, flooding the network may increase the traffic which affects the efficiency of the network. Some examples of unstructured networks are Napster, Gnutella, and Kazaa.
- **Structured P2P Networks:** This network is constructed as a solution for the problems we come across in unstructured networks. The overlay topology is strictly followed and the content files are stored in specified locations. These networks maintain distributed routing tables to answer queries efficiently. Some examples of structured networks are Chord, Pastry, Tapestry, and CAN.

2 Advantages of P2P Networks

1. The main advantage of P2P Networks is the efficient usage of resources. Since, all nodes in the network provide resources (bandwidth, storage space, computer power), the wastage is quite low when compared with other networks where the clients resources go waste without any usage.
2. The clients in the network share its resources increasing the networks scalability.
3. The distributed mechanism of data over multiple nodes increases the robustness of the network. Also there is no single point of failure.
4. The ease in the administration of the network has also achieved this network immense popularity.
5. The joining and leaving the network is quite easier when compared with other networks.

3 History

The idea of Peer-to-Peer systems is not new. It started in 1960's with the emergence of ARPANET. ARPANET is a large wide-area network created by United States Defense Advanced Research Project Agency (ARPA). The basic goal of this network is "to share computing resources around the USA". The challenge for this effort was to integrate different kinds of existing networks as well as future technologies with one common network architecture. This would allow every host to be an equal player. But, ARPANET is not considered as pure P2P because of some client-server applications like FTP, Telnet. The disadvantage of ARPANET is its inability in informing the nodes about where the content is stored [22].

In 1979, the USENET protocol was developed that overcomes the problem we come across in ARPANET on content storage. It is a newsgroup application. However, it is a typical client-server application [12].

After 10 years, there is an increase in popularity of distributing computing with emergence of Internet. Many applications like email, WWW, and streaming were developed. But, the basic communication model was the client-server model with some distributing mechanisms.

It is 1999 that gives immense popularity to P2P networks with the development of Napster, the first P2P application. It was developed by Shawn Fanning. The main idea in developing Napster is to utilize the enormous storage space of internet clients in a significant way. This network relies on a centralized mechanism (Hybrid P2P) where, the server maintains an index of all files and its locations. When a user queries the server it answers with the indexed list of clients that has the existing file [34].

The advantage of this network lies in its efficient way of handling queries. However, as stated earlier, it has a single point of failure. Napster is treated as first-generation of P2P networks.

In later 2000, a second major P2P network, Gnutella was established as a solution for the problem we come across in Napster. This network follows a decentralized mechanism (mixed P2P) distributing responsibility of the network between some special nodes in the network. These special nodes are named as super nodes. These networks are also known as second-generation P2P networks [48].

The third generation P2P networks are emerging recently and focus on drawbacks of previous P2P networks. The main goal of these networks is to provide anonymity to its users. Some of the examples of third-generation networks are GnuNet, Ants, Mute, Waste, Freenet.

4 Attacks on P2P Networks

As already stated, P2P networks achieve immense popularity in recent years. A way to increase this popularity, it is quite important for P2P networks to be more robust and fault tolerant. But, the open nature of P2P networks leads to the vulnerability of the whole network since, it is easy for malicious nodes to join the network. These malicious nodes also called as *attackers* will implement various attacks which will affect the network's integrity and security.

In general, attacks in computer networks can be defined as [2]:

Methods aimed at destroying, altering or obstructing information in computers, computer networks or the networks themselves.

According to my findings, there are three primary modes of attacks on P2P networks:

- **Attacks on communication channel:** These attacks will try to weaken the communication between two peers in the network.
- **Attacks on Anonymity of peers:** These attacks will try to reveal the identity of peers that are sharing information in the network.

- **Attacks on individual files:** These attacks will try to affect individual files either by polluting them or by changing its contents.

In this thesis, we discuss briefly about various existing attacks on P2P networks and the possible solutions to defend against these attacks.

5 Anonymous P2P Systems

Anonymity has become an important characteristic of P2P systems. Anonymous P2P system is a system in which the identity of the users is kept hidden using various anonymous and pseudonymous techniques.

The majority of anonymous peer-to-peer systems are friend-to-friend networks. Generally, the nodes in these systems will connect to only few nodes which are neighbours to them. And, also these neighbours will know the IP address of the node. The communication with distinct nodes is done anonymously based on some routing methods. It is quite difficult to find the IP address of distinct node and the neighbours will increase the level of anonymity by claiming themselves as mediators between sender and receiver [52].

In this thesis we are going to focus on various anonymous mechanisms and also on anonymous P2P systems that are developed based on these mechanisms.

6 Technical Terms

Here, we describes some technical terms that we use in this thesis:

- **Encryption:** Encryption is the process of obscuring information to make it unreadable without any special knowledge. The main purpose of encryption is to make communication secure. In encryption process, the message to be delivered is refereed as *plain text*. This message is encrypted to obtain *cipher text*. *Cipher text*, contains all the information of message(plain text) but, it is very difficult for a hunman or computer to read the text without any proper decryption methodologies. Decryption is the process of getting back the original message (plain text) from cipher text [20].
- **Hashing:** Hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database. As, it is very easy and fast to find a data item using its hash values rather than using the original value. Hashing also used to maintain security for original data. A hash value is a unique number generated from a string of text. The other names for hash value are digital finger print, message digest. Hashes play a role in security systems where they're used to ensure that transmitted messages have not been tampered with [3].

7 Thesis Organization

This thesis is organized into various sections and the brief description of these sections is as follows:

1. Section 2, deals with various attacks/vulnerabilities we come across Peer-to-Peer networks in general.
2. Section 3, will discuss briefly about anonymity and various anonymous mechanisms.
3. Section 4, is a brief description of various existing anonymous Peer-to-Peer networks.

Chapter 2

Attacks on P2P Networks

This chapter will discuss various attacks that are implemented by attackers on P2P networks. Before the discussion proceeds, it is quite important to know about the following two techniques that are used in implementing various attacks on P2P networks:

- **IP Spoofing:** It is also termed as Internet Protocol Address Spoofing. IP Spoofing deals with the generation of IP packets with a spoofed(forged) IP address. Tanase gave a generalized and brief description of IP spoofing in his article [51]. In IP Spoofing, an attacker gains unauthorized access to a computer or a network by making it appear that a malicious message has come from a trusted machine by spoofing the IP address of that machine [1].
- **Traffic Analysis:** As the name states, the adversary will stay passive in the network and will observe and analyze the traffic created by various nodes in the network. This analysis helps the attacker to get the statistical information about various nodes in the network. This information helps him to implement various attacks that are going to be discussed in this section [44].

1 Denial-of-Service Attack

Denial-to-Service attack also known as DOS attack is a type of network designed to bring the network to its knees. As the name states, the malicious nodes involved in this attack will prohibit the services provided by the network to its intended users [1].

There exist many forms or methods to execute a DOS attack [4, 42]:

1. **Flooding:** It is the most common method that is used in executing DOS attack. The malicious nodes will flood the network with some invalid bogus packets effecting the traversal of original packets.
2. **CPU load attack:** This attack will draw victim into a fastidious computation and making it hardly available to answer other queries.
3. **Greedy user attack:** This attack tries to prevent a particular node in accessing a service or disrupt a service to a node.

Involving multiple hosts into this attack makes this attack more troublesome. Then this attack is renamed as Distributed Denial-of-Service (DDOS) attack [26]. The computers which are involved in this attack can be treated as slaves (comprised nodes) to the attacker. The attacker hides behind these slave computers and operates this attack making it very difficult to find the original source of the attack (See Figure 2).

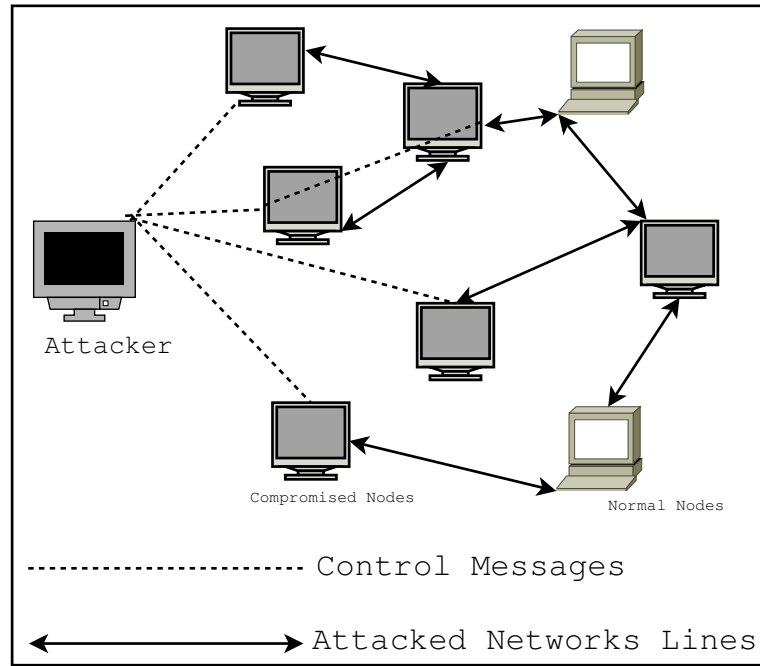


Figure 2: A DDOS Attack, attacking all communication channels.

This attack may lead to network failure which further affects the anonymity of nodes in peer-to-peer systems. Furthermore, the anonymity of peer-to-peer systems makes this affect even more effective as it is quite difficult to find and remove the node that performing DOS attack [52].

1.1 Solutions

The problem we come across in preventing DOS attack is in identifying the node which performs it. The heavy utilization of network and the anonymity of the P2P system make it quite difficult to identify the source of DOS attack. Also, the involvement of many nodes in DDOS attack makes it difficult to block it as the original attacker will hide his identity with in these so called slave nodes . So, an immediate solution to block all DOS attacks is quite difficult.

The most widely used technique to gradually slow down DOS attacks is “Pricing”. “Pricing” will limit the speed at which a node makes requests in the network. When attacker sends some request to some host, it will in turn send puzzles to the attacker making him busy with an equally expensive computation. And also, the attacker has to solve this puzzle and respond positively to get response from that corresponding host (See Figure 3). This solution seems to be effective because it makes the attacker to involve in some sort of computation limiting his time in flooding the network. But, this method will be ineffective when more number of attackers attacks a node simultaneously (DDOS attack). Another drawback is that this method may involve some honour nodes in computation with complex puzzles which will grab their valuable bandwidth and time [27, 42].

As stated earlier, anonymous P2P systems makes it difficult to find original DOS attacker. So, care should be taken in to ensure that their design will limit the affect of DOS attacks.

It is quite difficult to change the malicious behaviour of the node. The only way to get rid of malicious nodes is to identify and stop using them. In this process, the user should determine the response time of various nodes by pinging them occasionally. If a particular node drops sufficient number of packets then it can be treated as a misbehaving node and thereby, stop communicating with it [24].

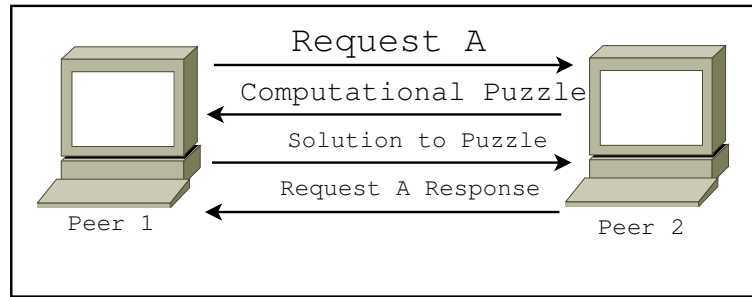


Figure 3: Diagrammatical representation of Pricing.

2 Man-in-the-Middle Attack

As the name states, malicious node itself hide between two nodes and the communication between these nodes will traverse through this node. Initially, the attacker stays passive and will spy on the communication between these nodes. It is quite difficult to identify attacker at this stage as he is in passive stage. The attacker will become active once he gets a complete knowledge of their communication. In this stage, he can modify the messages and even he can insert some fake message in to their communication [27, 42].

The basic version of this attack runs as follows and is shown in figure- 4 [19]:

- A broadcast a Query message and B responds.
- Malicious peer D intercepts the QueryHit message from B and modifies the IP and port fields to contain D's IP address and port. The modified QueryHit message is then sent back to A.
- A decides to download the file from D, which provides a fake resource.

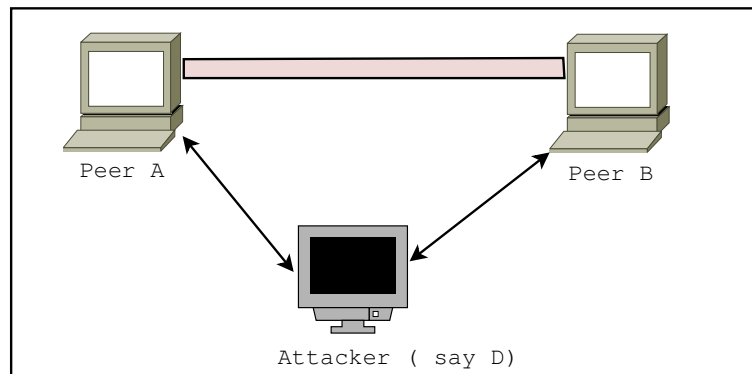


Figure 4: Man-in-the-Middle Attack.

As stated by Pretre in [42], The goals of the attack will vary depending on the protocol of the network. In general, Identity spoofing and dispatching false information are the goals of this attack.

The attacker takes the advantage of a routing mechanism in P2P Network to make him hide between the nodes.

2.1 Solutions

The best way of defending against this attack is to make it useless. But, it is quite difficult to detect this attack in P2P Networks as the Central trusted authority is lacking in these networks. Nodes will not have any information

about their neighbours and it is quite impossible to identify the attacker because of the anonymity feature of P2P network. As this attack won't affect the whole network, it is treated as useless in P2P networks.

The General method to defend the goals of this attack is by using Digital Signatures. These signatures are based on public key cryptography which are used to verify the authenticity of the messages. They can also be used to check if a message is modified. The general procedure is to attach a digital signature at the end of the message. After receiving the message, the receiving node can verify the authenticity, originality of the message with the help of these signatures. It can also check whether the message came from true source.

Encryption of the message can even further avoid an attacker from reading the contents of the messages.

3 Sybil Attack

This attack is named under a very famous character of 70's Sybil, a woman with multiple personality disorder of 16 characters [25]. As stated, the attacker (single malicious node) will get hold over a part of the network by presenting multiple identities (See Figure 5). The attacker will prefer to place his peers in the same portion of ID space . This helps the attacker to make the network more vulnerable since he will get control over a segment of network. Further, he can have a control on all the messages that passes through this segment [27].

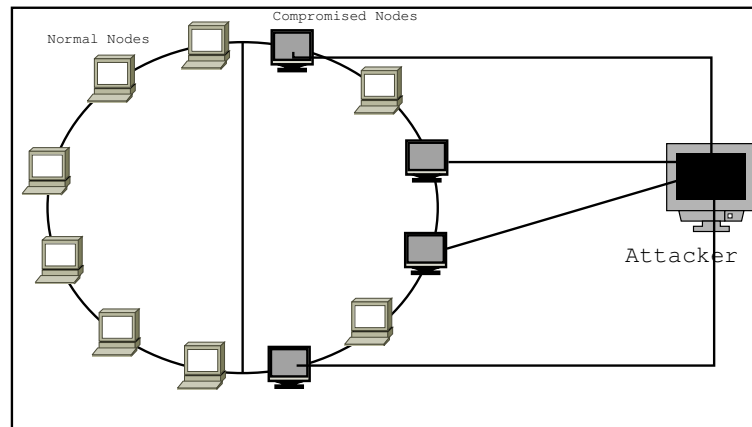


Figure 5: An example of a Sybil attack. Here attacker gains control over a part of network.

Thus, the attacker can impose large amount of damage to the network with his limited number of nodes. The goals of this attack depend on the attackers control over the network. Generally, once he gets control over the network, the goals can be like abusing the protocol in his way. He can gain responsibility over certain files and can choose to pollute them. He can even further extend this attack with eclipse attack, or slow down the network by rerouting all queries in a wrong direction. The level of damage created by an attacker depends on his strategic way of positioning his identities [27, 42].

3.1 Solutions

The problem we come across in defending Sybil Attack is the lack of central trusted authority in P2P Networks. The only way to defend this attack in an open and decentralized network is to make it ineffective.

One way to defend against Sybil attack is by not allowing attacker to place his malicious identities in strategic positions. Because, dispersed identities have less impact, even less dangerous than the strategically placed ones if the existing P2P network is of considerable size.

Another way is to include nodes IP in its identifier, limiting the attacker from generating multiple identities. As, this method can easily trace the malicious behaviour of generating fake identities by attacker, thereafter, can denounce it. But this solution cannot be considered as a perfect defence because, this will generate new problems to the networks allowing various kinds of attacks like generating fake identities for other nodes and then abusing them as malicious, which adds a layer of complexity to the existing protocol [42].

According to Engle, to slow down the Sybil attack, he suggested to follow the same method that is used to slow down DOS attack: Pricing. The solution engages attacker with certain number of puzzles which are to be solved by attacker before he attempts to join more nodes into the network. This kills attackers time thereby, limiting him to place the sufficient number of nodes into the network. Further modifications like adding nodes ID expiration to the network, will effectively limit the Sybil attack. This way the attack only has a limited amount of time to generate more nodes to execute this attack. As stated in DOS attack solution, this method has a drawback; it may involve some honour nodes into this complex computation wasting their valuable bandwidth and time [27].

4 Eclipse Attack

We can also treat this attack as a large scale Man-in-the-Middle (MitM) attack. It is a very serious threat to P2P networks because of its strategic arrangement of nodes dividing the network into two or more sub networks (See Figure 6). The communication between various nodes will traverse through the attacker's peers that are arranged strategically.

As stated in Sybil attack, this attack can also be as a continuation for Sybil attack. If an attacker starts an Eclipse attack, he can attack the network in the following ways [42]:

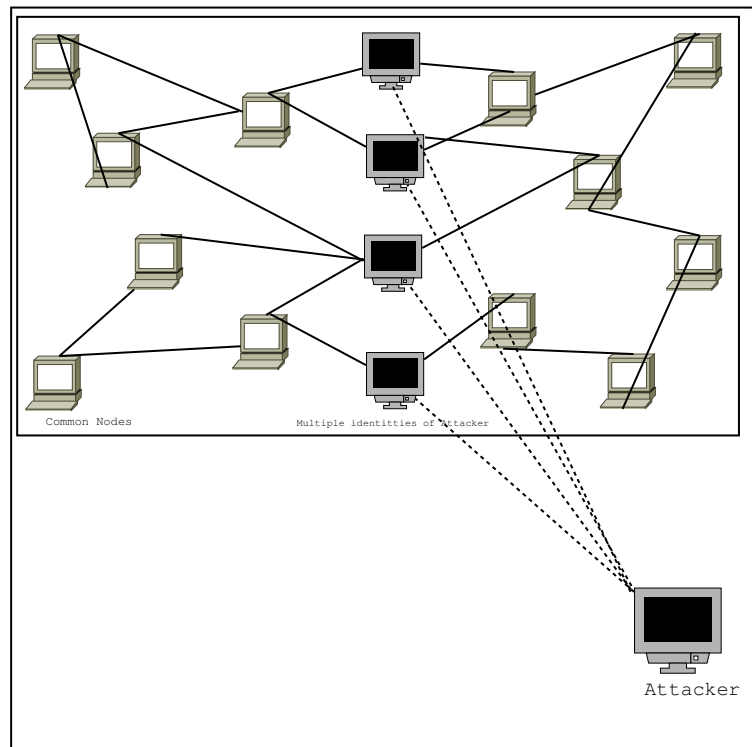


Figure 6: An Eclipse Attack, The malicious nodes separating network into two.

- Can affect control plane (affecting the functionality of P2P applications) of by inefficiently rerouting each message.
- Can decide to drop all messages he receives, thus completely separating both sub-networks.
- Faking messages, that are traversing between two nodes.
- Can affect data plane (affecting the data used by P2P applications) by injecting polluted files or requesting polluted files on behalf of innocent nodes.

Thus, the above discussed ways will help the attacker to meet his goal of slowing down the network.

4.1 Solutions

The solution provided to defend MitM like Digital Signatures and public key cryptography can also be applied here to defend this attack. But, still this attack is threatening the network because of its large scale damages [27].

Pricing can be a solution for Sybil attack version but the problem is that it will slow-down the network or even effect the scalability of the network. And, it is also important to defend Sybil attack in this case which lays a route to this attack. So, the solutions we discussed for Sybil attack will also be applicable for this attack.

Using pure P2P network can be a best solution to defend this attack. Further, including randomization algorithm that helps in the distribution of nodes over the network would be the enhancement for this solution. This makes attacker quite difficult to manage his node's position and will not allow him to place them in strategic positions. This will also help to prevent the attacker from partitioning the network which is an initial step for this attack [42].

5 Rational Attack

The success of P2P networks depends on the nodes participating in it. So, their cooperation is quite essential for the effectiveness of P2P services. However, as humans are involved in this issue it totally depends on their interests and we can't expect or even force them for cooperation. The general assumption is that many P2P nodes will stay rational, seeking to minimize their own sharing of resources while maximizing their consumption of systems resources [42].

The reasons behind this rational behaviour are [27]:

- To save their precious upload bandwidth by not sharing their resources.
- Legal issues While sharing copyrighted material, it may result in some legal actions against the owner of the node. And, also it is quite easy to track the nodes sharing the content
- Principle alone When left to their own choices, some people will not cooperate based solely on the desire to not to help the network, no matter how minimal the cost to them may be.

The two basic classes of this attack are [27]:

- Content Restriction, users are not sharing their content on the network.
- Resource Restriction, users are not contributing their resources to the network.

The main problem we come across because of this attack is that, this attack may lead to destabilization of the network. And, successful P2P systems must be designed to be robust against this class of failure.

5.1 Solutions

Napster, is the first P2P network that tries to solve this problem by giving nodes a Title for the level at which they shared the resources. Another solution by Samsara (a P2P backup system) as stated in [27] is that a node can use as much space on another node as it is giving up to the network.

Bit Torrent has a best solution against this attack. The focus of this network is to defend against resource restriction and it is not so interested on content restriction. As a solution, it implements a system for bartering for chunks of data. The more a node shares with others, the more it will get back. So, a node should upload accordingly to get more faster downloads [2].

6 Attacks as nodes live or join

This attack can be treated as a passive attack and it is quite difficult to find the attacker because of his passive behaviour. In anonymous P2P networks, peers use pseudo identities to hide their identity making it difficult for an attacker to identify various nodes in the network. So, the attacker implements this attack that helps the attacker to identify nodes in the network. The effective analysis of traffic between various nodes in the network, further helps the attacker in successful implementation of this attack.

After a node leaves the network, if the attacker finds a new query with certain identifier, then he can boldly state that this identifier doesn't belong to the node that left the network. In this way, the attacker will try to analyze the network and tries to trace the identity of various nodes in the existing network. Since, it will become easy for the attacker to trace the identities, if more nodes leave the network. Similarly, if there are n nodes in the network with n identities and if the attacker finds a query with new identity then it can easily state that the new identity belongs to new node [52].

6.1 Solutions

The best solution to defend this attack is to use dynamic identities. That is, to change a node's identity regularly and the life time of an identity should be shorter than the average time a node is a member of the system [52].

7 Intersection attack

The best way to trace a node's identity in a network is to observe its behaviour for a long time. In general, a particular node will exhibit typical online/offline periods, utilize similar resources over time and usually queries the same websites over different sessions. So, observing a set of active users over different times can give the attacker very interesting information like what users are active at a particular time and what users communicate with each other. With the observed information, the attacker can attack the node accordingly [24, 44].

7.1 Solutions

It is termed as a well known open problem to every system and it is quite difficult to prove a perfect solution for this attack [24].

8 Identity Attack

As the name states the main aim of this attack is to grab the identity of the sender, receiver or both. Attackers follow different ways in an order to grasp the identity of nodes in the network [24]. Some of them are:

- Spread a Trojan horse, worm, or virus into the network and look for the nodes that have been affected.
- By observing the network for a long time staying as an honour node thereby getting the identity of the nodes.
- Develop a customized virus which automatically contacts a given host upon execution.

Apart from the above discussed ways attackers also perform various attacks like Intersection attack, Trickle attack, Flooding attack, etc. to get the identity of the nodes [23].

8.1 Solutions

Different anonymous P2P systems follow different methods in their way to defend against Identity attack. The way these Systems provide anonymity will be discussed in the later part of this thesis. Also, defence against various attacks like flooding, intersection, trickle is quite essential and is the best way to defend this attack.

9 Spamming Attack

The other names for this attack are Flooding Attack, Flushing Attack. As the name states, the central task of this attack is to flood/spam the network/node with some unwanted garbage files. By flooding the network, the attacker aims for slow down in the network and by spamming a node he aims to get its identity [44]. This attack also seems to be an extension of the above mentioned attacks.

9.1 Solutions

The suggested solution in preventing a node from flooding is by authenticating the traversal of messages between nodes [24]. But, this solution is not considered. As, it is quite difficult to achieve anonymity by authenticating messages. It is suggested to apply solutions which we discussed in DOS attack, as this attack is implemented by DOS attackers. The solutions like Pricing can serve the purpose [44].

10 Poisoning

The other name for this attack is Pollution Attack. And it has grown as a common threat for P2P Networks. The goal of this attack is to replace a file in the network by a false one which is of no use. That file is called as polluted file.

Nash in [40], explained in more general way. A node *A* broadcast a query saying that it needs a file *X*. A malicious node can respond with Query Hit message saying that it has File *X* and send File *Y*, which is a polluted file.

Music Companies and also companies like Overpeer or Retsnap massively offer their polluted-based files for certain low amount in P2P Networks in a way to protect their copyrighted materials [42].

As stated by Pretre in [42], the way the attacker imposes polluted files into the network is by claiming falsely stating that he has a particular file and send a polluted file in that place to a particular node. All messages that pass through malicious nodes can be poisoned. They also make them more attractive to grab the users attention towards downloading that file.

10.1 Solutions

According to Pretre [42], file poisoning attacks are not serious threats for P2P networks even though they sound pretty dangerous. The reason behind this is that, once a polluted file is downloaded by a user, it will be deleted once it is proved to be a polluted file. So, after certain amount of time all the polluted files in the network will be removed and the original files will be more available in the network.

The reason for its success till today is because of three factors [42]:

1. Rational attack which states that clients are unwilling to share their resources/contents
2. Not removing the corrupted files immediately after their detection.
3. Users giving up downloading if their download seemingly stalls.

The above stated factors have been treated as serious flaws for the slow down in removing the polluted files from the network.

11 Trickle Attack

An adversary has complete active control of all the edges of the communications channel. The adversary stops all incoming messages from entering the channel except one. The next incoming message is not released until the first message is detected along some edge exiting the channel. As only one message is transmitted through the channel at once, the global observer can establish link ability between sender and receiver [24].

Alternatively, an adversary can achieve active control over all the links of some internal node with in the channel [24].

11.1 Solutions

We cannot determine any protection against this attack for communications channels which provide an explicit mapping of names to sender/receiver agents. For systems which provide partial anonymity even after exposure, the “edge” of the communications channel only reveals a k-anonymous set of possible receivers [24].

12 Worm Propagation

Before proceeding further about this attack, it is important to know the definition of worm. “A computer worm is a self-replicating computer program. It uses a network to send copies of itself to other systems and it may do so without any user intervention. Unlike a virus, it does not need to attach itself to an existing program. Worms always harm the network whereas viruses always infect or corrupt files on a targeted computer” [1].

Worms can be treated as serious threats for P2P networks because of their ability in infecting hundreds of thousands of hosts within hours. Better engineered worms can even infect in a matter of seconds.

The reasons which make P2P networks more suitable for worms are [27, 42]:

- P2P is a group of nodes running the same software making the tasks of the attacker quite simple. He can affect the entire network by finding only one exploitable security hole.
- P2P network is a network with set of nodes interconnected to each other. This gives worm a chance to spread tremendously through out the network and affect as many nodes as possible without wasting time.
- P2P networks are mend for transferring large files which makes P2P worms stronger by implementing more complicated behaviours. In general, worms have to limit their size in order to hold in a TCP packet which is not required in this case.
- Generally the nodes in P2P network are personal computers which makes an attacker to access even more sensitive information like credit cards, account passwords, etc.
- Since P2P systems are used to transfer illegal content (copyrighted music, pornography, etc.). The chances of reporting on unusual behaviour are quite negligible.
- The final and probably most juicy quality of networks possess is their potentially immense size.

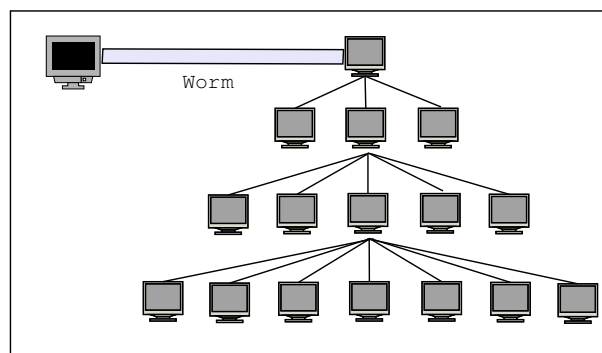


Figure 7: How worm spreads in a Network.

Once the worm gets hold of the network, it can use the entire network as a tool to execute other attacks on other hosts or networks. For example, the compromised networks can be used to apply DDOS attack on other networks [27].

12.1 Solutions

The main defense against worms is to keep the applications/nodes itself secure. Leaving a computer without a complete firewall and anti-virus on a broadband internet makes the worm to spread effectively in the network. So, a strongly recommended solution is to write P2P clients in strongly typed languages, which can avoid many security flaws [27, 42].

Another possible solution is to avoid using hybrid networks. As, hybrid nodes make some nodes as super nodes which are superior than other nodes. This makes the work of worms easier since they can directly target on these nodes to affect the whole network. So, it is recommended to use pure networks in which all nodes have same responsibility [42].

Operating System developers also provide some solutions by recommending hardened operating system like OpenBSDs 3.8 release which returns pseudo-random memory addresses. This again makes it more difficult to execute many attacks successfully [27, 42].

The most practical defense to worms on P2P networks (that are implemented on the Internet) is to use the open nature of the network itself. That is, to develop open standards. Freely releasing the protocol and even code to implement network clients encourages developers to make their own client for that specific P2P network.

These new clients will diversify the network, so not everyone will be vulnerable to the same exact flaw found in one client [27].

Chapter 3

Anonymity and Anonymous Mechanisms

Privacy deals with the control on dispersion of one's personal information. This can be achieved with the help of anonymity. Encrypting the communication between two persons will only hide the contents of their transaction. But, the attackers can get various details like their IP addresses, duration of their communication, etc that reveals their identity. And there is a necessity to hide this information to enhance the privacy of users in a network. Anonymity is the best solution.

1 Anonymity

What is Anonymity?

Anonymity is the state of being not identifiable with in a set of subjects, the anonymity set [41].

In general words, the idea behind anonymity is to hide both the communication between two users and also their identities in a system. There are different notions of anonymity explained by various authors.

In communication perspective, there exist three types of anonymity [41]:

1. **Sender Anonymity:** This deals with hiding the identity of user who initiates the communication.
2. **Receiver Anonymity:** This deals with hiding the identity of user who responds to sender's queries and send files accordingly.
3. **Mutual Anonymity:** This deals with hiding identities of both sender and receiver from each other and also from other nodes in the network.

Generally, anonymity in the above discussion is to hide one's identity from every node in the network. For example, sender anonymity means to hide sender's identity from every node in the network, including nodes that traverse the communication, responder node and from various attacking nodes in the network and vice versa.

Dingledine describes the following aspects of anonymity based on different types of users and their actions in a system [24]:

1. **Author-Anonymity:** This deals with hiding the identity of original author of a particular document.
2. **Publisher-Anonymity:** This deals with hiding the identity of the agent who introduces a particular document in to the system. Sometimes, author and publisher may resemble single agent.

3. **Reader-Anonymity:** This deals with hiding the identity of an agent who requests for a particular document from other nodes in the system.
4. **Server-Anonymity:** This deals with hiding the location of a particular document. In brief, this deals with hiding the identity of server that contains the document.
5. **Document-Anonymity:** This deals with hiding the contents of the document from server in which it is stored.
6. **Query-Anonymity:** This deals with hiding the identity of the document itself from server.

Besides the above stated notions of anonymity, Reiter and Rubin introduces a very special notion, *degree of anonymity*, which gives the information about the level of anonymity provided by various anonymous mechanisms [45].

The degrees of anonymity provided by various anonymous mechanisms ranges from *absolute privacy* to *provably exposed* [45, 52]

1. **Absolute privacy:** This gives an absolute privacy to the agent and the attacker in no way distinguish the identity of the agent.
2. **Beyond suspicion:** From the attacker's point of view, the detected user appears no more likely to have originated the action than any other node.
3. **Probable innocence:** From the attacker's point of view, the detected user appears no more likely to have originated the action than not to have.
4. **Possible innocence:** From the attacker's point of view, there is a nontrivial probability that the detected user did not originate the action.
5. **Exposed:** The attacker can identify the sender of the message.
6. **Provably exposed:** Here, the attacker not only identifying the sender but also, can prove the identity of sender to others.

The degree of anonymity can be chosen based on the user and his circumstances. The preferable degree of anonymity is probable innocence, which prevents various attackers from acting on their suspicions because of the probability that shows the incorrect of these suspicions. But, if the user demands for higher security, than he can opt for beyond suspicion.

2 Various existing Anonymous Mechanisms:

This section will briefly describe various existing anonymous mechanisms:

2.1 Mixes

Chaum introduced the concept of Mix-net for anonymous communication, which is achieved by using mixes. A mix is an enhanced proxy that helps in hiding the sender from receiver, and also provide sender and receiver unlinkability against a global eavesdropper [45].

This system will forward messages through chain of nodes, using layered encryption such that, each node can only know its predecessor and successor in the chain. The basic principle followed by this method, while

forwarding messages is that, every node in the chain waits until it receives a group of messages and then it forwards the mix-up of messages [16].

In brief, each mix will collect group of encrypted messages from the senders, decrypt them, batched, re-ordered, remove the sender's name and identifying information, and forward to next node. This makes difficult for an eavesdropper to correlate the output messages with the corresponding input messages[23]. For example, each processor P that wants to communicate with processor Q with message m , will encrypt m using Q 's public key to obtain m' . Then P will encrypt the pair (m', q) using the public key of the mix. The mix will then decrypts the message and forward it to its destination Q . The extension of this process is to use several mixes instead of a single mix to achieve better anonymity as already stated [13].

The main drawbacks of this system are: each node in the chain has to wait until it gets sufficient number of messages for a proper mix-up, which might be difficult in file-sharing systems[52].

2.2 Crowds

Crowds is an anonymous network developed by Michael K. Reitter and Aviel D. Rubin. The degree of anonymity given by crowds can be ranged as probable innocence which defends against large number of attackers. This mechanism will defend against internal attackers and a corrupt receiver, but provides no anonymity against a global attacker or a local eavesdropper [1].

Crowds, was named for the notion "*blending into a crowd*", which operates by grouping users into a large and geographically diverse group "*crowd*", in which every user's intention is to hide their identity while communicating with some web server. The anonymity is achieved by hiding one's actions with the actions of many others while traversing one's message. In this way, crowds make it difficult for web servers to trace the original initiator of the message because, it is equally likely to have originated from any member of the crowd. Even the members in the crowd cannot distinguish the initiator of a message from those who are forwarding the requests .

As already stated, Crowds works by making each node seems equally likely to be the initiator of the message. Initially, a user who wishes to join the "*crowd*" will start a small process in its computer called *jondo*. This *jondo* will forward and receive requests from other users in the crowd. All nodes in the network will be informed about the entry of new node so that they will select him as a forwarder. The Architecture of Crowds is shown in Figure- 8

To explain briefly, a node (Say User-A) who wants to initiate a message will first send the message to its *jondo* (*jondo-1*). The *jondo* will select a node (Say *jondo 4*) randomly from the pool of nodes in the network and forwards the message. The selected node upon receiving the request will flip a biased coin, to decide whether or not to forward this request to another node. The coin decides about forwarding the request based on probability p_f ($p_f > 1/2$). If the probability is to forward, then it will forward to another node and the process continues. If the probability is not to forward, then it will directly forward the request to the final destination. Each node when forwarding to another node records the predecessor's information and in this way a path is built, which is used for communication between the sender and the receiver [24].

Basic algorithms followed by each node [1]:

OnReceive (Node P, Message M)

1. Flip biased coin($\text{Pr}(\text{Heads}) = p_f$).
 - (a) if Heads Then Select a uniformly random node and forward to them.
 - (b) else Forward to final destination.
2. Record P, so that a tunnel can build.

As already stated, the main feature of Crowds lies on the way it makes the request looks same at every hop in the path. So, it is difficult for a hop to identify the initiator of a message [29].

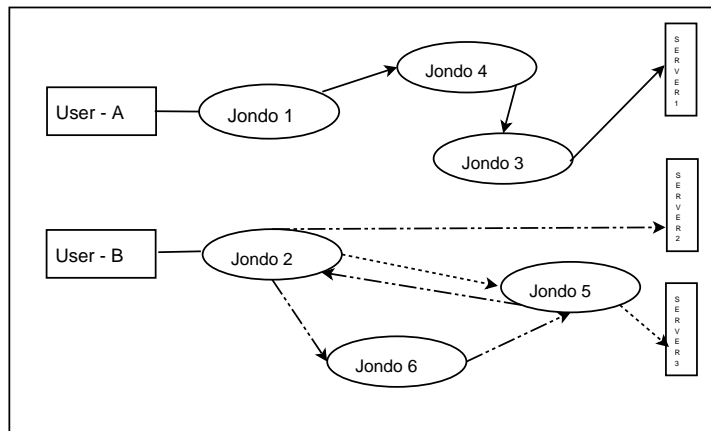


Figure 8: Crowd Architecture.

There is a possibility for a single node to be a part of multiple paths. So, it is important for a node to behave differently in each path by maintaining different path-ids for different paths. If a node fails in maintaining different path-ids then it will behave identically in every path, which results to an infinite loop [45]. For example, *jondo 2* in Figure 8 is forwarding two different messages. So, it is necessary for *jondo 2* to behave differently while forwarding these messages so that, they will reach the destination successfully.

Crowds will try to provide sender's anonymity from all the nodes in the network along with the receiver. On the other hand, it provides receiver anonymity only from adversaries and not from the sender. The other drawbacks of Crowds include, they failed to defend against denial-of-service attacks by rogue crowd members. Also, the anonymity of sender is exposed to local eavesdropper.

2.3 Onion Routing

Onion routing is a technique developed by David Goldschlag, Michael Reed and Paul Syverson for anonymous communication over a network. This technique is developed based on David Chaum's mix networks [1].

The main goal of this protocol is to protect the identity of an initiator and responder of a message and also the content of the message while it is traversing over the network. *Routing onions* is the core concept used by this protocol for anonymous communication. This concept deals with encoding routing information in a set of encrypted layers.

The anonymous communication between two nodes in this protocol happens in the following way [39]:

When an initiator wants to establish anonymous communication with a responder, he will approach *application proxy*. This will in turn forward this message to *Onion proxy*. Onion proxy will randomly select some routers and establish a route constructing an *Onion*. This *Onion*, is a recursively layered data structure that contains the information about the route to be followed over a network. These routers in *onion* are termed as *core onion routers*

The next step in this process is to forward this onion to entry funnel which is an entrance for routing in the network. Entry funnel will decrypt the first layer to see the information about next hop in the route. This process continues until the onion reaches exit funnel. Exit funnel is responsible for traversing the packet to its destination.

Onion Routing relies on *Public Key Cryptography* for anonymous Communication assuming that onion proxy will know public keys of all the routers in the network [39]. Here, a router can only decrypt the corresponding layer with its private key.

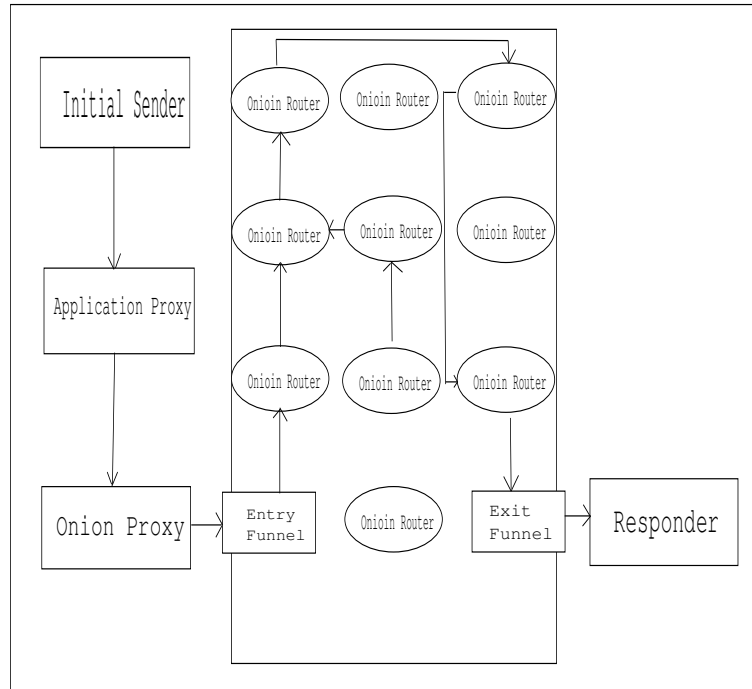


Figure 9: Architecture of Onion Routing [39].

Let us take an example for the better understanding of this protocol. Suppose, if Alice wishes to send information m to Bob, she will select a random routing path from the existing routers. Let the selected routing path be through $R_1, R_2, R_3, \dots, R_n$ where P_i be the public key for R_i . Another assumption is that all routers in the path use same encryption algorithm A . Now, Alice sends $m' = A_{P1} (A_{P2} ((A_{Pn} (m, Bob), R_n), R_3), R_2)$ to R_1 . Here, m' is an onion with multiple layers of encryption. Each router in the path can only decrypt the corresponding layer to know the next hop in the route and also the small onion that is to be traversed. In brief, R_1 can only decrypt the upper most layers to know the information about next hop R_2 [33].

The response send by the recipient is handled in two ways. As stated by Morain.et.al , when a responder responds with his response, the exit funnel will encrypts it with its own private key and forwards it to the hop from which it got the onion. Each hop in the route will in turn encrypt the response with their private key and send it back following the route. Finally, the encrypted onion will reach onion proxy which will decrypt the layers with public keys of routers to get the original message which is then forwarded to sender [39]. Another way is by reply onions, which contains a path to be followed by responder to send his response back to sender. Initially, sender will generate both onion and reverse onion to complete a two-way communication. The generated reverse onion is send to responder following the basic norms of onion routing along with the original onion [1].

The main advantage of onion routing lies in its multiple layers of encryption. Such that, even the routers in the network cannot see the content of the message. The Routers can only know the information about the previous hop and next hop in the route. If an attacker compromises a node in the route, it can only get very little information keeping the anonymity of original source and destination. But, it is stated that onion routing will provide anonymity only from third parties and the nodes involved in the communication knows each other [33].

2.4 Antnet Routing

P2P networks like Ants and MUTE, based on this routing algorithm, in a way to provide anonymity while sharing files among various peers in the network. The basic idea of this algorithm is inspired by the techniques of real ants in searching for food [31].

To explain briefly, when ants start to search for food they walk randomly towards the food leaving a scent called pheromone, throughout their route. This pheromone helps them to travel back to home once they find the food. Ants will continuously deposit pheromone on its way, while travelling between food and home. As the process continues, the path of the ant which finds shortest route between home and food smells with more pheromone. So, all other ants too choose this shortest path as their way upon getting the smell of pheromone. More information about ants and ant colony optimization is available in [5].

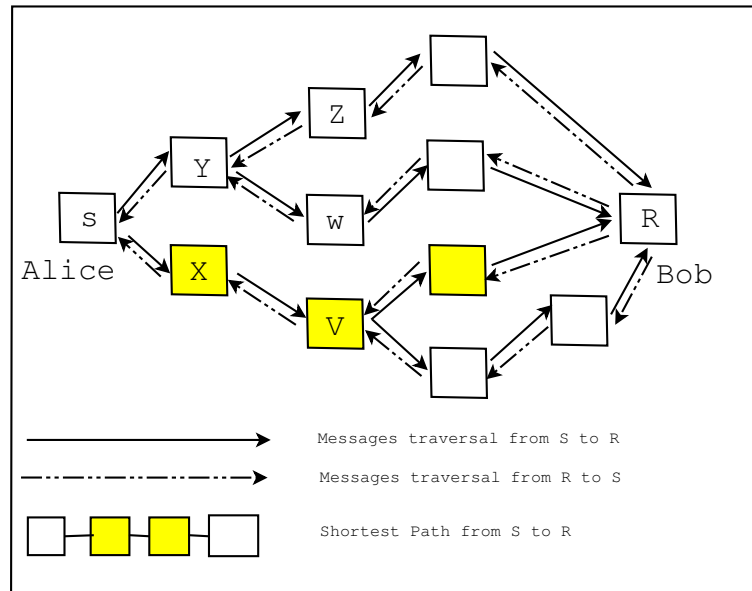


Figure 10: Antnet Routing.

Taking ants as inspiration, Antnet routing algorithm is developed with three phases [6, 31]:

- **Route Discovery:** As the name states, this phase helps in discovering various available routes between sender and receiver of a message. In a way to find a route to reach receiver, sender makes a random walk or a broadcast search that leads it towards the receiver.

Suppose, if sender S wants to send message to receiver R , it will first forwards its request to its neighbours (say X , Y). They will further forwards the request to their neighbours until they find the receiver R . During their traversal each node will keep track of S (stores some clues of S) to traverse back the response of R to S . Similarly, they even keep track of R to route further messages between S and R .

- **Route Maintenance:** This phase helps in the improvement of routes between a sender and receiver. This algorithm doesn't need any special packets for route maintenance. It depends on the pheromone deposited by every message in their traversal to find the shortest path and for better maintenance of route. The pheromone here is the clue left by a message at every node in its traversal. This clue is nothing but a pseudo identity of sender/receiver. Based on these clues, a shortest path is selected. For example, if X is traversing most of the messages that S sends then X is selected as a path to route messages to S after sensing the strong smell of clues at node X and the process continues.
- **Route Failure Handling:** This phase helps in handling the failed routes which is quite common in ad-hoc networks. The reason for route failures lies in the mobility of nodes in the network. This algorithm recognizes a route failure through a missing acknowledgement. If a node finds an error message, then it will first deactivate the link by setting the pheromone value to 0. The node then searches for an alternative link in its routing table. If it finds any alternative link, then it will traverse the message via, that link. Otherwise, a new route is created getting back to the initial phase.

To maintain anonymity, nodes in the network make use of pseudo identities while routing their messages. These pseudo identities help the nodes to hide their original identity from other nodes in the network. Moreover, it is very difficult for other nodes in the network to correlate the pseudo identity of a particular node with its original identity. The clues left at every node by a message of a particular node while routing may raise certain doubts about anonymity. But, these clues are just hints that help in routing messages and in maintaining paths. For example, let us assume that Alice (pseudo identity of S) wants to message to Bob (pseudo identity of R). The message traverses through various nodes in the network while reaching its destination. None of these nodes can correlate Alice to S and Bob to R . Here, V that traverses the message will only know that it receives a message from Alice through X but it doesn't know which node is actually Alice. In brief, none of the nodes along the path between Alice and Bob know enough to conclude that "*my neighbour is Alice*".

This algorithm serves effectively for anonymous searching and traversal of a file. But, this algorithm doesn't serve well, when two particular nodes want to communicate in the network. Since, this make lead to a random walk or a broadcast search between those nodes. It is also stated that, this algorithm may not sound good for large networks.

Chapter 4

Various Anonymous P2P Networks

This chapter will briefly describe various existing Anonymous P2P Networks:

1 MUTE

The MUTE Network or MUTE-net is a peer-to-peer and friend-to-friend file sharing network developed aiming at anonymity of peers in its network [1]. MUTE was developed by *Jason Rohrer* using a routing algorithm based on *ant colony optimization*. The developer got inspired by ants and their way in search of food source. He applied these techniques in developing a network which, provides easy search-and-download functionality while also preserving your privacy. Routing downloads makes the difference between MUTE and other peer-to-peer systems [54].

Generally, in every P2P systems nodes connect to each other forming a “*mesh network*”. In this type of networks, every node connecting directly to its neighbouring nodes and these nodes connect to their neighbouring nodes and so on forming a mesh. To preserve anonymity, search requests and responses are routed through this mesh in P2P networks. And, MUTE increase the anonymity level by extending this routing mechanism to file-transfers.

In a way to protect privacy, every node in MUTE network generates a random pseudo-id (virtual address) every time it starts up a communication and uses this as an address while traversing messages in the network. No node in the network can depict the pseudo identity of a particular node. And, it is also very difficult to connect this pseudo-id with the original-id of a particular node [6]. It is stated that only the neighbouring nodes of a particular node can know its identity, establishing friend-to-friend network.

As already stated, the messages make use of pseudo identities while traversing messages between nodes in the network and the routing mechanism depends on ant-inspired technique. We can get more details about this routing in [6, 31]. Further; *military-grade encryption* is used to hide the contents of the message. This technique helps to get rid of a spy who is monitoring the traffic of a particular node. The nodes in the network through which our message traverses can only decrypt the message [6].

The other features of MUTE include its three-phase “Utility counter” to control how far a search message travels and also to avoid flooding in the network [18]. Mute uses probabilistic algorithm for back route selection which, also helps in establishing a shortest path between sender and receiver.

Apart from the vast advantages provided by MUTE, we do face some problems. Since, the downloading is routing through many nodes, the download rate is quite low. It is not suitable for large networks. And, if a node wants to contact a particular node it will end up doing a broadcast search or random walk [50]. Moreover, all the nodes in the network can read the content of the message which may lead to certain vulnerabilities.

2 Anonymizing Peer-to-Peer Proxy

Anonymizing Peer-to-Peer Proxy in short known as AP3, is a decentralized anonymous P2P Network that provides cooperative and distributed anonymous communication services to its users [37]. This system is developed with high scalability and minimal state effectively defending various attacks. Moreover, AP3 suits well to large and dynamic networks with acceptable performance. AP3 uses the same technique that we discussed in Crowds with some differences. AP3 maintains less state by using dynamic routing paths and also it eliminates the using of central points like *blenders in crowds*.

The main goal of AP3 System is to provide strong anonymity together with usable performance by maintaining a lightweight, low-cost protocol. The level of anonymity provided by this system to its users can be ranked as *probable innocence* [17]. And, it is stated that AP3 provides strong anonymity to its peers defending against large scale co-ordinated attacks which was implemented by limited fraction of malicious nodes.

The anonymous communication between various nodes is possible by using three simple primitives provided by this system [35, 37]:

1. **Anonymous message delivery:** The anonymous routing of messages between peers in AP3 is similar to the strategies that are implemented by crowds. The originator of a message is hidden under a group of peers while forwarding its requests. Moreover, a peer that forwards the request will only know information about its predecessor appearing it as an originator of the message.

When a node decides to send a message, it will first generate an anonymous request containing a message and the address of the recipient. This request is forwarded to a node in an overlay network which was chosen by drawing a random key. The underlying routing substrate ensures efficient delivery to the node responsible for this key. Upon receiving a message, a peer will perform a weighted coin toss to determine the next peer in the path to which the intended message is to be delivered. The chosen next peer can be a randomly selected peer in the network or the destination. The decision to forward is made with probability p_f , the *forward probability*. It is also important to obscure the originator's identity from both the recipient of a message and also from the other nodes in the path to protect sender's identity. For probable innocence of anonymity, p_f should be at least 0.5, otherwise the chances for revealing the anonymity of the sender is more.

2. **Anonymous Channels:** Anonymous routing will route the messages of originator preserving its identity. But, to receive a response for the request it is important to reveal some information that helps the responder to send his response to originator. AP3 builds anonymous channels that allow responder to send this response while providing anonymity to an originator of a message.

A node, who wishes to setup an anonymous channel will choose a random id (the address of the channel) and send its message to a node closest to this id. Each node in the path that traverses this message to its destination node will remember the node from which it gets the message, by maintaining a forwarding table. The nodes in the random path will reconstruct the anonymous path back to the sender by using this forwarding table. It is important to provide an expiry for entries in forwarding table to maintain the efficiency in message delivery. The nodes in the network may go down breaking the channel. In such case, a new and fresh channel is generated by the node.

3. **Secure Anonymous Pseudonyms:** AP3 provides users with secure, persistent online identities that are different from original identities. These identities help user in encryption and authentication of their messages. Each node will create a public/private key pairs (K_{pub} , K_{priv}) corresponding to one pseudonym. And, it is allowed to generate multiple pseudonyms as per the requirement.

A node will create an anonymous channel at a location $H(K_{pub})$ where, H is a secure hash function that helps the recipient in sending the responses. The node who wants to send message securely to the owner of a pseudonym will encrypt the message with pseudonym's public key. This will prevent others from reading the messages.

AP3 also uses a timeout mechanism that helps a node to resubmit its request into the network. This mechanism is introduced to overcome various problems that cause the failure in delivering the request to the recipient. One such reason for failure is that weighted coin always forwards the message to a randomly chosen node but not to its destination. The sender will start a timer, once he send a request and wait until the timer goes out, it will again resubmit the request in a newly selected random path, with a new unique request id. The other main feature of AP3 is that it requires very less processing when a node joins or leaves the network [17]. AP3 also helps to build novel anonymous group communication facilities, preserving the identity of both publisher and subscriber [37].

3 Free Haven

Free Haven is an anonymous system developed aiming at distributed, persistent, anonymous data storage that resists the attempts of powerful adversaries to find and destroy any stored data. This project was started in 1999. Free Haven makes use of secure mix net for communication. The main research goals of Free Haven include [7, 22]:

- **Anonymity:** As many Anonymous P2P systems, this system also tries to give anonymity to all its participants: the publisher who publishes a documents in this system, the readers who retrieve these documents, and the servers that store these documents.
- **Persistence:** The main goal of this system is to achieve persistence of data. And, it is the publisher in this case who decides the persistence (life time) of data (document) but not the server that stores the document.
- **Flexibility:** The functionality of the system is quite smooth, making it easy for its peers to join/leave this system.
- **Accountability:** The main theme of this goal is to overcome the damage created by misbehaving nodes in the system. Reputation system serves this purpose that credits the behaviour of nodes in the system. Accountability is achieved in this system without any compromise on anonymity.

This system will achieve the above desired goals by its design. The design consists of the publication system and communication channel. The Publication system is responsible for storing and serving documents. And, the communication channel is responsible for anonymous communication between participants in the network [21].

Free Haven is based on a community of servers called the *servnet*. Each server in this community will host data from other servers in a way to store its own data in this network. The data (documents) stored are divided into parts known as *shares*. Publisher will store a document in servnet providing an expiration date to it. And, the servers in servnet should store the *shares* of a document until its expiry. In addition, each servnet node has a pseudonym that helps to hide its identity. Within the servnet, Free Haven is a peer-to-peer system.

When a publisher wants to publish a document (file) should first find a server, which is willing to store a document. As a first step, the server makes use of Rabin's information dispersal algorithm (IDA) [43] to break a file into shares f_1, \dots, f_n where, k shares out of n are sufficient to recreate document. Each share is assigned to a server in servnet to publish them in network. The server will generate a key pair (PK_{doc}, SK_{doc}) for its share and builds a data segment. Each share will consists of certain attributes like expiration information, share number, time stamp and the signature itself.

The reader who wishes to access the document will send a request to any server in servnet using its remailer address with a location and a key (PK) which can be used to deliver the document in private manner. The server will broadcast the request to all other servers in servnet. The servers which are holding the shares of that document with public key PK will encrypt them and deliver them to the reader's location.

Behind these scenes, the system also uses trading, reputation system for the effective maintenance of the system. Trading allows the servnet to be dynamic which helps the servnet nodes to join and leave the network very easily without and special treatment. It also helps to transfer data between trusted nodes quite easily. As already stated, reputation system helps to achieve accountability. To maintain accountability, every server in the network will maintain a database that comprise of information about other servers behaviour in the servnet. This infomrstion is gathered from its experience and also bassed on what other server's told. We can get more information on trading and reputation system in [21, 24].

For anonymous communication between servers and also between servers and readers, this system relies on existing *mix net* infrastructure that provides an anonymous channel for communication. More information on communication channel is available on [38].

Free Haven is still under construction with lot of problems being unsolved and with some disadvantages. These problems include inefficiency in network deployment, lack of proper anonymous communication channel (still under construction), tricky reputation system. The discussed problems lack this system from its better evaluation.

4 Freenet

Freenet is a decentralized peer-to-peer system for distributed data storage that aims to provide electronic freedom of speech through strong anonymity. This system was designed by Ian Clarke and can be treated as both an anonymous mechanism and also as anonymous peer-to-peer system [1, 52].

The main features of Freenet are to provide anonymous methods for storing and retrieving information [36]. Every node in the Freenet act as both server and client and the system doesn't have any central control. The Freenet routing protocol uses key-based routing protocols for routing packets over the network. The contents of the file are encrypted and distributed around various nodes in the system so, it is very difficult to find out a node that hosting a particular file. The portions of file are spread out in small shards among different hosts in the network .

When the user request for a particular file, he initially hashes the name of the document and forwards the request to his server to get the location. If the server can't find the information it will forward the message to nearby nodes that has the file with most similar hash value. Freenet clusters files with similar hashes nearby each other, and uses key-based routing to route the queries in tracing the desired file. Once the file is found, the request will travel back in the same path. All nodes in the path will cache that file so that the search for the file in future can be done more efficiently and quickly. The lifetime of these cached files by various nodes in the path depends on their popularity [24].

Freenet will provide both sender's and receiver's anonymity but there are some vulnerabilities. Let us consider an example, where Bob receives a request from Alice, it is difficult for him to tell whether the request is hosted from Alice or it is a forwarded message from her. As already stated, the requests will send back along the same path. The sender(originater of the message) will stop forwarding the request once he receives it. Here, an attacker who is monitoring the

5 Ants

Ants' protocol was launched by Gwren in 2004 for ad-hoc networks, in which the position of nodes is not fixed[1]. In this system, every node uses its pseudo identity while communicating with other nodes hiding its original identity . To maintain anonymity, the communication between sender and receiver of a message is cascaded through several nodes in the system so that both sender and receiver can sustain their identity.

When a node wants to send a request, it will generate a query with its pseudo identity, a unique message

identifier and forwards it to its neighbouring nodes with a time-to-live counter. And the neighbouring nodes will forward further to their neighbouring nodes until the time-to-live counter runs out. When a particular node receives a request it will record the path of the message and the pseudo identity of the sender. Thus, every node will maintain a routing table for all pseudo identities it sees. This table helps the node in sending request back to the sender using his pseudo identity as *To* address. In this process, if a node finds a pseudo identity in its table, it will forward the message along the most used connection. Or else, it will forward to all its neighbours [52].

It is stated that the degree of anonymity provided by ants' protocol can be rated as *probable innocence* if a proper probabilistic time-to-live counter is maintained.

6 Anonymous Peer-to-peer File Sharing

Anonymous Peer-to-peer File Sharing in short known as APFS, is a protocol developed to provide mutual anonymity for its users in file sharing. APFS also addresses a problem in providing anonymity for long-live internet servers in the face of anonymous degradation. It depends on onion routing for providing anonymity. It is stated that APFS efficiently uses features of peer-to-peer environment in providing solutions to problems in specific to responder's anonymity.

There exist two types of solutions followed by APFS in establishing mutual anonymity [47]:

- **Unicast transmission:** Here APFS relies on a central proxy called *coordinator*, which helps as a boot strapping point of any peer in the network. The peers will take the help of this proxy for initializing, and for starting a communication.
- **Multicast routing:** The centralized behaviour of proxy, which leads to single point of failure is eliminated in this model. Even though the complexity of the model increases, it gives more advantages when compared with prior mode.

The two stages of APFS protocol are [47]:

1. **Initialization:** In this stage, clients join into the network for anonymous file-sharing. All messages in this stage are overt, as the group membership is required to be known. In unicast model, a client who wants to join the network will approach the central proxy, coordinator. And, the client can get the entire information about the network from this coordinator. In multicast model, the central proxy is replaced by multicast communication between clients. A client who wants to starts communication will subscribe to multicast group and send messages periodically to the multicast address. Clients get the information about the network by hearing to sufficient number of other participants.
2. **Peer-to-Peer Services:** Peers will send anonymous requests to the existing servers and the servers in turn respond with anonymous replies. All messages in this stage are anonymous. The peers make use of *onion routing* for anonymous communication. In unicast, the clients who want to act as server will contact coordinator to establish themselves as servers. And the nodes who want to query the servers will initially contact coordinator to get the list of servers. In multicast, the coordinator is replaced by multicast group.

As this system mainly focuses on responder's anonymity, there are chances that leads to the leakage of initiator's identity.

7 WASTE

WASTE is a peer-to-peer and friend-to friend protocol developed by Justin Frankel in 2003 . It is an anonymous, secure, and encrypted collaboration tool with features like instant messaging, chat rooms and file browsing/sharing capabilities. This tool was named referring to Thomas Pynchon's novel "*The crying of Lot 49*", in which it is an acronym for "We Await Silent Tristero's Empire" (W.A.S.T.E) [1].

WASTE was developed aiming at small group of trusted users. It is stated that WASTE can support to build a network with 10-50 nodes. It guarantees anonymity to all its nodes in the network on condition that no one allows an attacker to join the network. Heavy encryption is used while traversing messages between nodes to ensure security from third parties. These types of networks generally referred as *darknet*.

WASTE uses a decentralized architecture allowing nodes to form a mesh network. As the nodes in the network trust each other, every node will know IP addresses of every other node in the network but to sustain one's anonymity, every node will create a proxy WASTE node to hide itself behind that proxy. We can learn more about anonymous communication in WASTE using proxies from [55].

There exist three types of messages: Broadcast messages, Routed reply messages, and local management messages. When a node wants to request for certain information, it will send broadcast message to all nodes in the network. The response of this message is routed reply message. Link management message is send directly between two nodes to negotiate link configurations, etc. Nodes in the network can broadcast and route traffic selecting low latency path (route) which, leads to an effective load balancing. This also improves privacy, because packets often take different routes [8].

In WASTE network, messages between trusted nodes are encrypted and the idle nodes generate dummy traffic making it difficult for attackers in analyzing traffic [52]. But, the drawback of this network is that every node through which a message traverses can spoof the message. This may also make an eavesdropper to spoof the message, if he is a member of the group. So, the behaviour of every node in this network is highly valued.

8 GNUnet

GNUnet is a searchable, decentralized, peer-to-peer network. The main goal of this network is anonymous censorship-resistant file-sharing. The framework of this network provide link encryption, peer discovery and resource allocation. The implementation of this network was first started in 2001 with new set of technical ideas. The ideas include a technique for efficient content encoding (ECSR - the encoding for censorship resistant sharing) and a new protocol for anonymous routing, GAP (GNUnet's Anonymity Protocol) that based on mixes [1].

GNUnet make use of content encoding that divides a file into small blocks of 1kb and are transmitted using GAP over the network.

Content Encoding

As stated, GNUnet transfers files in form of blocks of fixed size (1 kb). There exist three types of blocks: Data Blocks (DBlocks), Indirection Blocks (IBlocks) and Root Blocks (RBlocks).

A file is divided into small pieces of chunks known as DBlocks, which are hashed individually. These hash codes are grouped into indirection nodes (IBlocks). On top of these lies RBlock, contains the description of file and the query-hash to retrieve the root of the tree of IBlocks. We can get more information on Content encoding from [9, 15].

Splitting the file into small chunks makes content distribution much easier. Further, the query for RBlock is send encrypted so that the intermediaries cannot assess the information for which the user is looking for.

Routing

There exist two types of messages in GAP protocol, queries and replies. Further, there exists two types of queries search queries, download queries.

A node requests for RBlock of a particular file through search query. Once it gets the RBlock, it can download the entire file by sending several download queries. A query also specifies the address to where we should send replies. The steps followed by a particular node after receiving a query are as follows [14, 30]:

1. The node will check its resources like bandwidth, storage space, CPU to determine whether it can process the query or not. It will drop the query and exit if it is too busy. Or else proceed further in processing the query.
2. It will check for the availability of the requested content locally. If it found the content, a response is enquired in the message queue corresponding to the return address given in the query. Otherwise, it will decide to forward it to other nodes.
3. Decide to how many n nodes it should send query. If $n > 0$, enqueue for sending to n other nodes

The various nodes in this network achieve anonymity by acting as intermediaries of messages. It is also possible for a node to trade anonymity for efficiency by stopping the mixed nodes from writing the reply address of each message. GAP also credits the behaviour of nodes to distinguish malicious nodes in the network. Detailed information on GAP is available on [30].

9 Mantis

Mantis is a searchable peer-to-peer network of anonymous nodes developed in a way to protect the privacy of individuals sharing information in the network. The main goal of this network is to provide anonymity to nodes that are acting as servers. They do provide anonymity to its clients who are requesting for information but to increase the efficiency of data transfer, client's identity is revealed to servers [46].

Mantis is modelled after crowds in traversing a search query along the network with a difference in network structure. The nodes in this network are arranged in a tree-like structure, similar to Gnutella. When a particular node sends a reply to a search query, it traverses back to the client along the path traversed by the original search query. Thereby, establish a tunnel between client and server through which future transactions between them are traversed. This way also aims in minimizing the traffic relayed through P2P network.

Mantis makes use of Blenders as in crowds which is a directory server that allows jondos (peers) to find other peers in the network. Every node makes use of several blenders to avoid certain vulnerabilities like Sybil attack, to get rid of evil blenders etc. A blender requires node's IP address and a listening port for new connection for registration and revealing any other additional information to blenders is discouraged. There after registration, nodes request for connection list and connect to a random set [49].

The other features of Mantis include easy entry/exit of nodes into the network, Minimal back channel overhead since, the jondos only pass search / request responses and control data. As this system is lenient in providing absolute anonymity to clients, it may lead to some vulnerability. This system tries to solve this drawback by making it difficult to trace the identity of the client.

10 I2P

I2P, also known as Invisible Internet Project, is a pseudonymous, overlay network. It was started in 2003. The main purpose of this network is to enable anonymous communication in a dynamic decentralized network resilient

to attacks. I2P uses a technique called *garlic routing*, for anonymous communication in which, the data is wrapped with several layers of encryption.

The I2P developers are known to general public only with their pseudonyms. The founder of I2P is known as “jrandom” and the project is still under development [28].

It is important to know three critical concepts used by this network for its better understanding. First concept, I2P makes a keen distinction between the nodes participating in the network as “routers” and the anonymous endpoints also known as “destinations”. Secondly, I2P make use of “tunnels” to traverse messages in the network. A tunnel is a direct path selected from the set of routers. There exist two types of tunnels, “inbound” tunnels and “outbound” tunnels. When a sender wants to traverse a message to certain destination, he will select one of his outbound tunnels to forward the message to one of the inner bounds of destination, thereby reaching the destination. And finally, the third concept is “network database” (or “netDb”), maintains a pair of algorithms that are used to share network metadata. The two types of existing metadata are “routerInfo” and “leaseSets”. The routerInfo, as the name states gives the information that is required to contact particular router and the leaseSet gives the necessary information required fro routers to contact a destination [4, 32].

As already stated, the message is wrapped with several layers of encryption so that the message looks entirely different in each hop of the tunnel. One more feature of I2P is that, every participant in the network chooses length of the tunnels, and in doing so, makes a tradeoff between anonymity, latency, and throughput according to their own needs.

11 Entropy

Entropy is a decentralized, peer-to-peer communication network designed to be resistant to censorship and is similar to Freenet [1]. ENTROPY stands for *Emerging Network To Reduce Orwellian Potency Yield* and as such describes the main goal of the project. The aim of this network is to create a “net inside the net” for its users making it very difficult to observe the behaviour of its user. Apart from the anonymity provided by entropy to its users, it will also help its users to hide the content of their files by its distributed feature as in Freenet. Data is encrypted while traversing between two nodes [10].

The decentralized mechanism of entropy makes it difficult for attackers to learn the activities of the nodes in the network but it is important to handle sufficient amount of traffic to provide efficient surveillance to its users in this network. It is stated in, that the creator of entropy stops his work on entropy project since from 2004 [1].

12 Nodezilla

Nodezilla is a peer-to-peer system with secured, distributed and fault tolerant routing system. The main purpose is to serve as a link for distributed services built on top of it providing anonymity for participants. The services offered by this network include Anonymous file sharing, Hierarchical multimedia streaming, and Digital photo sharing with selected friends [1].

Nodezilla uses Freenet technology in its routing mechanism to offer various distributed services anonymously. Cryptography is an important part in Nodezilla’s routing. Unlike Freenet, when a node receives a packet and cannot forward that packet further in the network, then it will assume that it is the packet’s intended recipient [52]. As in Freenet, Nodezilla also provide cache features; a server will copy all the data that passes through it so that the future retrieval of that file will be more efficient increasing the robustness of the system. We can get more information on Nodezilla, from its official website [11].

Chapter 5

Acknowledgement

My Sincere thanks to my supervisor Dr. Apostolas Georgakis, who suggested me this thesis work and guided me with his valuable suggestions and comments that helped me in completing this work.

I am grateful to Mr. Perl Lindstrm, Director of Studies at Computer Science Department in Ume University for his continuous guidance throughout my entire Masters Program.

My Special Thanks to Praveen Kumar Nalli, Ravi Mandadi and Purushotham Kothapalli, for their advices and suggestions. They also helped me in settling down in Sweden.

I would like to thank my Classmates in under graduation Chaitanya, Nikhil, Satish, Pradeep and Jagadeesh, who helped me to improve my report by proof reading and correcting grammar.

I should also thank my friends in Ume Krishna, Brahmaiah, Chandra and Srinivas for all their help, support and nice weekend parties.

Finally, I thank God for my existence and my heart-felt thanks to my family who supported me throughout my life.

References

- [1] <http://www.wikipedia.com>.
- [2] http://www.foi.se/FOI/templates/Page_4068.aspx, accessed 2006-12-08.
- [3] http://whatis.techtarget.com/definition/0,289893,sid9_gci212230,00.html, accessed 2006-12-19.
- [4] <http://www.i2p.net>, accessed 2006-10-31.
- [5] <http://iridia.ulb.ac.be/~mdorigo/ACO/about.html>, accessed 2006-11-23.
- [6] <http://mute-net.sourceforge.net/>, accessed 2006-11-04.
- [7] <http://www.freehaven.net/index.html>, accessed 2006-11-10.
- [8] <http://waste.sourceforge.net/>, accessed 2006-11-02.
- [9] <http://gnunet.org/>, accessed 2006-11-02.
- [10] <http://entropy.stop1984.com/en/home.html>, accessed 2006-11-03.
- [11] <http://www.nodezilla.net/>, accessed 2006-11-10.

- [12] LNCS 3485. *Peer-to-Peer Systems and Applications*. Springer-Verlag Berlin Heidelberg, 2005.
- [13] A. Beimel and S. Dolev. Buses for Anonymous Message Delivery. In *Second International Conference on FUN with Algorithms*, pages 1–13, Elba, Italy, 2001. Carleton Scientific.
- [14] K. Bennett and C. Grothoff. GAP - practical anonymous networking. In *Proceedings of Privacy Enhancing Technologies workshop (PET 2003)*, London, UK, 2003. Springer-Verlag, LNCS 2760.
- [15] K. Bennett, C. Grothoff, T. Horozov, and I. Patrascu. Efficient Sharing of Encrypted Data. In *ACISP '02: Proceedings of the 7th Australian Conference on Information Security and Privacy*, pages 107–120, London, UK, 2002. Springer-Verlag.
- [16] N. Borisov and J. Waddle. Anonymity in Structured Peer-to-Peer Networks. Technical report, EECS Department, University of California, Berkely, California, 2005.
- [17] C. Bornstein, G. Oberoi, and C. Reis. Anonymization of Network Usage Through Peer-to-Peer Proxies. Technical report, Dept. of Computer Science, Rice University, Houston, USA.
- [18] T. Chothia. Analysing the MUTE Anonymous File-Sharing System Using the Pi-calculus. In *26th Conference on Formal Methods for Networked and Distributed Systems (FORTE 2006)*, LNCS, Amsterdam, 2006.
- [19] E. Damiani, D.C. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks. In *CCS '02: Proceedings of the 9th ACM conference on Computer and Communications Security*, pages 207–216, Washington, DC, USA, 2002. ACM Press.
- [20] H. Delfs and H. Knebl. *Introduction to Cryptography: Principles and Applications*. Springer, 2002.
- [21] R. Dingledine, M. J. Freedman, and D. Molnar. The Free Haven Project: Distributed Anonymous Storage Service. In *International workshop on Designing privacy enhancing technologies*, pages 67–95, New York, NY, USA, 2000. Springer-Verlag New York, Inc.
- [22] R. Dingledine, M.J. Freedman, and D. Molnar. *PEER-TO-PEER Harnessing the Power of Disruptive Technologies*. O'Reilly, Sebastopol, CA, USA, first edition, 2001.
- [23] R. Dingledine and P. Syverson. Reliable MIX Cascade Networks through Reputation. In *Proceedings of Financial Cryptography*, LNCS 2357, 2002.
- [24] R.R. Dingledine. The Free Haven Project: Design and Deployment of an Anonymous Secure Data Haven. Master's thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA, 2000. under supervision of Ronald Rivest.
- [25] J. R. Douceur. The Sybil Attack. Presentation by Lakshmi Santhanam.
- [26] T. Dubendorfer and A. Wagner. Past and Future Internet Disasters: DDOS attacks survey and analysis, 2003. *A talk in seminar of "Security Protocols and Applications"*.
- [27] M. Engle and J.I. Khan. Vulnerabilities of P2P Systems and a Critical look at Their Solutions. Technical report, Dept. of Computer Science, Kent State University, Kent, OH, USA, 2006.
- [28] H. Erkkonen and J. Larsson. Anonymous Networks : Onion Routing with TOR, Garlic Routing with I2P. Technical report, Chalmers University of Technology, Gteborg, Sweden.
- [29] S. Gritzalis. Enhancing Web privacy and anonymity in the digital era. *Information Management & Computer Security*, pages 255–288, 2004. Emerald Group Publishing limited.

- [30] C. Grothoff, K. Grothoff, T. Horozov, and J. T. Lindgren. An Encoding for Censorship-Resistant Sharing, 2003.
- [31] M. Gunes, U. Sorges, and I. bouazzi. ARA - The Ant-Colony Based Routing Algorithm for MANETs. In *ICPPW '02: Proceedings of the 2002 International Conference on Parallel Processing Workshops*, pages 79–86, Washington, DC, USA, 2002. IEEE Computer Society.
- [32] jrandom(Pseudonym). Introducing I2P : A scalable framework for anonymous communication, 2006. http://dev.i2p.net/cgi-bin/cvsweb.cgi/i2p/router/doc/techintro.html?rev=HEAD#_intro, accessed 2006-10-31.
- [33] E. Kay, J. Nelson, B. Wu, and V. Yeung. Anonymous Routing in a Peer-to-Peer Network. Technical report, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2004.
- [34] D. Klinedinst. A New Generation of File Sharing Tools, 2003. As part of GIAC practical repository.
- [35] B. Lipinski and P. MacApline. A Security Review of an Anonymous Peer-to-Peer File Transfer Protocol. Technical report, Rice University, Houston, TX, USA.
- [36] D.S. Milojevic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-Peer Computing. Technical report, HP Laboratories Palo Alto, USA, 2003.
- [37] A. Mislove, G. Oberoi, A. Post, C. Ries, and P. Druschel. AP3: Cooperative, decentralized anonymous communication. In *EW11: Proceedings of the 11th workshop on ACM SIGOPS European workshop: beyond the PC*, pages 30–36, New York, NY, USA, 2004. ACM Press.
- [38] M.J.Freedman. Design and Analysis of an Anonymous Communication for the Free Haven Project. Technical report, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, May 2000.
- [39] M. Morain, V. Titov, and W. Verbuggen. Onion Routing for Anonymous Communications. <http://ntrg.cs.tcd.ie/undergrad/4ba2.05/index.html>, accessed 2006-10-26.
- [40] A.L. Nash. Attacking P2P Networks. Securing p2p networks, 2005.
- [41] A. Pfitzmann and M. Hansen. Anonymity, Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology. 2005. http://dud.inf.tu.dresden.de/Anon_Terminology.shtml.
- [42] B. Pretre. Attacks on Peer-to-Peer Networks. Master’s thesis, Swiss Federal Institute of Technology(ETH), Zurich, Switzerland, 2005.
- [43] M. O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. ACM*, pages 335–348, 1989.
- [44] J. F. Raymond. Traffic Analysis: Protocols, Attacks, Design Issues and Open Problems. In *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, LNCS, pages 10–29, New York, NY, USA, 2001. Springer-Verlag New York, Inc.
- [45] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Transactions on Information and System Security*, pages 66–92, 1998.
- [46] C. A. Soghoian S. C. Bono and F. Monrose. Mantis: A Lightweight, Server-Anonymity Preserving, Searchable P2P Network. Technical report, Information Security Institute of The Johns Hopkins University, Baltimore, Maryland, 2004.
- [47] V. Scarlata, B. N. Levine, and C. Shields. Responder Anonymity and Anonymous Peer-to-Peer File Sharing. In *ICNP '01: Proceedings of the Ninth International Conference on Network Protocols*, pages 272–281, Washington, DC, USA, 2001. IEEE Computer Society.

- [48] O.A. Skaflestad and N. Kaurel. Peer-to-Peer Networking Configuring Issues and Distributed Processing. SIE50AC, Self Configuring Systems, NTNU, Norway, 2001.
- [49] C. Soghoian. Mantis :A server-anonymity preserving, searchable, P2P network, 2000.
- [50] M. Suvanto. Privacy In Peer-to-Peer Networks. Technical report, Helsinki University of Technology, Helsinki, 2005. *Seminar on Internetworking*.
- [51] M. Tanase. IP Spoofing: An Introduction. SecurityFocus, 2003. <http://www.securityfocus.com/infocus/1674>, accessed 2006-11-28.
- [52] T.Chothia and K. Chatzikokolakis. A Survey on Anonymous Peer-to-Peer File-Sharing. In *IFIP International Symposium on Network-Centric Ubiquitous Systems (NCUS 2005)*, LNCS, 2005.
- [53] S. A. Theotokis and D. Spinellis. A Survey of Peer-to-Peer Content Distribution Technologies. *ACMCS*, pages 335–371, 2004.
- [54] H. Wen. Anonymous, Open Source P2P with MUTE, 2004. <http://www.onlamp.com/pub/a/onlamp/2004/08/12/mute.html?page=1>, accessed 2006-11-04.
- [55] M. Wieczorek. Anonymous Communication with Waste, 2005. <http://www.marktaw.com/technology/AnonymousWaste.html>, accessed 2006-11-02.