A Survey of Peer-to-Peer Network Security Issues

James Li

Abstract

In recent years, peer-to-peer (P2P) networks have soared in popularity in the form of file sharing applications. With this popularity comes security implications and vulnerabilities. In this paper, we examining the framework on which most P2P networks are built, and from this, we examine how attacks on P2P networks leverage the very essence of the networks itself: decentralization of resources and of control. Additionally, we look at the privacy and usage attacks that arise in P2P networks as well as approaches that can be used address some of these issues.

Table of Contents

- 1. Introduction
  - 1.1 Definition of P2P
- 2. Background of P2P Networks
  - 2.1 Applications of P2P Networks
  - 2.2 Centralized Directory
  - 2.3 Query Flooding
  - 2.4 Distributed Hash Table
- 3. Attacks on P2P Networks
  - 3.1 Distributed Denial-of-Service
  - 3.2 Poisoning the Network
  - 3.3 Privacy and Identity
  - 3.4 Fairness in Sharing
  - 3.5 Blocking of P2P Traffic
- 4. Securing P2P Networks
  - 4.1 Encrypting P2P Traffic
  - 4.1 Anonymous P2P
- 5. Summary
- 6. References
- 7. List of Acronyms

1. Introduction

In a traditional computer network, one or more central servers typically provide all of the services available on the network. An example of this is the numerous FTP (File Transfer Protocol) and HTTP (HyperText Transfer Protocol) servers on the Internet that provide file resources for download from clients seeking these services. In contrast to this client-server model of a network, another approach is to distribute the brunt of providing services among the nodes, or peers, such that each node is both a client and a server. This type of network is called a peer-to-peer (P2P) network.

1.1 Definition of P2P

More technically, a P2P network is a special type of computer network that exhibits self-organization, symmetric communication, and distributed control [Risson04]. The network is self-organizing in that there is typically no centralization of resources. As a result, link capacity is typically distributed throughout peers in the network, and as a result control is distributed, as well. As such, the P2P network model stands in direct contrast to the traditional client-server networking model. Whereas a client-server network requires that the server has copious link capacity to feed clients, a P2P network pools the resources of each peer for the common good. However, due to the decentralized and peer-relying nature of P2P networks, they are also susceptible to attacks, which we will explore in this paper.

First, we present some background on P2P networks, including its inception, rise in popularity as an application, and its

querying structure. Next, we examine different ways that P2P networks are often attacked, including denying services, contaminating the network, and compromising personal information of the peers. Finally, we look at some solutions to the attacks and security issues.

Back to Table of Contents

---

2. Background of P2P Networks

The notion of P2P was first established in 1969, in the first Request for Comments, RFC 1. The RFC implies a "host-to-host" connection, indiscriminate of a client-server categorization, which provides responses in the fashion of teletype (TTY) terminals [Peer07] [Crocker69]. However, the first true implementation of a P2P network was Usenet, developed in 1979 [Sundsted01]. In Usenet, while end-user clients still access resources through servers, servers themselves peer with each other in the fashion of a P2P network, sending messages to each other on demand without a central authority [Usenet07].

2.1 Applications of P2P Networks

Since the late 1990s, there has been a surge of popularity in P2P network applications, mainly in the form of file sharing applications used to exchange multimedia files. Some of the most popular and high-profile file sharing protocols include Freenet, Napster, Direct Connect, Gnutella, eDonkey2000, and BitTorrent. By some estimates, file sharing accounts for more traffic than any other application on the Internet [Kurose05]. By far, the recent rise in research interest generated in the P2P field has come from the popularity of file sharing systems. Below is a table of the timeline of development of the most influential of these P2P protocols [Peer07]:

First released	P2P Protocol
July 1999	Freenet
September 1999	Napster
November 1999	Direct Connect
March 2000	Gnutella
September 2000	eDonkey2000
April 2001	BitTorrent

Table 1: timeline of first release dates of popular P2P protocols

Interestingly, the early file sharing application, Napster, was really more of a directory service than a pure P2P system. Nonetheless, Napster opened the way to more advanced approaches to file sharing, as seen with the subsequent of applications such as Gnutella, eDonkey, and, currently the most popular, BitTorrent. While these applications are all considered P2P applications, peer and resource discovery is a distinguishing feature of different P2P networks, as explained below.

2.2 Centralized Directory

One major issue with any P2P system is the discovery of peers and resources in the network. Since there are no fixed servers, peers must rely on some method to locate fellow peers. The most basic approach is a centralized directory where resources are indexed on a central server, and peers query this server for a lookup to find the peer with the desired resource, then make a connection to the peer [Kurose05]. This approach was taken by Napster, for example. BitTorrent also uses a centralized directory server, calling it the tracker. Note that while resource lookup is still client-server, the actual resource transmission, which accounts for the bulk of the network capacity usage, is still P2P. Below is a diagram of the basic layout of this type of network:

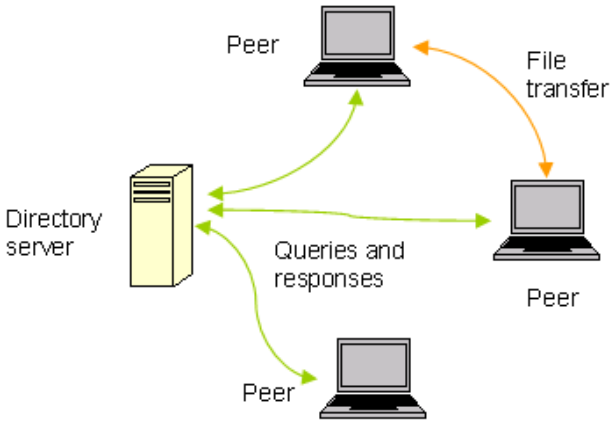

Figure 2: a centralized directory server network

2.3 Query Flooding

Another approach towards peer discovery is query flooding, which is used by newer applications such as Gnutella. The premise here is that instead of relying on a central directory server, a peer would directly broadcast a query to the network, and whomever has the desired resource would respond. Notably, in this approach, there is no central point of failure. However, flooding the network has bandwidth usage considerations that could as well lead to an unintended self distributed denial-of-service (DDoS) attack on the network (a network storm). A variant of the query flooding approach is to select certain, high-availability and high-capacity nodes, as supernodes. These supernodes are given the task of indexing peers within its own domain and answering and creating queries from and to other supernodes. This approach reduces bandwidth usages by a large margin, but it does not really remove the inherent problems with query flooding [Kurose05].

2.4 Distributed Hash Table

Distributed hash tables (DHT) have been introduced around 2001 via the projects Chord, Kademlia, Pastry, and Tapestry. A DHT is essentially a hash table, possessing key-value lookup functionality, with the index distributed among peers in a group. There are variations in the hash function, but the general idea is to minimize the number of peer lookups upon querying for a resource. Typically, *O(log n)* lookups given *n* nodes are needed for a query [Bala03]. DHT systems essentially distribute the centralized directory approach, eliminating a single point of failure. Most of the newer P2P protocols, including trackerless BitTorrent, have been updated to support DHT lookups. Below, we have a sample lookup operation originating from node 2 until the desired resource is found on node 37:

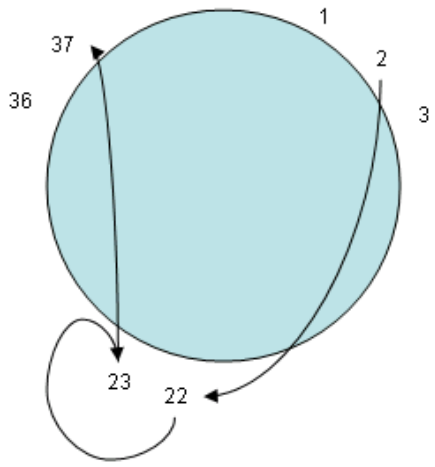

Figure 2: a DHT query from node 2 to 22 to 23 to 73.

Despite these different methods of querying, the actual transmission of resources is still done in P2P fashion, whether using centralized directory, query flooding, or DHT. Unfortunately, attacks have been found that can be used to disrupt or disable the P2P network.

Back to Table of Contents

3. Attacks on P2P Networks

Since P2P systems inherently rely on the dependence of peers with each other, security implications arise from abusing the trust between peers. In a traditional client-server model, internal data need not be exposed to the client, but with P2P, some internals must be exposed to fellow peers in the name of distributing the workload [Schoder05]. Attackers can leverage this in compromising P2P networks.

3.1 Distributed Denial-of-Service

In a traditional denial-of-service (DoS) attack, a server is usually the target of massive connections, rendering the server inoperable. A classic example of this is a TCP SYN flood attack [Naoumov06], in which the client sends the server a SYN message, the server responds with a SYN-ACK message, and the server awaits an ACK message from the client. However, the attacking client simply does not reply with an ACK message, hence tying up server resources (memory) as it futilely waits. Meanwhile, the client can continue to open many more new non-ACK'ed connections, bring the server ultimately to its knees, and hence a denial-of-service to other legitimate clients. In a P2P network, attackers can make use of the querying nature of P2P networks to overload the network. In the case of the query flooding P2P network, the attack is straightforward: simply send a massive number of queries to peers, and the resulting broadcast storm will render portions of the network inoperable.

More recently, attacks can harness the P2P network as an agent to attack some *other* target, such as web sites. Essentially, peers in the network are subverted, as explained in the next section, to request files from a target, overwhelming the victim with enormous bandwidth usage. An example of this kind of attack surfaced in 2007 in the Direct Connect network with users using the DC++ file sharing application [DDoS07]. Below is a figure of a DDoS attack of this kind:

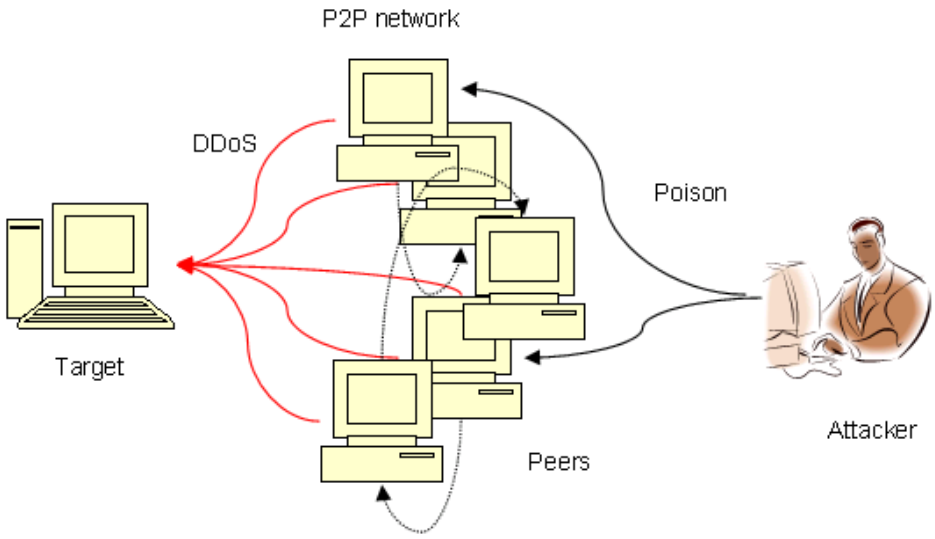

Figure 3: a DDoS attack in effect via poisoning the peers (dotted lines are transferred poisoned records)

3.2 Poisoning the Network

Another approach towards attacking a P2P network is to inject useless data (poison) into the system. Since P2P networks must implement a lookup service in some way, whether it be a centralized directory or a DHT, an attacker can inject large amounts of useless lookup key-value pairs into the index. Bogus items in the index could slow down query times or, worse, yield invalid queries results. Even DHTs are not immune to this attack, but since DHTs have *O(log n)* lookup time, a large amount of poison is required. In fact, poisoning a P2P network has already been witnessed on the Internet as large publishing organizations attempt to lessen the potential losses of pirated media by attacking the FastTrack P2P network [Liang06].

Poisoning can also be used as fodder for DDoS attacks. This can be accomplished in two ways, by index poisoning or route table poisoning. In index poisoning, fake records are inserted into the index pointing to a target IP and port number. When a

peer goes to search for a resource, it would receive bogus location information from a poisoned index, either from a central directory or from another peer. The requesting peer then makes a connection to the target, perhaps confusion the target or, if the target accepts the connection, a TCP-connection DDoS comes into effect. In route table poisoning, the attack leverages the fact that almost all P2P clients need to maintain some kind of routing state of the current peers with which it is connected. Particularly in a DHT system, the route table of each peer contains its *O(log n)* neighbors given *n* nodes in the network. The attacker dupes peers into adding bogus neighbors into each peer's route table, and in some cases, this as simple as making an announce message pointing to the target. The result is that the target receives a flood of connection requests, and the target will likely reject them. Typically, P2P protocols have a mechanism to remove stale peers from the routing table, updating it constantly. Thus, after the burst of traffic to the target, the target is removed from the route tables of connecting peers [Naoumov06].

3.3 Privacy and Identity

P2P networks also present privacy and identity issues. In respect to privacy, a peer's data stream may be compromised by fellow peers who assist in transmitting the data. A direct example is that of VoIP applications, such as Skype, which route traffic in a P2P fashion. Though the data stream is encrypted, a peer which carries the stream now has direct access to the data packets, which would not be the case in traditional routing. Furthermore, Skype's encryption scheme is proprietary, so there can be no verification that the method is completely secure [Suvanto05]. Also, inherent in the nature of P2P applications is the open sharing of private files. In a survey of users, a very small minority were actually aware of the specific files that the user was sharing. In another study on the Kazaa network, many peers were found to be unknowingly sharing their financial, email, and web cache data [P2P04]. Due to the ease-of-use of typical file sharing applications, many users may very well not be savvy enough to realize the privacy implications of using a P2P application, making the job of the attacker very easy.

In P2P networks which distribute resources of dubious legality, the issue of lack of anonymity becomes apparent. For example, the BitTorrent file sharing system directly exposes the IP address of peers to each other in a swarm. This would allow peers in the swarm to know the identity of other peers who are downloading certain resources, for example. Once the peer's identity is compromised, further attacks, whether physical or legal, can continue to be directed at that specific target.

3.4 Fairness in Sharing

Since P2P networks depend on the cooperation of its peers, an assumption is made that all peers to should contribute to the resource distribution process. However, since there is no authority in the system, no real administrator, peers are sometimes free to freeload off other peers. In the file sharing community, this is typically called leeching and is frowned upon and considered cheating. While extremely prevalent in older P2P networks, including the IRC (Internet Relay Chat) network, leeching has been somewhat mitigated in newer P2P applications. For example, in BitTorrent, a choke system is in place to throttle bandwidth to peers who do not upload a fair amount. Thus, leechers are able to do so for a short amount of time before other peers learn of its presence and subsequently refuse to cooperate with it, sharing with it at an increasingly slower rate.

3.5 Blocking of P2P Traffic

An important issue that looms over P2P networks is blocking and throttling of P2P traffic. According to a 2007 Internet study, 69% of Internet traffic in Germany is P2P, with HTTP way behind at 10%. Within P2P traffic, BitTorrent accounts for 67%, with the next highest being eDonkey at 29% [Internet07]. Given the staggering proportion of Internet traffic accounted for by P2P applications, especially BitTorrent (from the numbers above, BitTorrent alone accounts for nearly 50% of the Germany's Internet traffic), it is not surprising that ISPs are starting to block ports on which well-known file sharing applications run. For example, Comcast recently started to throttle and drop packets of BitTorrent traffic, effectively blocking its customers from running the software [Comcast07]. Going even further, Ohio University recently started to block all P2P traffic on its campus [Ohio07].

While security issues with P2P are becoming increasingly rampant, recent efforts have attempted to nullify some of the above vulnerabilities by securing P2P networks.

Back to Table of Contents

---

4. Securing P2P Networks

Given the security issues with P2P networks described above, there are two straightforward approaches to securing P2P networks: encrypting P2P traffic and anonymizing the peers.

4.1 Encrypting P2P Traffic

By encrypting P2P traffic, the hope is that not only will the data be safely encrypted, but more importantly, the P2P data stream is encrypted and not easily detectable. With the actual connection stream completely encrypted, it becomes much harder for the P2P traffic to be detected, and, thus, attacked, blocked, or throttled. A very good example of development in this arena is encrypted BitTorrent, which can encrypt both the header and the payload. Using only 60-80 bits for the cipher, the aim is not to protect the data but instead to simply obfuscate the stream enough so that it is not detectable without incurring much of a performance hit. Although it is still possible to detect encrypted BitTorrent streams using sophisticated methods based on pattern and timing of the traffic, in practice, it is much harder to filter encrypted streams now [PE07]. Encryption of P2P traffic seems to be picking up, as currently about 20% of BitTorrent traffic is encrypted [Internet07].

4.2 Anonymous P2P

By anonymizing peers, the P2P network can protect the identity of nodes and users on the network, something that encryption only cannot ensure. While true anonymity cannot really exist on a network, an anonymous P2P provides enough anonymity such that it is extremely difficult to find the source or destination of a data stream. It does this by making all peers on the network universal senders and universal receivers, thus making it practically impossible to determine if a peer is receiving a chunk of data or simply passing it through. It is not possible to rely solely on anonymous P2P to hide the file sharing application's use without using encryption [Anon07]. However, using encryption together with anonymous P2P would yield possibly the most secure P2P usage experience available today.

Back to Table of Contents

---

5. Summary

We have gone over some of the basics of P2P networking and examined some attacks and issues with P2P networks. A key problem for securing P2P networks is that, because of its inherent decentralized nature, there lacks the means for central administration, and thus control, required to combat security attacks [Friedman03]. Nonetheless, by securing the P2P network using encryption and anonymous systems, some attacks and most of the privacy and usage issues can be address. To combat the direct attacks on P2P networks, careful design and implementation of protocols is required of P2P designers. Thus, P2P network security will continue to be an important issue, as attacks will no doubt become increasingly sophisticated and designers retort with cleverer P2P protocols.

Back to Table of Contents

---

6. References

- [Risson04] J. Risson, T. Moors. "Survey of Research Towards Robust Peer-to-Peer Networks: Search Methods." Technical Report, University of New South Wales, Sydney, Australia. 2004. http://www.cs.umd.edu/projects/p2prg/p2p-overview.pdf
- [Peer07] "Peer-to-peer." Wikipedia. 2007. http://en.wikipedia.org/wiki/Peer-to-peer
- [Crocker69] S. Crocker. "Host Software." RFC 1. 1969. http://www.faqs.org/ftp/rfc/rfc1.txt
- [Sundsted01] T. Sundsted. "The Practice of Peer-to-Peer Computing: Introduction and History." IBM develeoperWorks. 2001. http://www.ibm.com/developerworks/java/library/j-p2p/
- [Usenet07] "Usenet." Wikipedia. 2007. http://en.wikipedia.org/wiki/Usenet

- [Friedman03] A. Friedman, J. Camp. "Peer-to-Peer Security." Harvard University. 2003. http://allan.friedmans.org/papers/P2Psecurity.pdf
- [Kurose05] J. Kurose, K. Ross. Computer Networking: A Top-Down Approach Featuring the Internet. Addison-Wesley. 2005.
- [Bala03] M. Balakrishna, M. Kaashoek, D. Karger, R. Morris, I. Stoica. "Looking Up Data in P2P Systems." Communications of the ACM. 2003. http://www.project-iris.net/irisbib/papers/dht:cacm03/paper.pdf
- [Schoder05] D. Schoder, K. Fischbach. "Core Concepts in Peer-to-Peer (P2P) Networking." http://www.idea-group.com/downloads/excerpts/Subramanian01.pdf
- [Naoumov06] N. Naoumov, K. Ross. "Exploiting P2P Systems for DDoS Attacks." International Workshop on Peer-to-Peer Information Management. 2006. http://cis.poly.edu/~ross/papers/p2pddos.pdf
- [DDoS07] "P2P Networks Hijacked for DDoS Attacks." Netcraft. 2007. http://news.netcraft.com/archives/2007/05/23/p2p_networks_hijacked_for_ddos_attacks.html
- [Liang06] J. Liang, N. Naoumov, K. Ross. "The Index Poisoning Attack on P2P File-Sharing Systems." Infocom. 2006. http://cis.poly.edu/~ross/papers/poison.pdf
- [Suvanto05] M. Suvanto. "Privacy In Peer-to-Peer Networks." Helsinki University of Technology. 2005. http://www.tml.tkk.fi/Publications/C/18/suvanto.pdf
- [P2P04] "P2P or Peer-to-Peer Safety, Privacy and Security." Federal Trade Commission. 2004. http://www.ftc.gov/os/comments/p2pfileshare/OL-100005.pdf
- [Internet07] "Internet Study 2007." ipoque GmbH. 2007. http://www.ipoque.com/userfiles/file/Internet_study_2007_abstract_en.pdf
- [Comcast07] "Comcast Continues to Block Peer to Peer Traffic." Slashdot. 2007. http://yro.slashdot.org/article.pl?sid=07/12/01/0011253
- [Ohio07] " Ohio University Blocks P2P File Sharing." Slashdot. 2007. http://yro.slashdot.org/article.pl?sid=07/04/25/219257
- [PE07] "BitTorrent Protocol Encryption." Wikipedia. 2007. http://en.wikipedia.org/wiki/BitTorrent_protocol_encryption
- [Anon07] "Anonymous P2P." Wikipedia. 2007. http://en.wikipedia.org/wiki/Anonymous_P2P

Back to Table of Contents

---

7. List of Acronyms

- ACK - Final message in a TCP connection
- DDoS - Distributed Denial-of-Service
- DoS - Denial-of-Service
- DHT - Distributed Hash Table
- FTP - File Transfer Protocol
- HTTP - Hypertext Transfer Protocol
- IRC - Internet Relay Chat
- ISP - Internet Service Provider

- P2P - Peer-to-Peer
- RFC - Request for Comments
- SYN - First message in a TCP connection
- SYN-ACK - Second message in a TCP connection
- TCP - Transmission Control Protocol
- TTY - Teletype terminal
- VoIP - Voice over Internet Protocol

Back to Table of Contents

Last Modified: Dec 2007

Note: This paper is available online at http://www.cs.wustl.edu/~jain/