Security Enhancement Technique In Peer to Peer Network: A Review

Varun Chanana[1] , Sandeep Kumar[2], Ravikat Jaiswal[3]
[1,2,3] Department of CSE,Ganpati Institute of Technology and Management,Bilaspur,Yamunanagar
[1] varun.chanana88@gmail.com

Astarct: Peer-to-peersystems and applications are used in the Internet to share resources (i.e. computing power, data storage and sharing, and bandwidth) between computers. Resources are therefore distributed all over the P2P network. Pure P2P networks do not have any centralized control or organization. Therefore, they differ fundamentally from the traditional client-server (CS) model. Resources are fully decentralized and the nodes have an equal role, no hierarchy or central servers are needed. Nodes in a P2P system are called peers and they function simultaneously as clients and servers. Between these models lay the hybrid model, where a server is used for lookups of resources, but the data is distributed and transferred in a P2P manner. Napster is an example of a hybrid model. Today many P2P systems are used for file sharing. P2P file sharing networks often involve illegal and copyright-violate sharing of movies, music etc.

Keywords: Super nodes, Structured Networks, Backdoor attacks, Proximity routing**,** Self-stabilization

I. Introduction to Peer-to-Peer Networking

Peer-to-peer (P2P) is an alternative network model to that provided by traditional client-server architecture. P2P networks use a decentralized model in which each machine, referred to as a peer, functions as a client with its own layer of server functionality. A peer plays the role of a client and a server at the same time. That is, the peer can initiate requests to other peers, and at the same time respond to incoming requests from other peers on the network. It differs from the traditional client-server model where a client can only send requests to a server and then wait for the server's response.APeer is one of many entities connected to a P2P network with a P2P application. When referring to a peer it will be both the application and inherently the user of the application that is being referred. In pure P2P networks there are only peers. Other P2P networks rely on centralized servers in one form or the other, or relies on concepts such as special peers referred to as super nodes. When concepts such as super nodes or servers are discussed, they will not be referred to as peers.[1,2]

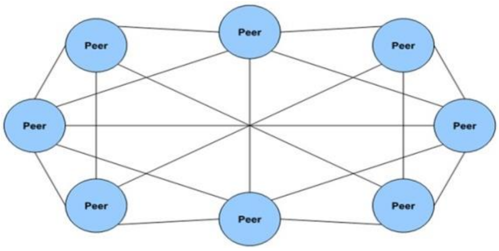

Figure1: Peer-to-Peer Network Architecture

In recent years, Peer-to-Peer (P2P) technologies have become increasingly popular. Peer-to-Peer (P2P) participants share a part of their own hardware resources, such as processing power, storage capacity, or network bandwidth. The service or content provided by the P2P network is accessible by other peers directly, without passing intermediary entities. Peer-to-Peer (P2P) systems make it possible to harness resources such as the storage, bandwidth, and computing power of large populations of networked computers in a cost-effective manner. Actually P2P is a decentralized and distributed and here all the nodes are equivalent. [3]Some operators even obstruct P2P traffic in their network in order to prevent excessive network load that P2P file sharing networks often cause. Measurements show that as much as 60-80% of network traffic is caused by P2P traffic, while only 30-35% of all subscribers use P2P. Bit Torrent, one of the most popular P2P file sharing networks, alone accounted for 30% of all Internet traffic. [4]Bit Torrent and other P2P file sharing networks are just one way to use P2P. P2P networking has also been widely exploited for making voice calls and for instant messaging (IM) over the Internet. Skype is a good example of a P2P system that is widely being used in the Internet. In addition to file sharing and media communication, P2P can also be used e.g. for emergency information flow, SPAM detection filtering as well as for sharing computing power like SETI@home.Participant nodes in a P2P network can be situated all over the world, as long as there are physical links that can be used to interconnect the nodes. In a pure CS model, a resource to be searched is always only one hop away. In a P2P model, a resource to be searched in a P2P overlay network may take one or more hops to be found. Also, as the resources are decentralized and the location information of the resources is distributed, every peer has to participate in other peer's resource lookups. After a resource has been found, usually a direct connection between the two peers can be used. Thereby, peers are usually only helping in resource lookups, but the resource utilization like file download or a voice call is made directly between the corresponding peers. A fully decentralized P2P network is very difficult to shut down, as there are no central servers or other entities that the network is dependent of. In general, P2P networks potentially offer an efficient routing architecture that can be self-organizing, massively scalable and robust. They can also provide good fault-tolerance, load balancing and explicit notion of locality. [5]

II. Client/Server Limitations

- Scalability is hard to achieve
- Presents a single point of failure
- Collaborative applications
- Requires administration [6]

III. P2P Benefits

- Efficient use of resources
- Scalability
- Reliability [7]

IV. P2P applications

- Instant Messaging (IM): technologies for sending nearly instantaneous messages between users. Examples of such software are Microsoft's MSN Messenger, Trillian and ICQ.
- File Sharing: technologies for sharing data between equal peers in large networks; one identifying characteristic of such networks is the lack of any central entity. Examples of such software areKazaa , Shareaza and Limewire.
- Grid Computing: technologies for sharing computer resources, most commonly CPU cycles, among many different systems. This can be used to perform processing of large amounts of data distributed over a large number of computers. An example of such software is the SETI@home project. [10, 11]

V. Properties of P2P Network

Decentralization: The data structure should be distributed among all the participants of the system. A central server, or even a cluster of such servers, may prove to be intolerant to faults, and will require considerable investment for high-performance hardware and high bandwidth. **Scalability:** The Internet user community has grown to be so large that distributed systems need to cope with millions of users. In an ideal peer-to-peer system, the cost borne by each participant should not depend too much on the size of the entire system. **Load balancing**: We would like the cost of maintaining the system to be uniformly shared between all the peers. Similarly, the system should be able to manage flash crowds i.e., high data request volume due to temporal locality, when a particular resource becomes extremely popular for a short period of time **Dynamic maintenance**: The massive parallelism in peer-to-peer systems, due to high rate of machine arrival and departures, presents some very challenging issues that are trivially solved in a system with fixed membership. The system should be self-configuring, and machines and resources should be added and deleted from the system quickly without manual intervention or oversight.

Fault tolerance: The data structure should be resilient to both machine and link failures in the system. Even if a part of the system has failed, the data available in the surviving machines should still be accessible, as long as it is located in the same connected component as the requesting peer. Further, the system should gracefully degrade with increasing failures. [9]

Self-stabilization: Not only should the system survive disruptions due to failures, but it should also heal automatically to restore ideal performance. The system should have a repair mechanism that detects local inconsistencies such as machine failures or link outages, and triggers maintenance operations with minimal overhead in terms of network traffic.

Efficient searching: The primary goal of a peer-to-peer system is to locate resources efficiently, and hence support for searching using a variety of specifications is a very desirable property. Complex queries to locate resources such as range queries, near matches to a key, and keyword matches should be supported by a rich query language. [13]

Security: The system should be secure against attacks such as a denial-of-service attack, where some miscreant participants may "flood" the system, thereby preventing legitimate traffic. In some applications, it may also be desirable to maintain anonymity of the users, or provide resistance to censorship by preventing certain data items to be deleted from the system. [8]

Topologically-sensitive construction: Routing should be sensitive to network locality such as distance traveled or latency along transmission paths. Two possible approaches are: (i) Proximity routing where machines are placed in the network to exploit the underlying topology, and (ii) Proximity neighbor selection where the closest neighbors are chosen among the set of potential neighbors. [12]

VI. P2P Architecture

There are several ways to classify P2P networks. One approach considers the application a P2P network is used for (e.g., file sharing, telephony, media streaming etc.). Another approach includes the degree of centralization and distinguishes between pure P2P without central server (peers act as equals) and networks with central server keeping information on peers. In this context, the following terminology can be found: centralized, or decentralized P2P networks, structured, unstructured, or hybrid (so-called super-peer architectures) P2P networks. P2P overlays can be divided into unstructured and structured overlays, but P2P overlays based on a hybrid structure also exist. Unstructured P2P overlays are divided into two categories: ones with a flooding based lookup method and ones with a random walk-based lookup method. Both of these have random topology of peer connections. Structured P2P overlays with a structured and logical topology are more advanced. Structured P2P overlays can be classified based on the routing mechanism or on the topology. Chord, Pastry, CAN and Kademlia, have a flat topology and the routing of which is classified as second generation multi hop-based routing. This means that they use incrementally converging routing with multiple hops in order to find the target.

VII. Super nodes in a Peer-to-Peer Overlay Network

In a basic P2P overlay network, the peers have equal roles and they participate equally in the lookup queries. However, super nodes can be used together with regular peers. These super nodes are more capable peers and they function as server-like peers causing a hierarchical difference between regular peers and super nodes.

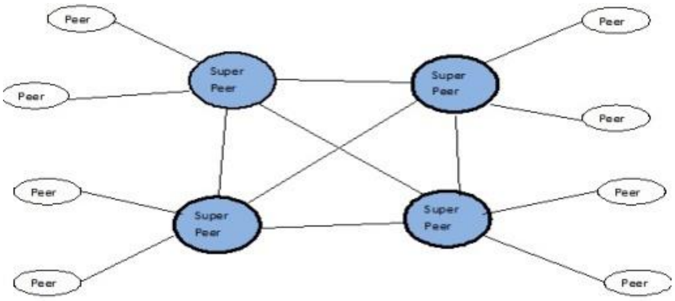

Figure2: Peer-to-Peer Supernode Architecture.

A super node is a well-known P2P node that has some guarantee of high availability, computing resources and available networking bandwidth. Accordingly, they can provide more resources for other peers and they are usually more stable than regular peers. A regular peer may also become a super node, if the requirements are fulfilled. Thereby, it does not necessarily need to have a static public IP address or DNS name for super node, if it is otherwise well-known and has sufficient bandwidth capacity. However, these are useful capacities especially if an operator provides super node functionalities for a network. [14]

Examples of Structured P2P Networks

Many of the first popular P2P networks such as Napster, Gnutella and KaZaA are unstructured. However, most of the recent P2P systems use a more advanced topology of structured algorithms. Bit Torrent and eDonkey are examples of very popular P2P networks that are used for file sharing. For voice and other media communication, Skype has proven the good applicability of P2P networking for communication other than just file sharing. Besides, P2P networks could also be used for emergency information flow as well as for SPAM detection and filtering.

Chord: Chord is a simple and popular structured P2P algorithm that implements the DHT abstraction. Chord can be used to map a given key onto a node i.e. a peer in the context of P2P network in a Chord overlay. This mapping is the only function that the Chord protocol has.

CAN: The Content-Addressable Network (CAN) is basically similar to Chord, but it also has many differences. Rather than a virtual ring, CAN uses d-dimensional Cartesian coordinate space to implement DHT that maps keys onto values. CAN allow nodes to specify their own identity. State maintained by a CAN node does not depend on the network size of N, and the lookup cost increases faster than log N (O(dN1/d)). CAN require an additional maintenance protocol for periodical remapping of the key space. CAN optimizes the forward path by the best round trip time (RTT) of neighbors. This implies that the queries are forwarded without interacting with the querier. Thus, the querier cannot verify the forward process of its lookup and the algorithm is susceptible to misrouting attacks.

Pastry: Pastry is similar to Chord, but differs from it in some details. Pastry is a prefix-based routing protocol and not based on numerical difference like Chord is. Pastry has hybrid tree-ring geometry, while Chord has ring geometry. Pastry is self-organizing and takes proximity into account by using a scalar proximity metric, such as the number of IP routing hops or geographic distance.

Bamboo: The geometry of Bamboo is similar to that of Pastry, as Bamboo also uses hybrid tree-ring geometry. Bamboo is designed to handle churn, which is a big concern in P2P networks. Bamboo has lower routing latency than Chord (and even smaller under churn). This is a good feature especially when considering VoIP. The quick "local tuning" part of the routing algorithm of Bamboo is similar to the routing algorithm of Pastry, but it is incremental and more frequent. "Global tuning" of Bamboo routing maintenance is similar to the stabilization of Chord and is used for optimization of the static network. Tuning is only one part of neighbor discovery and state maintenance. Leaf set maintenance and routing table filling occur before the tuning of routing tables.

Kademlia: Kademlia is a DHT implementation for decentralized computer networks and it has been used for file sharing. The Kademlia algorithm is based on calculating a distance of two node IDs. This distance is used to maintain a similar list to the finger list in Chord. The list is filled from IDs of the requests of reply messages the node receives. Kademlia is resistant to certain DoS attacks, as the list cannot be flushed of valid node-items. Lookup is similar to Chord, but Kademlia can perform multiple parallel requests for the same query. **JXTA:** JXTA is an open source P2P platform that is defined as a set of XML based protocols. JXTA is a very mature P2P framework and it has been designed to enable decentralized communication for a wide range of devices such as PCs, cell phones, PDAs. JXTA is a modular platform that provides simple building blocks for developing a wide range of distributed services and applications. JXTA specifies a set of protocols rather than an API. Thus, JXTA technology can be implemented in any language on any Operating System. JXTAis optimized for frequent churn (i.e. devices joining and leaving the network).

Key privacy and security concerns

- Inadvertent sharing of sensitive personal information
- Installation of spyware or adware that communicates with a third party without the user's knowledge or consent
- Legal risks for those who, knowingly or unknowingly, violate copyright law or share illegal material (copyrighted material)

VIII. Factors that affect security in P2P networks

There are many factors affecting the security of any given P2P system. This section will focus mainly on the P2P software. Open P2P networks are often insecure since users can join without any authentication of their identity or proof that the data they are sharing is not malicious software. It is a known fact that P2P networks are used by malicious users to spread viruses, Trojans and other malicious programs. In this system several computers in the network will disseminate information about probable security attacks to each other; this will ensure a rapid spread of information regarding new attacks between the cooperating nodes. Each node will be responsible for:

1. Detecting whether a virus or worm is propagating through the network and possibly causing an epidemic.
2. To automatically send out warnings and information to other peers connected to the network.
3. Take precautions for protecting its host. This can be done by a stricter security policy during the time span of the suspected epidemic.

The hope is that by gathering this information the nodes will be able to estimate when a new wave of attacks are about to happen, and take appropriate countermeasures without the intervention of the user.

This method can provide protection against the spread of viruses and Trojans, but will not be able to protect an application against attacks that rely on the actions of the user. It would therefore be important to find ways to protect the user from performing actions that would result in an increased chance of exposure to attack. [13, 15]

One such method is to make a trust based system available in P2P networks. This goes for both P2P applications by themselves and the data shared on P2P networks. Today there are few ways to confirm the integrity and authenticity of P2P programs; these are programs that usually require full access and privileges on the host computer to operate in a satisfactory way. Since it is nearly impossible to control that the P2P software itself is secure, it is necessary to have architecture to safely run un-trusted code on. When it comes to protecting the host computer from malicious nodes, there are some methods that can be implemented. When users share their data with others, there is a chance that they accidentally share more data than they know. Windows XP users can reduce the chance of malicious users gaining access to sensitive data by using the built in file-sharing features. They can then designate data as either shared or private. Private data can only be accessed by the machine's owner. User should not depend on the built in protection of the P2P software as it can easily be bypassed by an experienced hacker.

Backdoor attacks are also a common form of attack, not only on P2P networks, but throughout the Internet. As much as 45% of files downloaded from P2P networks have been shown to contain some form of malicious code. Malicious users can disguise viruses and trojans in well-known file formats; this is done with software commonly known as "Wrappers". The most efficient way to defend against such attacks is by having up to date antivirus software. This software will analyze any suspicious files and alert users when it detects malicious code. This means that unknown variations of such malicious code will go undetected.

IX. Possible attacks when using P2P

As with most software implementations today P2P software is insecure. It is widely known that the installation of such software will create new ways for malicious users to cause damage. While some of these weaknesses are relatively unknown by the users and developers, others are known and could have been easily avoided had the developers considered the problem during development. Information leakage is a serious concern when it comes to the use of P2P. Several problems exist with Gnutella that could have been solved in the development phase:

- It announces IP addresses. This represents a serious problem, especially for those networks which do not safeguard their users with hiding processes such as Network Address Translation (NAT) or various other types of proxies. This exposure can have two consequences. The first is the possibility of users being monitored by third parties. The second is that attackers could, once they recognize the IP address used for the connection, use it to perform security probing or more severe attacks.
- It announces full path names, making it possible for attackers to get a complete picture of the system on which the software is running.
- It announces Gnutella topology, which may reflect real-world patterns of association. The worst case scenario would be that attackers get a complete picture of the number and placement of clients on an internal corporate network.
- It can use any port number which makes it very hard to detect and to control outbound connections via the firewall. Gnutella even has a special "Push" command that asks the receiver to establish an outbound connection to the sender of the "PUSH" command, thereby possibly bypassing the firewall.
- An eavesdropper can easily record queries and responses, making it possible to create content that will attract special groups of users (e.g. those who search for a specific type of content) and target these users for attacks.
- The combination of "Query/Push" makes it possible for an attacker to forge the return address, and thereby induce other nodes to try to send a large file to some arbitrary destination. This method has been used to create DoS attacks similar to "FTP Bounce" attacks.
- There is no guarantee that what a user receives is what he wanted. A node can return false content (virus or trojans) or users can receive obscene and possibly illegal content in response to innocent queries.
- Nodes can falsely advertise a high-speed connection to attract more clients, and thereby spreading malicious software quicker.

X. Conclusion

After examined Peer-to-Peer network and problems in Peer-to-Peer network it is concluded that there must be a system that minimize these problems. A reputation system is a good choice for handling these types of problem. Because of the open nature of P2P models, the free-riding phenomenon is popular and degrades the system performance. Anonymity may exacerbate this problem since the free-riders cannot be located, and since selfish behaviors might be prevalent without any punishment. So objective of this dissertation is to design a reputation system for Peer-to-Peer network that can handle free rider problem in this network efficiently with minimum network load.

References

[1] Tseng, T.-Y. Lee, R. Lin, S.-W. Han, T.Huafan Univ., Shihding "Mixed Client Server and Peer to Peer System for Internet Content Providers" 1-4244-0099-6 IEEE 2006

[2] D. Liben-Nowell, H. Balakrishnan, and D. Karger, “Analysis of the evolution of peer-to-peer systems” in Proceedings of the Annual ACM Symposium on Principles of Distributed Computing, Monterey, California, USA, 2002.

[3] RiidigerSchollmeier, Institute of Communication Networks, TechnischeUniversitat "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications" 0-7695-1503-7102 IEEE 2002 , pp.101-102

[4] Doval D. and O’Mahony D., “Overlay networks: A scalable alternative for P2P”. IEEE Internet Computing, vol. 7, no. 4, pp. 79-82, 2003.

[5] Min Yang Yuanyuan Yang "Peer-to-Peer File Sharing Based on Network Coding" 978-0-7695-3172-4 IEEE 2008, pp.168-173

[6] Damiani E., Capitani di Vimercati S., Paraboschi S. and Samarati P., “P2P-based collaborative spam detection and filtering”. In Proc. 4th IEEE Conf. on P2P, Zurich, Switzerland, August 2004.

[7] DongyuQiu, R. Srikant "Modeling and performance analysis of BitTorrent-like peer-to-peer networks" 1-58113-862-8 Volume 34 Issue 4 ACM, October 2004

[8] Reckerd, D. Vico, J. "Application of peer-to-peer communication, for protection and control, at Seward distribution substation" 0-7803-8896-8 IEEE 2005, pp. 40-45

[9] B. Pourebrahimi K. Bertels S. Vassiliadis "A Survey of Peer-to-Peer Networks" In ACM Wireless Networks, Vol. 3, no. 5, pp. 589-606, 2005

[10] LoubnaMekouar, Youssef Iraqi, RaoufBoutaba "Peer-to-peers most wanted: Malicious peers" Computer Networks 50 (2006) ScienceDirect, pp. 545–562

[11] Schäfer, K. Malinka, P. Hanácek"Peer-to-peer Networks Security" The Third International Conference on Internet Monitoring and Protection, 978-0-7695-3189-2/08 IEEE 2008, pp.74-79

[12] Murali Krishna Ramanathan, VanaKalogeraki, Jim Pruyne "Finding Good Peers in Peer-to-Peer Networks" Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS?02), 1530-2075/02 IEEE 2002

[13] Sepandar D. Kamvar, Mario T. Schlosser, Hector GarciaMolina "The EigenTrust Algorithm for Reputation Management in P2P Networks", ACM May 20–24, 2003

[14] Sergio Marti, Hector Garcia-Molina "Taxonomy of trust: Categorizing P2P reputation systems" Computer Networks 50 (2006) ScienceDirect, pp. 472–484

[15] PrashantDewan and ParthaDasgupta "P2P Reputation Management Using Distributed Identities and Decentralized Recommendation Chains" TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 22, NO. 7, IEEE JULY 2010,pp.1000-1014.