ISSN: 2393-8528

IJICSE
Open Access

International Journal of Innovative Computer Science & Engineering (IJICSE)

Available Online at www.ijicse.in

# Analyzing Classification and Security Issues of Peer to Peer Networks

**Hemant Gaur [1], Chandra Prakash Verma[2], Kapil Saini[3]**

[1] M Tech Student, Bhagwant University, Ajmer, Rajasthan, India
hgaur91@gmail.com

[2]M Tech Student, Govt. Engineering College, Ajmer, Rajasthan, India
chandra.rbs@gmail.com

[3]M Tech Student, Govt. Engineering College, Ajmer, Rajasthan, India
kapil88888@yahoo.com

**ABSTRACT**

Peer-to-Peer systems have become, in a short period of time, one of the fastest growing and most popular applications. Peer-to-peer systems have emerged as a popular way to share huge volumes of data. Peer-to-Peer (P2P) file sharing is the hottest, fastest growing application on the Internet. The evident success of file sharing networks such as Napster, later followed by Gnutella and Bit-Torrent among others, triggered a new wave of research on this new computing discipline.

In this survey, we propose classification of P2P network based on architecture as well as based on their structure. Although a P2P network has a number of advantages over the traditional client-server model in terms of efficiency and fault-tolerance, additional security threats can be introduced. So this paper also discusses some important security threats and issues with peer to peer and some solutions to them.

So in this paper, the classification and security aspects are underlined, in order to survey p2p systems.

**Key Words:** P2P, Bit-Torrent, Gnutella, fault-tolerance, Threats, Napster

## INTRODUCTION:

P2P networks are a recent addition to the already large number of distributed system models. P2P systems, an alternative to conventional client–server systems, mostly support applications that offer file sharing and content exchange like music, movies, etc. A major benefit of P2P file sharing is that these systems are fully scalable, each additional user brings extra capacity to the system. A node (peer) can act both as a client and a server. The participating nodes mark at least part of their resources as 'shared', allowing other contributing peers to access these resources. Thus, if node A publishes something and node B downloads it, then when node C asks for the same information, it can access it from either node A or node B. As a result, as new users access a particular file, the system's capability to provide that file increases.

The peer-to-peer networks differ from the conventional client-server approach in several ways. The most distinguishing characteristic is that a peer acts both as a client and a server of the system. A second important difference is the transient lifetime of peers and their asynchronous arrivals and departures. A peer-to-peer system as one that satisfies three criteria [4]:

**A.** Self organization: There is no central directory. Nodes organize themselves.

**B.** Symmetric communication: Nodes act both as clients and servers.

**C.** Decentralized Control: No centralized mechanism exists that guides nodes. They decide for themselves how they participate in the network.

Two main key characteristics of peer-to-peer systems [1]:

**A.** Scalability: there is no algorithmic, or technical limitation of the size of the system, e.g. the complexity of the system should be somewhat constant regardless of number of nodes in the system.

**B.** Reliability: The malfunction on any given node will not affect the whole system (or maybe even any other nodes).

There are mainly three different architectures for P2P systems: centralized, decentralized structured and decentralized unstructured (Fig 1). In the centralized model, such as Napster, central index servers are used to maintain a directory of shared files stored on peers with the intention that a peer can search for the location of a

desired content from an index server. Currently, there are two types of P2P lookup services widely used for decentralized P2P systems: structured searching mechanism and unstructured searching mechanism. Roughly speaking, the P2P networks that do not rely on a centralized directory can be classified as either structured or unstructured. Structured P2P networks (e.g., Chord, CAN, Pastry, and Tapestry) use specialized placement algorithms to assign responsibility for each file to specific peers, as well as "directed search" protocols to efficiently locate files. Unstructured P2P networks (e.g., Gnutella and Freenet) have no precise control over the file placement and generally use "flooding" search protocols [3].

In unstructured networks the placement of the available content is not relevant to the network topology.

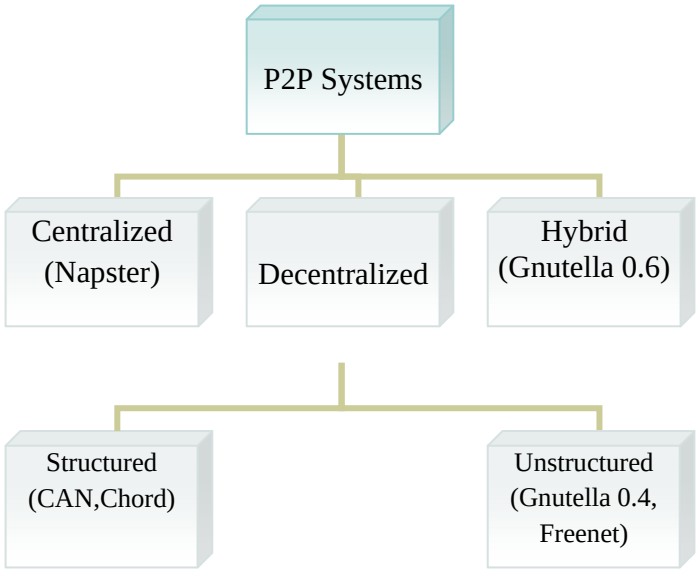


**Figure 1: Classification of Peer to Peer systems**

In a structured network a completely decentralized network topology is used which is tightly controlled by the algorithm. The nodes collaborate in order to maintain this structure while allowing for the system to be dynamic (in the sense that nodes can leave and join at any time).

Throughout this paper the terms "node", "peer" and "user" are used interchangeably, according to the context, to refer to the entities that are connected in a peer-to-peer network.

**1. PEER TO PEER ARCHITECTURES**

A distributed architecture consisting of a collection of resources (computing power, data, meta-data, and network bandwidth) performing a distributed function is called a peer to peer architecture [2].

Currently, there are several different architectures for P2P networks:

A. Centralized Architecture
B. Decentralized Architecture
C. Hybrid Architecture

**2. CENTRALIZED ARCHITECTURE**

The search method in these networks such as Napster is actually based on the client server model. Each node sends an update with its content to the centralized server on joining the network or when its content is updated. The centralized server keeps track of all node content and answers search queries from nodes. The problems associated with this search method are, see [5]:

A. Single point of failure: A crash of the central index paralyzes the entire network since no searches can be initiated.

B. Scalability: Expenses of the centralized index are high when the network expands. This can cause searches to be executed slowly.

C. Legal issues: In some countries the owners of centralized indexes of illegal content can be prosecuted.

**3. DECENTRALIZED ARCHITECTURE:**

In a pure P2P architecture, no centralized servers exist. All peers are equal, hence creating a flat, unstructured network topology. In order to join the network, a peer must first, contact a bootstrapping node (node that is always online), which gives the joining peer the IP address of one or more existing peers, officially making it part of the ever dynamic network. Each peer, however, will only have information about its neighbors, which are peers that have a direct edge to it in the network [1].

Since there are no servers to manage searches, queries for files are flooded through the network. The act of

query flooding is not exactly the best solution as it entails a large overhead traffic in the network. An example of an application that uses this model is Gnutella.

In a fully decentralized system, not only is every host an equal participant, but there are *no* hosts with special facilitating or administrative roles [7]. In practice, building fully decentralized systems can be difficult, and many peer-to-peer applications take hybrid approaches to solving problems.

Many other current peer-to-peer applications present a decentralized face while relying on a central facilitator to coordinate operations. To a user of an instant messaging system, the application appears peer-to-peer, sending data directly to the friend being messaged. But all major instant messaging systems have some sort of server on the back end that facilitates nodes talking to each other. The server maintains an association between the user's name and his or her current IP address, buffers messages in case the user is offline, and routes messages to users behind firewalls.

This hybrid approach seems to scale well, the directory can be made efficient and uses low bandwidth, and the file sharing can happen on the edges of the network [7].

In practice, some applications might work better with a fully centralized design, not using any peer-to-peer technology at all. One example is a search on a large, relatively static database. Current web search engines are able to serve up to one billion pages all from a single place. Search algorithms have been highly optimized for centralized operation; there appears to be little benefit to spreading the search operation out on a peer-to-peer network (database generation, however, is another matter).

Also, applications that require centralized information sharing for accountability or correctness are hard to spread out on a decentralized network. For example, an auction site needs to guarantee that the best price wins; that can be difficult if the bidding process has been spread across many locations. Decentralization engenders a whole new area of network-related failures: unreliability, incorrect data synchronization, etc. Peer-to-peer designers need to balance the power of peer-to-peer models against the complications and limitations of decentralized systems.

## 4. HYBRID ARCHITECTURE

The basic idea behind this architecture is similar to the one behind purely decentralized architectures. Some nodes of the network are now assigned a more important role. They act as central indexes for the surrounding nodes. It is worth-noting that this does not constitute central points of failure though, as such nodes are dynamically assigned and in the case of failure the network takes action to replace them [7].

The way such “super nodes” are selected differs from implementation to implementation. For example, Bit Torrent (BT), where a central server called a tracker helps coordinate communication among BT peers in order to complete a download (as shown in figure 2).

The advantage of hybrid decentralized systems is that they are simple to implement, and they locate files quickly and efficiently.

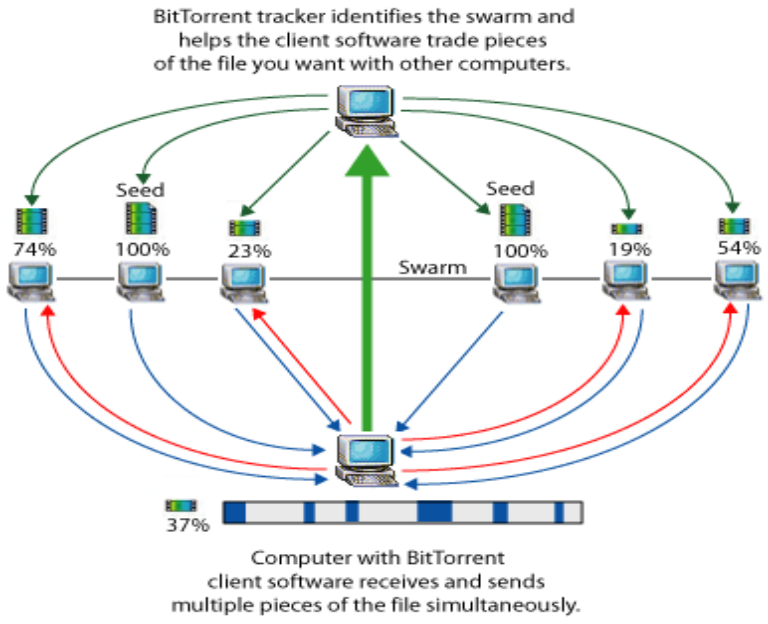


**Figure 2 Bittorrent: hybrid architecture**

Their main disadvantage is that they are vulnerable to censorship, legal action, surveillance, malicious attack, and technical failure, since the content shared, or at least descriptions of it, and the ability to access it are controlled by the single institution, company, or user maintaining the central server. Furthermore, these systems are considered inherently unsalable, as there are bound to be limitations to the size of the server database and its capacity to respond to queries.

## 5. CLASSIFICATION BASED ON STRUCTURE

There are many different p2p systems, each one with various advantages and disadvantages. They differ both in their object query mechanism and in their logical topology. By structure, we refer to whether the overlay network is created non-deterministically (ad hoc) as nodes and content are added, or whether its creation is based on specific rules. We categorize peer-to peer networks as follows, in terms of their structure:

### A. Unstructured Networks

These are systems in which there is neither a centralized directory nor any precise control over the network topology or file placement. Unstructured systems are designed more specifically for the heterogeneous Internet environment, where the nodes' persistence and availability are not guaranteed. Under these conditions, it is impossible to control data placement and to maintain strict constraints on network topology, as structured applications require [2]

In an unstructured P2P network, each node is connected to any nodes arbitrary. There are also no rules of the location in which data or meta-data is placed. To find a file, a node queries its neighbors. The most typical query method is flooding, where the query is propagated to all neighbors within a certain radius (as shown in figure 3). These unstructured designs are extremely resilient to nodes entering and leaving the system [6].

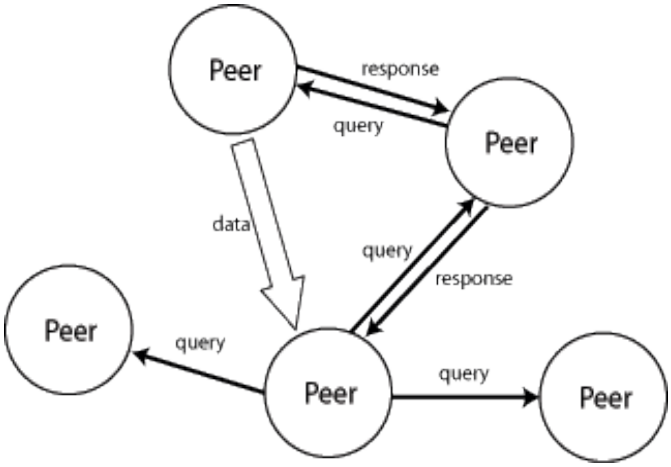


**Figure 3 Decentralized and Unstructured**

However, the flooding-based techniques cause some traffic on the network and delay to search the data or meta-data. On the other hand, since any topology of the network can compose, it is easy to maintain the system.

### B. Structured Networks:

Structured systems are designed for applications running on well-organized networks, where availability and persistence can be guaranteed. In such systems, queries follow well-defined paths from a querying node to a destination node that holds the index entries pertaining to the query [2]. In a structured network a completely decentralized network topology is used which is tightly controlled by the algorithm. The nodes collaborate in order to maintain this structure while allowing for the system to be dynamic (in the sense that nodes can leave and join at any time).

The basic idea behind this approach is to combine the advantages of unstructured networks, but on the same time avoiding the drawbacks. These systems are scalable and efficient, and they guarantee that content can be located within a bounded number of hops. To achieve this performance level, the systems have to control data placement and topology tightly within their networks.

By far the most common type of structured P2P network is the distributed hash table (DHT), in which a variant of consistent hashing is used to assign ownership of each file to a particular peer, in a way analogous to a traditional hash table's assignment of each key to a particular array slot.

Essentially, DHT-based approaches provide a way to map content to location through a universal algorithm. This same algorithm is used for inserting data and nodes deterministically in the network and for locating content. Queries for content then can be routed to the location of the appropriate node without searching since that location is already known through this algorithm [5].

Examples of DHT-based infrastructures include Chord, CAN, Pastry and Tapestry.

## 6. APPLICATIONS OF P2P SYSTEMS

Various applications of different types of peer to peer networks:

A. Freenet is a loosely structured system that uses file and node identifier similarity to produce an estimate of where a file may be located, and a chain mode propagation approach to forward queries from node-to-node [3]

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

B. Chord is a system whose nodes maintain a distributed routing table in the form of an identifier circle on which all nodes are mapped, and an associated finger table [3].

C. CAN is a system using an n-dimensional Cartesian coordinate space to implement the distributed location and routing table, whereby each node is responsible for a zone in the coordinate space [3].

D. Tapestry (and the similar Pastry and Kademlia) are based on the plaxton mesh data structure, which maintains pointers to nodes in the network whose IDs match the elements of a tree-like structure of ID prefixes up to a digit position [3].

E. Napster is a file-sharing P2P application that allows people to search for and share MP3 music files through the vast Internet. It was single handedly written by a teenager named Shawn Fanning (Tyson, 2000 and Shirky, 2001). the most well known example utilizing centralized architecture is Napster [1].

F. BitTorrent is a peer-to-peer file sharing protocol used for distributing large amounts of data. BitTorrent is one of the most common protocols for transferring large files, and it has been estimated that it accounted for roughly 27% to 55% of all Internet traffic (depending on geographical location) as of February 2009.

G. Instead of using a centralized index directory like Napster, Gnutella uses a flat network of peers called servents, to maintain the index directory of all of the content in the system. In a Gnutella network, servents are connected to each other in a flat ad-hoc topology.

## 7. SECURITY ANALYSIS

Peer-to-peer systems (P2P) have grown in importance over the last 5 years as an attractive way to mobilize the resources of Internet users. As more and more users have powerful processors, large storage spaces and fast network connections, more actors seek to coordinate these resources for common goals. Because of their unique decentralized nature, security in these systems is both critical and an interesting problem. How do you secure a dynamic system without central coordination? Good security on P2P systems must reflect the design goals of the system itself.

A P2P network treats every user as a peer. In file sharing protocols such as BT, each peer contributes to service performance by uploading files to other peers while downloading. This opens a channel for files stored in the user machine to be uploaded to other foreign peers. The potential security risks include:

### A. TCP ports issues:

Usually, P2P applications need the firewall to open a number of ports in order to function properly. Each open port in the firewall is a potential avenue that attackers might use to exploit the network. It is not a good idea to open a large number of ports in order to allow for P2P networks.

### B. Propagation of malicious code such as viruses:

As P2P networks facilitate file transfer and sharing, malicious code can exploit this channel to propagate to other peers. Trojan horses have been found over P2P networks. An example is W32/Inject-H, which contained an IRC backdoor Trojan that utilized P2P networks to propagate itself. The Trojan would open a backdoor in a user's Windows PC to allow a remote intruder access and control of the computer18. Theoretically speaking, sensitive and personal information stored in the infected computer could be copied to other machines on the P2P network.

### C. Risks of downloaded content:

When a file is downloaded using the P2P software, it is not possible to know who created the file or whether it is trustworthy.

### D. Vulnerability in P2P software:

Like any software, P2P software is vulnerable to bugs. As each peer is both a client and a server, it constantly receives requests from other peers, and if the server

component of the P2P software is buggy, it could introduce certain vulnerabilities to a user's machine.

In light of these security threats, appropriate security and preventive measures should be implemented to protect against any potential leakage of sensitive information and breaches of security. The following are best practices for organizations and end-users when considering the use of P2P technologies [3]:

A. To mitigate the risks associated with exposing TCP ports, the organization should review the need for P2P technologies in supporting their day-to-day business operations. If a P2P network is not required, security policies should be established to block off unnecessary port ranges across the network.

B. Users should be educated about proper use of P2P networks, as well as the dangers associated with file sharing.

C. In particular, a P2P network is not a recommended channel for the sharing of sensitive or personal information because communication links in P2P networks are not usually encrypted, and any content is at risk of sniffing by external parties.

D. A clear firewall policy should be defined to block network ports used by common P2P applications (such as the ports mentioned for Bit-Torrent in the previous section) so as to deny P2P network traffic from entering or leaving the internal network.

E. Security controls such as personal firewalls, anti-virus programs with latest virus definitions, the latest security patches and system administrative rights restrictions need to be implemented to avoid potential security breaches and system misuse in end-users/home networks.

F. For young people sharing files over the Internet, they must be educated on the dangers of downloading files from untrustworthy or suspicious sources.

G. If a P2P download is necessary, it is advisable to quit the P2P client application after completion of the download.

## 8. SECURING P2P NETWORKS:

### A. 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 [4].

### B. 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 [4]. It is not possible to rely solely on anonymous P2P to hide the file sharing application's use without using encryption. However, using encryption together with anonymous P2P would yield possibly the most secure P2P usage experience available today.

## REFERENCES:


1. Choon Hoong Ding, Sarana Nutanong, and Rajkumar Buyya, Peer-to-Peer Networks for Content Sharing, Grid Computing and Distributed Systems Laboratory, Department of Computer Science and Software Engineering, The University of Melbourne, Australia
2. Potharaju S R P Saradhi Mohmed Nazurudin Shaik Potharaju S R Aditya, An Innovative Approach to Content Search Across P2P Inter-Networks.
3. Stephanos androutsellis-theotokis and Diomidis spinellis, A Survey of Peer-to-Peer Content Distribution Technologies, Athens University of Economics and Business.
4. James Li, A Survey of Peer-to-Peer Network Security Issues.
5. R. Denneman, P2P searching methods, research issues, solutions and their comparison, Faculty of Management and Governance University of Twente, the Netherlands.
6. Qin Lv, Pei Cao, Edith Cohen, Kai Li and Scott Shenker, Search and Replication in Unstructured Peer-to-Peer Networks.
7. Erik Schierboom and Dennis Meffert, Searching in Peer-to-Peer Networks.