PEER-TO-PEER TRAFFIC IDENTFICATION

By

Sonia Riaz

(2005-NUST-MS-PhD-CSE-37)

A thesis submitted to the Department of Computer Engineering,

College of E & ME, National University of Sciences and Technology,

Rawalpindi in partial fulfillment of the requirements for the degree

of MS in Software Engineering

August 2008

ABSTRACT

The term peer-to-peer refers to sharing of computer resources through direct exchange between computers and peer-to-peer network is a network of personal computers which permits sharing of specific files and folders with everyone or with selected users. Sharing content and finding content are two key functions of peer-to peer systems.

Sharing remains the dominant P2P application on the internet, allowing users to easily contribute, search and obtain content so, peer-to-peer traffic is an important class of service. Accurate identification of this traffic is a challenging problem. Port and Payload based techniques are traditional methods of traffic identification. Many peer-to-peer applications use random ports so port based detection does not remain effective. Payload based analysis, though, gives accurate results but has limitations. It is required to pre-determine the signatures and match signatures to identify the traffic.

In order to overcome the drawbacks of the traditional identification methods, non-payload based method has been introduced that examines transport layer information and statistics to identify peer-to-peer traffic patterns. It does not deal with user payload and does not require predetermination of any kind as in Payload detection method. The proposed method uses heuristics to implement this technique. It rejects the traffic that show behaviors similar P2P according to applied heuristics but are actually non peer-to-peer.

This Work Is Dedicated

To

My Parents and My Teachers

Who Are Like Stars,

Guiding Me

Through

The Darkness of Life

ACKNOWLEDGEMENT

All praises be to ALMIGHTY ALLAH, The Most Merciful. HIS kind blessings made the completion of this task possible for me.

This thesis is the result of contribution from my family, faculty, friends and colleagues, whose assistance was there for me at each step.

I find no words to express my gratitude towards my beloved parents, who have made me what I am today.

Becoming a part of an institute like NUST opened new horizons for me to research and explore under the guidance of diligent teachers. I am deeply grateful to my thesis supervisor, Dr.Shoab A.Khan. His intellect and professionalism made the completion of this thesis possible. His wide knowledge and logical way of thinking have been of great value for me. His understanding, encouraging and personal guidance have provided a good basis for this work. I owe him lots of gratitude for his important support throughout this work.

With tremendous regards, I extend my sincerest thanks and higher degree of gratitude to the Head of Department Brig.Dr.Younas Javed for keeping the atmosphere of the department so affable for the carrying out research work. His guidance remained with me during my entire stay at the department.

My profound thanks are also due to all the faulty and staff members of Computer Engineering Department and the administration of College of E & ME.

Special gratitude goes to my parents for their prayers and patience. This work is dedicated to my parents who stayed with me till the completion of this work and provided me full support. I wish to extend my warmest thanks to all those who have helped me with my work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

1 INTRODUCTION

The term peer-to-peer refers to sharing of computer resources through direct exchange between computers and peer-to-peer network is a network of personal computers which permits sharing of specific files and folders with everyone or with selected users. Mainly, the peers are organized in such a way that each peer interacts directly with other peers without involvement of any central authority [1].

Sharing content and finding content are two key functions of peer-to peer systems. Sharing content involves direct transfer between peers and structured vs. unstructured placement of data. Finding content involves centralized and decentralized searching [1].

Such networks are useful for many purposes like sharing content files containing audio, video, data .Sharing remains the dominant P2P application on the internet, allowing users to easily contribute, search and obtain content. Napster, Gnutella, Kazaa and Freenet are popular peer to peer systems [1].

1.1 Peer-to-Peer Network Generations

The simplest peer-to-peer network consists of two computers connected at home sharing a printer, and the complex one consists of thousands of computers that exchange millions of files using Internet P2P software.

Internet P2P file-sharing evolved through three generations. Details of all the generations are as follows:

1.1.1 1st Generation

Initially, there was a central directory of files available for downloading in peer-to-peer network architecture. It was decided later on that that the company or individual controlling such a directory would be responsible for any illegal activities that were expected to occur because of this information, including copyright violations. The original MP3 file sharing system, Napster became the world's most popular Internet software application overnight. Napster offered chat rooms for millions of users and performed a new and exciting service [2].

1.1.2 2nd Generation

The P2P networks introduced after Napster used decentralized file lists.Napster usage transferred to Kazaa , Kazaa Lite software applications and the FastTrack network. FastTrack received more popularity than original Napster network. Some legal issues were associated with Kazaa, but various other systems, like eDonkey / Overnet, sustained the legacy of free P2P file sharing software[2] .

1.1.3 3rd Generation

Current P2P file-sharing networks are like previous P2P generation networks. They have followed the concept of optimality that existed previously. They have included the features that made such networks efficient and reliable. There are two types of current generation P2P networks:

Friend-to-friend peer-to-peer network

A friend-to-friend (or F2F) computer network is a type of peer-to-peer network in which users make direct connections only with people known to them. Users in this network

cannot find out if any one beyond their own circle of friends participates. Softwares like Turtle, WASTE, GNUnet and Freenet can be used to build F2F networks.

Anonymous peer-to-peer network

An anonymous P2P computer network is another type of peer-to-peer network in which users are anonymous or pseudonymous by default. The difference between regular and anonymous networks lies is in the techniques of routing of their respective architectures.Interest in anonymous P2P has increased in recent years [2].

1.2 Applications of P2P

A number of networking protocols such as SMTP (email) and NNTP (Usenet news) use peer-to-peer model. Currently, the role of peer-to-peer in file sharing networks is more popular. It enables the free and comparatively anonymous exchanging of local files between computers that are connected to the Internet.

Some file sharing networks like Napster, IRC @find and OpenNap have some functions like search queries that are based on a client-server structure. But the file sharing itself is done through peer-to-peer. On the other hand, Gnutella and Freenet are true peer-to-peer networks as the structure of their networks is based on peer-to-peer entirely.

Bandwidth is one of the crucial features of P2P networks. In case of peer-to-peer networks, the bandwidth available to the average user increases as more nodes are connected to the network. While, in client/server network, the available bandwidth between their computers and the central server is to be divided among users when they files on these networks. It results in slower data transfer rates when number of clients that are connected to server increases [3].

1.3 Working of Internet P2P Network

P2P on the Internet functions as a temporary network between computers running a common P2P software application e.g. Gnutella, Napster. It enables users to share files that are stored on local hard drives of their respective computers.

It is required to first download and install a P2P application on the computer .This application then connects to other computers on the P2P network through an IP address. A list of IP addresses is available on the P2P software provider's website. Once connected, the IP address of the connected computer is added to the list. Other users then connect to it when they go on-line. This process repeats itself as more users are added to the network.

Since there are no limitations besides what the user can define as to which files are to be shared, this has raised the concerns of major media publishers because it has permitted for the free exchange of copyrighted files as well [4].

1.4 Classification of P2P Architectures

Two general aspects of P2P architectures, according to which the P2P systems can be differentiated and categorized, are: the degree of centralization, and the network structure.

1.4.1 Degree of Centralization

P2P architectures can be classified according to "degree of centralization", i.e. to what extent they depend on one or more servers to interact between peers. Three categories are identified:

1.4.1.1 Purely decentralized Architecture

All nodes in this architecture act both as servers and clients and there is no central coordination of their activities. Nodes are termed as servants (SERVers+clieENTS). P2P architectures such as the original Gnutella architecture and Freenet are examples of such type [5].

1.4.1.2 Partially centralized Architecture

This architecture is based on purely decentralized systems. However, some of the nodes perform more important role than other nodes. They are termed as Supernodes. Supernodes act as central indexes for files shared by peers. It is important to note that there is no single point of failure in this architecture since supernodes are dynamically assigned and in case of failure or malicious attack, the network takes action for replacement of nodes. Systems such as Kazaa and Morpheus have partially centralized architectures [5].

1.4.1.3 Hybrid decentralized Architectures

There is a central server that interacts between peers by maintaining directories of shared files. These shared files are stored on respective PCs of registered users in the form of meta-data. The end-to-end interaction is between two peer clients. This interaction is facilitated by performing the lookups and identifying the nodes where the files are located. There is a single point of failure (the central server). This makes hybrid decentralized architectures vulnerable to technical failure. So, hybrid decentralized systems are not considered real P2P systems, as they follow the standard client-server model, and only the file transfer takes place between peers [5].

1.4.2 Network Structure

P2P systems comprise of highly dynamic networks of peers. Topology of these networks of peers is complex. Peers build an Overlay Network in which each peer maintains some point-to-point connection with some other peer. The topology of overlay network can be adhoc or have some structure. P2P systems can be differentiated by the degree to which overlay networks contain some structure or are created ad-hoc [5].

1.4.2.1 Unstructured Networks

In unstructured networks, the placement of data (files) is not related to the overlay topology. Random search is performed since it is not known that which nodes have the related files. Nodes are searched and inquired if they have any files that match the query. The way in which unstructured networks construct the overlay topology and distribute queries from node to node differentiates them from other networks. Queries are distributed widely to find the relevant files. Gnutella network is an example of unstructured network [5].

1.4.2.2 Structured networks

Structured systems introduced controlled overlay network topology and files (or pointers to them) are positioned at exactly specified locations. These systems provide a mapping between the file identifier and location in the form of a distributed routing table. Queries can be efficiently transferred to the node with the desired file using this mapping. Chord, CAN, PAST, Tapestry etc are examples of structured networks [5].

1.4.2.3 Loosely structured networks

Loosely structured networks have characteristics of both structured and unstructured networks. File locations are influenced by routing hints. Not all searches succeed as routing clues are not completely precise [5].

1.5 Characteristics of Peer-to-Peer Networks

Following are the major characteristics of peer-to-peer networks.

1.5.1 Ad-hoc Nature

Peers join and leave the system without any direct control. Therefore, the number and location of active peers as well as the network topology interconnecting them are highly dynamic. [6].

1.5.2 Reliability of Peers

Peers fail more often. The unreliability of peers proposes that fault tolerance should be important part of the peer-to peer protocols [6].

1.5.3 Rationality of Peers

Computers in a peer-to-peer system are owned and controlled by independent entities (peers). Peers may make decisions regarding whether to share data, leave the system, and forward queries. These decisions are not always made keeping in view the performance objectives of the system. This conflict of interest may cause danger to the performance of entire system. Therefore, peer rationality should be given importance in designing the peer-to-peer protocols [6].

1.6 Advantages of Peer-to-Peer Networks

1.6.1 Efficient use of Resources

An important objective in peer-to-peer networks is that all clients provide resources like bandwidth, storage space, and computing power. [7].

1.6.2 Distributed Nature of Peer-to-Peer Networks

Data is replicated over multiple peers. This distributed nature of peer-to-peer networks increases robustness in case of failures and in pure peer-to peer systems by enabling peers to find the data without relying on a centralized server. In the latter case, there is no single point of failure in the system[7].

1.6.3 Scalability

Resources, that belong to peers are aggregated to provide improved scalability. This reduces dependence on centralized servers[7].

1.6.4 Ease of Administration

Self-organization of nodes provides ease of administration. There is built-in fault tolerance, replication, and load balancing [7].

1.7 Challenges in Peer-to-Peer Systems

Peer-to-peer systems offer a number of advantages over conventional client-server systems such as scalability, fault tolerance and performance. However, these systems deal with some challenges. They are described as follows:

1.7.1 Security

Distributed mechanism in peer-to-peer systems brings additional challenges for security as compared to mechanism in client-server architecture. The set of active peers is dynamic in peer-to-peer systems so achievement of high level of security is more difficult than in non-peer-to-peer systems. Conventional security mechanisms to protect data and systems from intruders and attacks cannot act as shields for peer-to-peer system. Therefore, new security concepts are required that can protect peer-to-peer systems [8].

1.7.2 Reliability

A reliable system can be termed as a system that can be recovered when failure occurs. Factors like data replication, node failure detection and recovery and the availability of multiple paths to data are taken into account for reliability. Two concepts are involved in replication i.e. owner replication and path replication. In owner replication, data received from successful search is stored at the requester node only. In path replication, data received from successful search is stored in all nodes along the path from requester node to provider node [9]. Peer-to-Peer communities can also replicate and replace the data to achieve acceptable performance [10].

1.7.3 Flexibility

Autonomy of peers is one of the important aspects of peer-to-peer systems so that they can join and leave at their own. Peer-to-peer systems are featured by decentralized control, large scale and extreme dynamic nature of their operating environment. Concept of self-organization is required to be considered in building peer-to-peer systems in order to cope with scale and dynamism. In unstructured systems like Kazaa, queries are

forwarded only to supernodes which retain a list. This list contains file names of their connected peers. This mechanism prevents overloading all the peers of the system. In structured peer-to-peer systems, static identifiers are assigned to peers so the overlay network structure is determined by the selection of these identifiers. Hence, self-organization is prevented in structured peer-to-peer systems. [11]

1.8 Downside of peer-to-peer Networks

Some peer-to-peer programs share everything on the computer with anyone by default like medical information, copyrighted documentation, financial and other personal and corporate information. Viruses, Worms and Trojans are being distributed. Much of the peer-to-peer activity is automatic, and its use is unmonitored. Computers running this software will be busy exchanging files whenever the machine is turned on. Some of the most destructive viruses such as Swen, Fizzer, Lirva, and the Benjamin have been propagated through peer-to-peer applications on a massive level. Many peer-to-peer applications also contain spyware that allows monitoring of system activity, use system resources, and monitor internet surfing without user knowledge. Since the computers running the peer-to-peer programs are usually connected to a network, they can be used to spread malware, share private documents, or use file server for store-and-forward.

Worms propagating through peer-to-peer applications would be disastrous: it is probably the most serious threat posed by peer-to-peer. Peer-to-peer networks are composed by computers all running the same software. Peer-to-peer nodes tend to interconnect with many different nodes. Indeed a worm running on the peer-to-peer application would scan for other victims and would simply have to fetch the list of the victim's neighboring nodes and spread on. As Peer-to-peer programs often run on personal computers rather

than servers, it is thus more likely for an attacker to have access to sensitive files such as credit card numbers, passwords or address books. Peer-to-peer users often transfer illegal content (copyrighted music etc...) and may be less inclined to report an unusual behavior of the system. [11]

1.9 Network Traffic Detection

Networks are vulnerable to a range of problems, which may cause traffic condition changes, and thus may produce a negative performance impact on network applications as a result. Such problems include cyber attacks and power failures. Adverse effects created by traffic condition changes comprise traffic congestion and denial of service. Therefore, it is needed to observe and identify significant changes in traffic conditions. If the traffic conditions are not noticed, these can produce a consequence that may be uncontrollable as local disturbance quickly propagate to a larger scale, and become a critical network performance problem. Some known causes of traffic condition changes on a network are the computer and network attacks, viruses and worms. Yet, malicious activities are one of the many concerns of network management.

In detection, information is gathered from a computer or a network and analyzed possibly identify traffic, intrusions or malicious activities.

Traditionally there have been two detection methods:

- Signature based detection
- Anomaly Based detection

1.9.1 Signature based detection

Signature based detection involves searching for protocol specific string in the payload for identification of peer-to-peer traffic.

It has the following advantage:

- The signatures are easy to develop and comprehend if the network behavior is known.

Signature engines have the following disadvantages:

- They cannot be used if payload information is not available
- They cannot, in general, identify unknown classes of traffic.

Payload information is not always accessible for a number of reasons. Most applications encrypt their payload, thus making it impossible to read. Finally, examining the payload to classify traffic in real time is unreasonable due to its high overhead, especially if there is high utilization of network [12].

1.9.2 Anomaly Based detection

In Anomaly-Based Detection, computer intrusions and misuses are detected by monitoring system activity and classifying it as either normal or anomalous. The detection is based on heuristics or rules, rather than patterns or signatures, and detects any type of behavior that is different from normal system operation. The anomaly-based detection has the following advantage over signature-based detection:

- Any attack for which there is no signature can be detected if it falls outside the normal traffic behavior.

In Anomaly-Based Detection, there are high false alarm rates that are produced by incorrect or mistaken profiles of normal use. The observed traffic behavior violating the normal traffic behavior is classified as malicious traffic [12].

1.10 Motivation for Peer-to-peer traffic detection

Peer-to-peer applications have been gaining popularity over the last few years. Since they are typically used to share large files such as video/audio files or software, peer-to-peer flows amount to a significant portion of the network traffic.

Peer-to-peer traffic, therefore, is an important class of service .Accurate identification of this traffic is a very desirable feature and a challenging problem. There is a need for it because it is an important building block of many network management tasks such as flow prioritization, traffic shaping/policing, security and diagnostic monitoring. Monitoring traffic offers many security advantages like possibility to identify and block worms, scanning activities and Denial-of-service attacks. It also offers the possibility to provide different Quality of Service (QoS) to different flows [12].

The dramatic increase in use of P2P applications suggests that the P2P traffic has significant impact on underlying network. To ensure accurate performance for Web Traffic and real time applications, ISP's must have to apply some type of traffic Engineering (traffic shaping, priority policies...) and to do this efficiently, the ISP's must first be able to perform classification of internet traffic. Thus, this highlights the need for detection of peer-to-peer traffic.

1.11 Problem statement

The problem is to introduce new approach to peer-to-peer traffic detection that examines information in packet header, connection behavior and statistics to identify peer-to-peer traffic patterns. It does not examine user payload. The proposed method uses heuristics to implement this technique.

1.12 Research Objective

The objective of the research is to improve peer-to-peer traffic detection method. Previous techniques of port based detection do not work with dynamic port usage and payload based detection can be avoided by changing signatures or using encrypted data. So, aim is to introduce a methodology that is efficient and flexible enough to handle these flaws.

1.13 Thesis Organization

The thesis is organized as: Chapter 2 gives the review of techniques implemented in past for peer-to-peer traffic detection. Chapter 3 describes the methodology adopted for the proposed approaches and explains the detailed design of proposed approaches. Comparative analysis of the detection techniques and examination of results is performed in Chapter 5.Chapter 6 concludes the research and focuses on extended goals for future work.

LITERATURE REVIEW

2 LITERATURE REVIEW

Traffic on the Internet has increased enormously over the last few years, both in variety of applications and in terms of amount of traffic. Voice, video and other real-time applications have altered the way internet is used. This has motivated the need for research in traffic management on the Internet. There has been a lot of research in the area of network traffic classification and several different classifiers have been suggested. Identifying peer-to-peer traffic is an important part of this research.

Peer-to-peer (P2P) file sharing applications have significantly grown in popularity over the past few years and today comprise of important portion of the total traffic in many networks. These applications have increased in variety and have become sophisticated due to increased scalability, added functionality, improved search capability and download times. Within duration of six years, peer-to-peer applications have evolved from first, second, to third generation. One reason for the rapid development of these applications has been the need to avoid detection. The first generation [13] of peer-to-peer systems consisted of centralized systems like Napster. A centralized server was used for indexing of files. Due to this mechanism, it is comparatively easy to trace the server and block it. Also, peer-to-peer applications used renowned ports to transfer data. This methodology made identification of peer-to-peer traffic easy for the network operators. and as a result block the matching ports to discourage peer-to-peer traffic. The second generation [13] of peer-to-peer systems includes protocols like Gnutella [14]. Gnutella was totally distributed system where queries are transferred to neighboring nodes. Peers also used dynamically assigned ports to transfer data so that it was difficult to identify

peer-to-peer traffic. Third generation [13] peer-to-peer systems were quite sophisticated. These are hybrid systems that combine features of centralized as well as distributed systems. There is a concept of supernodes as in KaZaA or ultrapeers as in Gnutella2. The supernodes or ultrapeers have relatively more computing resources than other neighboring peers and are liable to handle indexing of files for peers. They often transmit data using randomly selected ports. Sometimes, they hide their traffic by using ports of other well-known applications. Moreover, a single large file can be downloaded in smaller pieces from numerous other peers at the same time. Besides, a few protocols like FastTrack have encrypted the application-layer data in the packets. These techniques make it harder to detect peer-to-peer traffic.

2.1 Related research

The standard approach to traffic classification relies on mapping applications to well-known port numbers and has been very successful in the past. Several peer-to-peer applications also have their default service port number, Gnutella [12] (6346, 6347), Kazaa [13], BitTorrent [17] (6881–6889) and so on. As a result, many research studies for peer-to-peer traffic use the default service port number identification methods in [18], [19] and [20]. However, some recent peer-to-peer applications, WinMX [18] and Winny [22], do not use a default service port number that would allow their services to be acknowledged. For these applications, service port number identification method does not work well. To avoid detection by this method, peer-to-peer applications started using dynamic port numbers and also started hiding themselves by using port numbers for commonly used protocols such as HTTP and FTP. Many recent studies confirm that port-based identification of network traffic is unsuccessful [23]. In general, most applications

use a known port for their communication. Hence, it is easy to classify traffic in an ideal, cooperative environment simply by a port number lookup method.

Payload-based analysis techniques have been proposed [24, 25, 26, 27] .In this approach, packet payloads are examined to find out whether they contain signatures of known applications. By using application-specific information, comparatively correct identification results can be obtained and this approach is used to authenticate other methods. However, signatures are changed whenever applications are evolved or new versions of existing applications are released. Therefore, the signature must be updated regularly. Also, when applications such as Skype encrypt their data, this approach cannot be functional. Studies show that these approaches work very well but use of plain-text ciphers and encryption decreased efficiency of this technique. These techniques only identify traffic for which signatures are available and are not able to identify any other traffic. Secondly, these techniques normally require increased processing and storage space.

Sen et al. [28] introduced an approach for identifying peer-to-peer traffic through application-layer signatures. They examine packet-level traces to identify these signatures and then use them to develop filters that can trace peer-to-peer traffic on network links. Their study analyzes TCP packets in the download phase of file transfer. Signatures in peer-to-peer applications are decomposed into predetermined pattern matches within a TCP payload [28].

Signature matching identification methods [29], [30] are effective when the applications exchange the specific strings in packets' payloads. This traffic identification method is generally applied for Intrusion Detection Systems (IDS) [31], [32] to deal with traffic

management. In this method, each packet needs to be examined and it requires huge computation power. In [33], the authors propose a signature matching identification method for peer-to-peer traffic and compare application level signature matching method with the default service port number identification method. In signature matching methods, the application signatures need to be updated with the release of new version of applications.

In [34], Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang have provided an efficient approach to identify traffic belonging to peer-to-peer applications through application level signatures. Examining packet-level traces results in identification of application level signatures. They have made use of the known signatures and online developed filters that can proficiently and appropriately track the peer-to-peer traffic even on high-speed network links.

The performance of the application-level identification approach is assessed using five popular peer-to-peer protocols. According to the analysis done, technique attains less than 5% false positives and false negative ratios in most cases. They have shown that their approach needs the examination of the only very first few packets to identify the peer-to-peer connection. Estimation of peer-to-peer traffic volumes is improved as compared to port based approaches using this technique. They have verified that sophisticated application layer signatures can be examined on high-speed links [34].

Packet payload based approaches work very well for Internet traffic including peer-to-peer traffic but these techniques also have negative aspects. Payload based techniques normally require increased processing and storage capacity. They do not work when transmissions are encrypted. Finally, these techniques only identify traffic for which

specific signatures are available and are unable to classify traffic when signatures are unknown.

The limitations of port-based and payload-based analysis motivated the use of transport layer statistics for identifying the traffic [35, 36]. These techniques rely on the fact that different applications show distinct behavior patterns when they communicate on network. These patterns are analyzed at three levels (i) the social, (ii) the functional and (iii) the application level. This approach has (a) no access to packet payload, (b) no knowledge of port numbers and (c) no additional information other than what current flow receivers provide.

To deal with the ineffectiveness of port-based classification, recent studies have used statistical classification techniques to assign flows to classes on basis of probability procedures, e.g., machine learning [37] or statistical clustering [35, 31]. In such approaches, flows are grouped in a preset number of clusters according to a set of discriminants. These discriminants usually include the average packet size of a flow, the average duration of flow and the inter-arrival times between packets. Studies have also examined how the details of exact timing and sequence of packet sizes can describe particular applications [38].

In [39], Alok Madhukar, and Carey Williamson focus on peer-to-peer network traffic measurement on the Internet. The study compares two methods to classify peer-to-peer applications: application-layer signatures and transport-layer analysis. The study uses empirical network traces collected from the University of Calgary Internet connection. The results showed that port-based Application signatures are accurate but may not work for legal or technical reason e.g. encryption. The Transport-layer heuristics offer a new

method that classifies the peer-to-peer traffic based on connection-level patterns. The results show that the transport-layer method can give significant information regarding aggregate peer-to-peer traffic.

In [40], Marcell Perényi, Trang Dinh Dang, András Gefferth, and Sándor Molnár have presented a new peer-to-peer traffic identification method. The method collects a set of rules derived from the general behavior of this traffic. The validation results show that the algorithm is able to identify the P2P traffic very effectively. The method was used to identify P2P traffic in Internet traffic traces. These traces were taken from one of the largest Internet providers in Hungary. They also presented a comprehensive traffic analysis study focusing on the most significant features like the performance of active users, the proportion of the P2P users and the total number of users and flow size. It was found that the profile of P2P traffic intensity on daily basis is less variable and shows a robust P2P user existence. They also demonstrated that packet-level statistics of P2P and non-P2P data flows are basically similar. However, there are some applications that generate data packets with typical size [40].

One possibility is to recognize particular features of the application traffic through machine learning. A machine learning approach called *clustering* is applied for classifying traffic [41]. Recent work by McGregor *et al.* [42] and Zander*et al.* [43] shows that cluster analysis using K-Means, DBSCAN and AutoClass algorithm.has the ability to classify internet traffic considering only transport layer characteristics.

Some non-clustering techniques also use transport layer statistics to classify internet traffic [44]. Roughan *et al.* used nearest neighbor and linear discriminate analysis [43]. The duration of connection and average packet size are used for classifying traffic into

four distinct classes. This approach has some limitations in that the analysis from duration of connection and average packet size may not be enough to classify all applications classes.

Supervised Machine-Learning has also been applied to classify network traffic by application. ‘Naïve Bayes estimator’ was developed to categorize traffic [44]. In this method, hand-classified network data is used, using it as input to a supervised Naive Bayes estimator. High level of correctness is achievable with this method.

Machine learning techniques usually involve model building and then classification. A model is first built using training data. This model is then applied as input to a classifier that then performs classification of a data set. Machine learning techniques can be categorized into unsupervised and supervised. McGregor *et al.* hypothesized the ability of using an unsupervised approach to group flows. This approach was based on connection-level (i.e., transport layer) statistics for traffic classification [45]. In this method, an EM algorithm [46] is used and McGregor *et al.* concludes that this approach is promising. In [47] and [48], Zander *et al.* extend this work by using an EM algorithm called Auto Class [49] and finds the best possible set of attributes to be used for building the classification model. Some supervised machine learning techniques, such as [49], also use connection-level statistics for traffic classification.

The unsupervised machine learning approach is based on a classifier built from clusters. These clusters are found and marked in a training set of data. Once the classifier has been built, the classification process consists of the classifier that calculates which cluster a connection is closest to, and label from that cluster is used to identify that connection.

A method based on the support vector machine (SVM) was proposed to perform the P2P traffic identification. Ideally, the proposed method has the ability of identifying all P2P traffics only if the training set includes all features of P2P and non-P2P traffics. Since the feature extraction algorithm describes the traffic features properly, the proposed method reaches a higher accuracy of identification than the existing methods [12].

Another method was proposed to identify pure P2P traffic, Winny, for evaluating its basic characteristics. Using the decoy node, The IP address and service port of Winny peers is identified. This IP and service port number can be selected in the traffic log of the back-bone.Traffic log is collected from the other stub networks which have users for the Winny application. The identification method depends on the access number of accesses of the decoy peers by peers in the Winny networks and the number of users in the stub network.

The introduced identification method can perform better by improving the access patterns among the peers. This identification method is effective for pure P2P applications since the technique depends on the basic relationships among client/server computing in the Internet applications [50].

A set of tests are proposed for identifying masqueraded peer-to-peer file-sharing based on traffic summaries (flows). The applied approach is based on the hypothesis that these application have certain observable behaviors that can be differentiated without relying on deep packet examination. Tests are actually developed for these behaviors that, when integrated, provide an accurate method for identifying the masqueraded services without relying on payload or port number. The approach is tested by demonstrating that the proposed integrated detection mechanism can identify BitTorrent with a 72% true

positive rate and virtually no observed false positives in control services (FTP-Data, HTTP, and SMTP)[51].

A new approach called flow-based identification has been proposed. In this approach, specific characteristics of P2P flows are retrieved statistically and P2P flows are identified. A streaming media (both unicast and P2P) traffic identifier is developed using this approach and there is a controller for identification of flows contain streaming media. For the traffic identifier, proposed methods identify streaming media traffic on a Flow by Flow Basis, using Port, Deep Packet Inspection and Flow information. For the controller part, a control model is implemented to update and manage new signatures for both Deep Packet Inspection and Flows. Accuracy of these methods is analyzed on a real world network that heavily uses streaming media applications [52].

The method uses only packet header information and is based on characteristics of P2P nodes and super nodes rather than protocol specific information, thus it makes the system protocol independent. The system works in two stages. In first stage, the unfiltered transport layer traffic data is applied as input to the prefilter. The prefilter produces the P2P traffic data .This filtered data set is then fed into the second stage where heuristics are applied to identify P2P super nodes and regular nodes. This method of P2P traffic identification has been proven to be useful as it identifies P2P flows with 95% accuracy and it uses only transport level data and behavioral characteristics to identify P2P traffic. As all super nodes are also P2P nodes, it is logical to apply the distinguishing heuristics to a data set that only contains candidates for either node type [53].

A method is proposed to identify the P2P traffic based on machine learning. The uniqueness of the proposed method is that it uses only the size of packets that are

exchanged between IPs within seconds. The ratio between the upload and download traffic volume of several P2P applications is examined and a characteristic library is constructed. The unknown network traffic can be recognized online using this library. The proposed method has distinguished features like fast computation, high identification accuracy, and resource-saving capability. Finally, experiment results show the satisfactory performance of the proposed method [54].

Research was conducted to focus the problem of traffic classification in the network core. Classification at the core is challenging as some information about the flows is available. In this research, the problem of classifying network traffic when only one direction of network flows are observed, is considered. To address this problem, a clustering-based machine learning framework is developed for classifying network traffic. It uses only unidirectional flow statistics. This classification framework is evaluated using a set of full-payload packet traces. The results show that, in general, traffic classification using only unidirectional statistics is feasible. It shows accuracies of 95% in terms of flows and 80% in terms of bytes [55].

An algorithm has been developed during a master's thesis by Lukas Hammerle [] for tracking population of P2P hosts. Its name is Peer Tracker.75-88% of the hosts were identified as P2P peers by Peer Tracker. Later on verification of the peer-to-peer host identification method was performed. Polling algorithms have been developed for eDonkey, Gnutella, Overnet and FastTrack networks. Polling P2P hosts is a difficult task because the host uses firewalls and intrusion detection systems. The verification algorithm works with an accuracy of at least 75% for the examined P2P networks.

Around 50% of all identified P2P hosts are found to accept no TCP connections from outside. Many significant findings about the P2P traffic in the SWITCH network have been discovered. In case of SWITCH network there is about 30% more outgoing P2P traffic than incoming traffic. It is also concluded that there are only a few peers that are responsible for large amount of P2P traffic. In the switch network, Less than 4% of the verified hosts are responsible for more than 30% of the total verified P2P traffic [56].

METHODOLOGY

3 METHODOLOGY

This chapter provides a detailed description of monitoring, analyzing and detection processes involved in peer-to-peer traffic identification.

3.1 System Parameters

System parameters are described below.

3.1.1 Hardware Parameters

Hardware parameters are as follows:

- Pentium IV
- 256 MB RAM
- 2.5 GHz processor speed.

3.1.2 Software Parameters

Software parameters are as follows:

- Windows XP
- Visual Studio 2007
- Winpcap 4.0 installer
- Ethereal Setup
- Peer-to-peer applications

3.2 System Methodology

System methodology consists of monitoring, analysis and detection phases. These phases are described in detail:

3.2.1 Monitoring Process

Monitoring process includes capturing of internet packets. In the initial step, packets of peer-to-peer traffic are captured through ethereal software. Ethereal is a packet capturing tool that lets the user see all the traffic on the network. Captured Packets are saved in ethereal files. These files are analyzed in analysis phase.

The tool developed for identification of peer-to-peer traffic is called “peer-to-peer traffic detector” .Internet packets are captured through this tool to perform online detection. It uses winpcap library for capturing internet packets. Winpcap is an API that provides built-in functions and routines to perform the capturing process. Five tuples of packets are received at realtime. These five tuples include the Source IP, Source Port, Destination IP, Destination Port, and the protocol. The packets are received for an interval of. 30 minutes and then are processed online for detection. After processing the flow, the program is able to process the next flow and so on.

3.2.2 Data Analysis

Ethereal packet files are examined to identify the characteristics of peer-to-peer traffic. First, analysis is performed to determine ports that are specific to peer-to-peer applications. Further study is done to find out the strings in packet payloads that uniquely identify P2P traffic belonging to specific P2P application from other applications. These strings are called “signatures”. Heuristics described in this chapter are also based on peer-to-peer traffic behavior examined during this analysis.

Peer-to-peer traffic detector implements three methods to identify peer-to-peer traffic. Identification process utilizes the data analyzed through ethereal files.

Description of the three methods and the information collected from analysed data is as follows:

3.2.2.1 Port based Identification

Peer-to-peer applications have default ports on which they function. When these applications run, they use these ports for communication. To perform port-based analysis, network traffic is observed and checked whether there are connection records using these ports. If a match is found, it indicates a peer-to-peer activity. Known port numbers of major peer-to-peer applications are listed below:

Table 3-1: P2P Applications and Known Ports

P2P Applications	Known P2P Ports
EDonkey (emule, xMule)	2323, 3306, 4242,4500,4501, 4661-4674, 4677, 4 678, 7778
FastTrack	1214, 1215, 1331
KaZaA	1337, 1683, 4329
BitTorrent	6881-6889
Gnutella	6346, 6347
MP2P	41170, 10240-20480, 2231
DirectConnect(DC++)	411, 412, 1364-1383, 4702,4703,4662
ShareShare	6399, 6388, 6733, 6777
Freenet	19114, 8081
Napster	5555, 6666, 6677, 6688, 6699-6701,6257
SoulSeek	2234, 5534

Blubster	41170
Morpheus	6346/6347 TCP/UDP
BearShare	6346 TCP/UDP
Limewire	6346/6347

3.2.2.2 Signature based Identification

Most protocols contain a protocol specific string in the payload that can be used for identification. These strings are identified through pattern matching in the packet payload.

Distinctive bit strings are derived empirically by monitoring both tcp and udp traffic using Ethereal. Following protocol signatures have been used for Signature based identification.

- ***Gnutella***

The Gnutella protocol uses TCP to set up a highly interconnected hub network topology. When the TCP connection is established between two Gnutella nodes, a handshaking phase must be accomplished. The Gnutella handshaking process consists of three header blocks. The node that started the connection sends a preliminary header block, as below:

GNUTELLA CONNECT/0.6

User-Agent: Shareaza 2.2.5.0

Listen-IP: 202.61.58.166:6346

Remote-IP: 75.109.29.139

Accept: application/x-gnutella2

Accept-Encoding: deflate

X-Ultrapeer: False

X-Ultrapeer-Needed: True

The receiver then responds by sending its own header block as below:

GNUTELLA/0.6 200 OK

User-Agent: Shareaza 2.2.5.0

Listen-IP: 217.132.242.170:6346

Remote-IP: 202.61.58.166

Accept: application/x-gnutella2

Content-Type: application/x-gnutella2

Accept-Encoding: deflate

Content-Encoding: deflate

X-Ultrapeer: True

X-Ultrapeer-Needed: True

Finally, the initiator accepts the receiver's header block, and gives any final information as below:

GNUTELLA/0.6 200 OK

Accept: application/x-gnutella2

Content-Type: application/x-gnutella2

Accept-Encoding: deflate

Content-Encoding: deflate

Gnutella UDP messages start with ‘GND’.

For identification of Gnutella application, the simple signature strings "GNUTELLA" and “GND” are used.

- ***Edonkey***

Edonkey is a file-downloading protocol. Signature for edonkey is “0xc319010000” and “0xc53f010000”.

- ***Bittorent***

BitTorrent is a file-downloading protocol. The BitTorrent handshake message contains the string "BitTorrent protocol" in the beginning of the message.

BitTorent header block is described as follows:

BitTorrent protocol.....V-S

.....s.e.H.I.C.N.... X.f.BitTorrent

protocolex..... V-S.....exbc.. LORD....

Based on this observation, the simple signature string "BitTorrent" is used for identifying BitTorrent traffic.

- **Ares**

Payload of Ares peer-to-peer application uses “GET hash” and “Get sha1:” These strings are used for identification of Ares.

3.2.2.3 Non Payload based Identification

Two main heuristics (from [57]) are applied that examine the behavior of the network traffic on the basis of IP addresses. First heuristic is named as “TCP/UDP Pair” and second, “IP/Port pairs”.

- ***TCP/UDP IP Pairs***

This heuristic state:

“If a source-destination IP pair uses both TCP and UDP transport protocols concurrently then flows between this pair will belong to P2P provided ports of some specific applications are excluded”.

This heuristic identifies source-destination IP pairs that use both TCP and UDP transport protocols. EDonkey, Gnutella, Bittorent use both TCP and UDP as transport protocols. UDP is used by control traffic, queries and query-replies generally, and TCP is used by actual data transfers. Only a few applications use both TCP and UDP transport protocols such as: DNS, NETBIOS, IRC, gaming and streaming, which use a small set of port numbers such as 135, 137, 139, 445, 53, 3531, etc.

The following table shows all such applications with their well-known ports:

Table 3-2: Excluded ports for TCP/UDP IP Pair Heuristic

Applications	**Ports**
NETBIOS	135,137,139,445
NTP	123
DNS	53
ISAKMP	500
Streaming	554,7070,1755, 6970,5000,5001
IRC	7000, 7514, 6667
P2Pnetworking.exe	3531
Gaming	6112, 6868, 6899

Flow of heuristic is as follows:

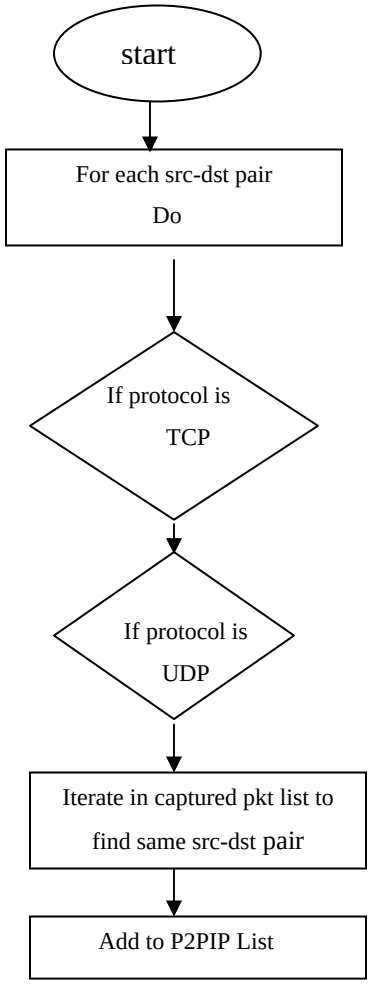

Figure 3-1: TCP/UDP IP Pair Heuristic

- ***IP/Port pairs***

This heuristic states:

"For the destination {IP, port} pair of host A, The number of distinct IPs connected to it are equal to the number of distinct ports used to connect to it".

When a P2P host initiates either a TCP or a UDP connection to new host, the destination port is the broadcasted listening port of the new host, and the source port is a temporary

random port selected by the client. So, for the broadcasted destination {IP, port} pair of the new host, the number of distinct IPs connected to it should be equal to the number of distinct ports used to connect to it to meet the conditions for {IP, Port} heuristic. If this condition is satisfied then traffic will be of P2P packets.

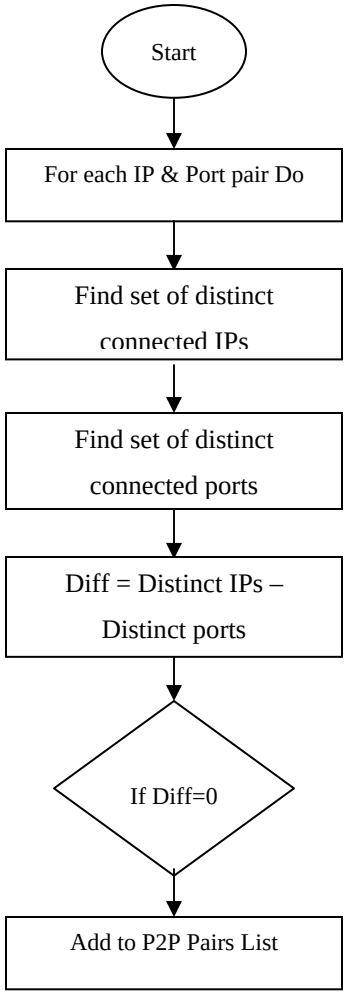

Figure3-2: IP/Port pairs heuristic

- ***Mailserver***

Connection behavior of E-mail protocols such as Simple Mail Transfer Protocol (SMTP) or Post Office Protocol (POP) resembles {IP, port} heuristic so they contribute false positives. All flows where one of the port numbers is equal to 25 (SMTP), 110 (POP) or

113 (authentication service commonly used by mail servers) are examined. For identification of this pattern, the set of destination port numbers are observed for each IP provided there exists a source pair {IP, 25}. If port numbers in source and destination pair match, this IP is considered a mail server and all of its flows are classified as nonP2P. The following flow illustrates the heuristic behavior

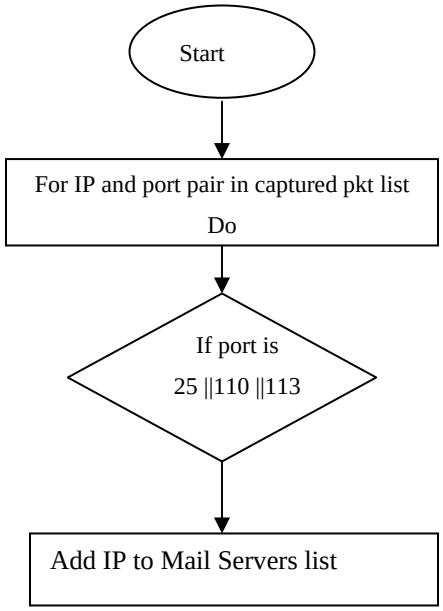

Figure 3-3: Mail Server

- ***DNS***

The Domain Name Server protocol runs on both TCP and UDP port 53. Its connection patterns fulfill conditions for {IP, Port} pair heuristic. DNS pairs are easier to identify since most DNS source and destination ports are 53.

Instead of limiting this heuristic to DNS, This heuristic is applied to all flows and

pairs where one of the ports is less than 501.It facilitates the removal of other false positives in commonly used ports (as 25), and particularly those caused by a service that runs on port 500.

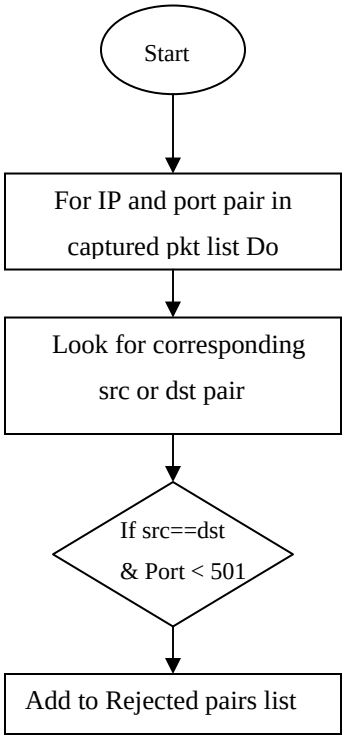

Figure 3-4: DNS

- ***Malware***

Malware means worm traffic as ‘MyDoom’ on ports 3127, 3128, or ‘Beagle’ on port 2745 and port or address space scans, which appear often in backbone traces.

Malware traffic satisfies the condition for {IP, port} heuristic, so all the pairs would be accepted as P2P pairs. So, all pairs satisfying this heuristic are inserted in a list of rejected pairs.

Table 3-3: Malware Ports Set

Malware Ports Set	3127, 3128, 1433, 1434, 3531, 1080, 10080, 17300, 6129, 27015, 27016, 901, 2745

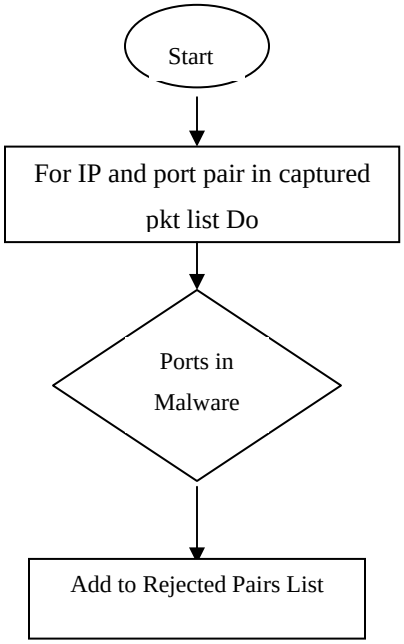

Figure 3-5: Malware

3.2.3 Detection Process

Peer-to-peer traffic detector detects peer-to-peer traffic using the three identification methods. Detection process is performed online.

For the port-based identification, the tool captures the packets of ports specific to P2P applications and displays the corresponding source ip, destination ip, source port, destination port and protocol.

For the signature or Content based identification, it inspects payload of packets, gets the signatures and compare them with the pre-determined signatures. The signatures normally appear at the start of the connection. If there is a match, it identifies the P2P protocol and displays the corresponding flow. Corresponding source IP, source port, destination IP, destination port and the protocol are displayed.

The tool checks for TCP/UDP pair heuristic and IP/Port pair heuristic and displays the corresponding IPs and ports. All IPs and ports are checked for false positives using DNS, Mail, and malware. The IPs identified by DNS, Mail, and malware are inserted in a list of rejected pairs and displayed.

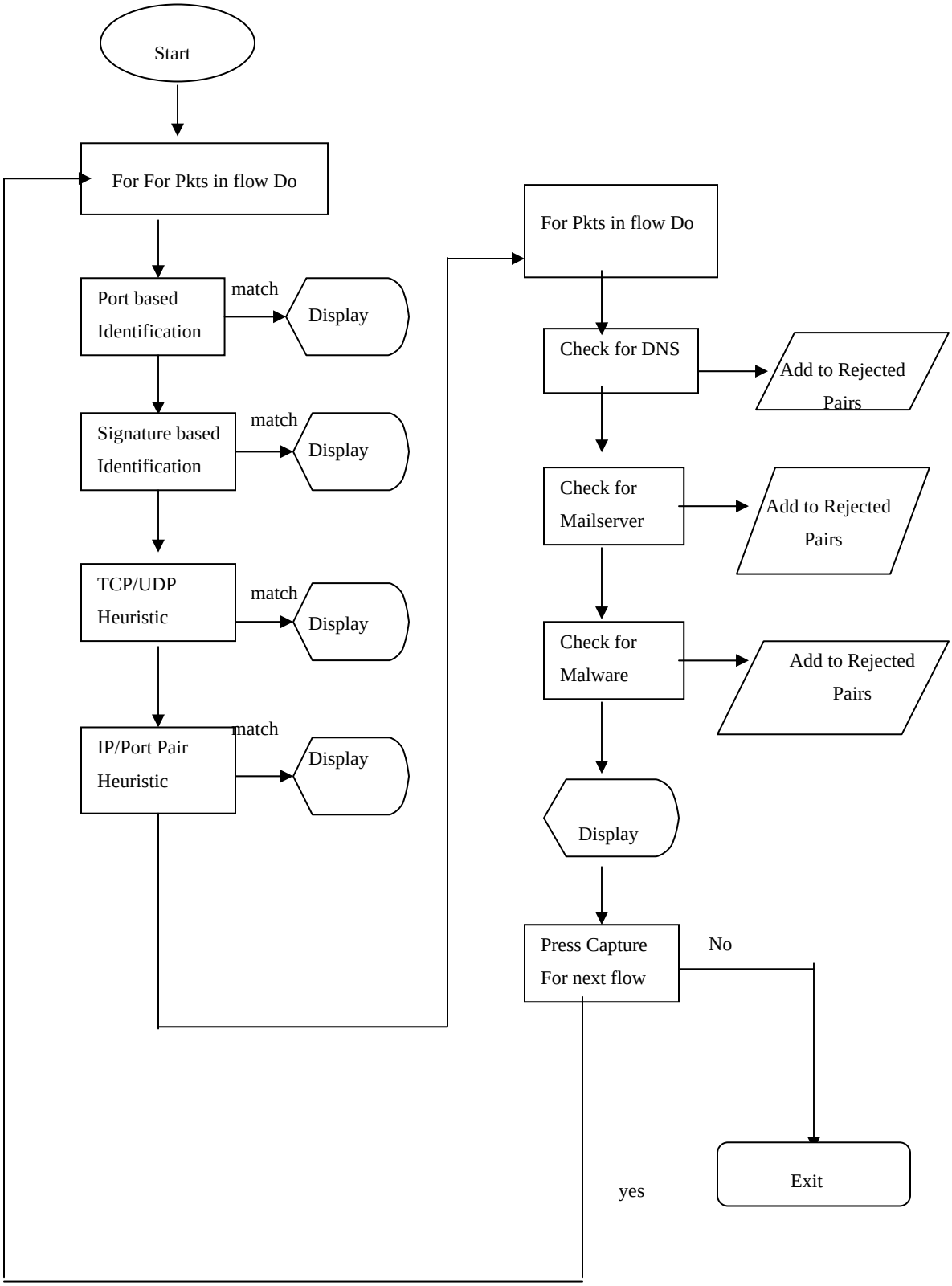

Figure 3-6: System Diagram

RESULTS AND ANALYSIS

4 RESULTS AND ANALYSIS

Port based, Payload based and heuristic based detection techniques are implemented in the current research. Following analysis provides detailed comparison of three techniques.

4.1 Comparative Analysis of peer-to-peer traffic detection techniques

4.1.1 Port based detection

When applications use ports that are known, this method proves to be an accurate and efficient way for identification as it is easy to classify traffic by port number look up methodology. On the other hand, when applications use random ports to disguise their traffic, this methods fails to detect the peer-to-peer traffic. e.g. the peer-to-peer software application Shareazae uses default port 6346/6347.Port based technique identifies the traffic through this port as peer-to-peer traffic but when port number is changed, it can not identify the traffic as peer-to-peer.

Case I: Peer-to-peer applications use well-known ports

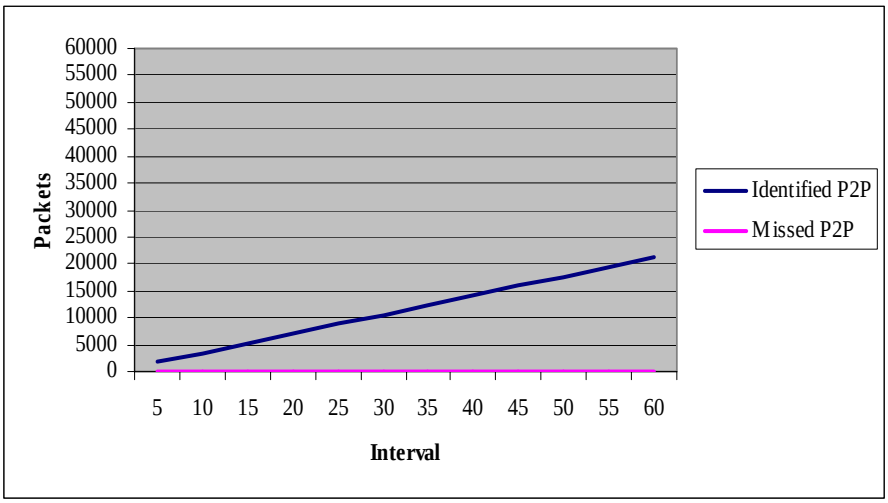

Figure 4-1: Identified and missed P2P when well-known ports are used

In this case, all traffic belonging to P2P is identified because of applications using unknown ports.

Case II: Some Peer-to-peer applications use random ports

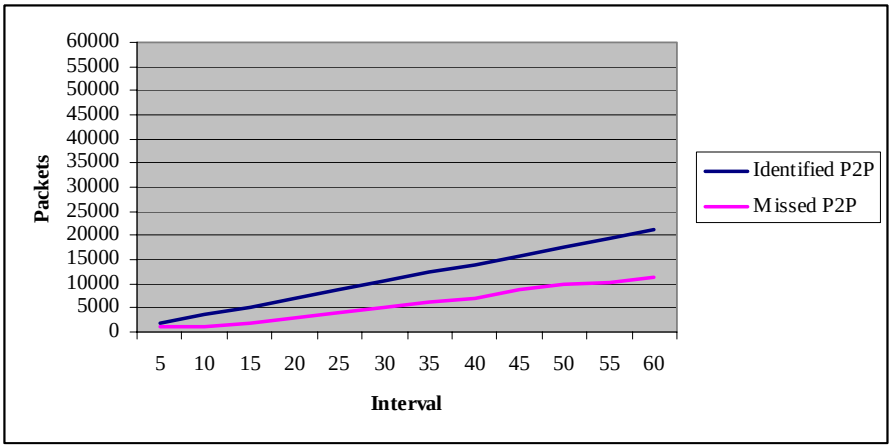

Figure 4-2: Identified and missed P2P when arbitrary ports are used

In this case, traffic belonging to peer-to-peer applications that use random ports is missed.

On the basis of graphical representation, percentage of missed and identified P2P is as follows:

Table 4-1: Percentage of Identified and missed P2P

Port-based Detection method	**Identified P2P**	**Missed P2P**
Case I: Well-known ports	97.6%	2.4%
Case II: Well-known and Arbitrary ports	65.27%	34.70%

4.1.2 Payload based detection

This technique uses specific strings in the payloads to perform identification. Peer-to-peer traffic detector detects the protocol specific signatures from the packet payloads. This technique shows accurate results when signatures are known whereas it cannot identify the traffic for those peer-to-peer applications whose signatures are not predetermined. In the current research, signatures of gnutella, bittorent and eDonkey protocols are identified and fed into the system. So, all peer-to-peer applications using these signatures are accurately detected as P2P and the rest remain unidentified.

Following graphical representations show rate of missed and identified P2P considering two cases:

Case I: Signatures for Peer-to-peer applications are predetermined

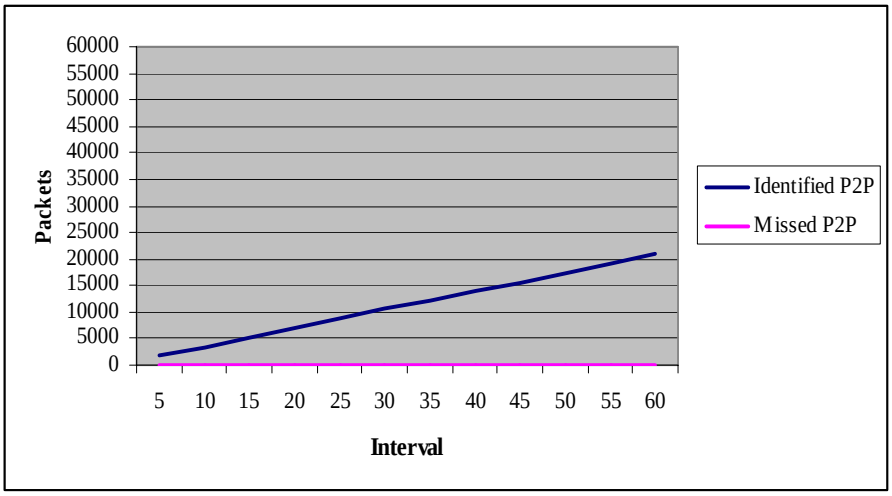

Figure 4-3: Identified and missed P2P when signatures are predetermined

In this case, all traffic belonging to P2P is identified because applications have predefined signatures.

Case II: Signatures for Some Peer-to-peer applications are unknown

In this case, traffic belonging to peer-to-peer applications whose signatures are unknown, is missed by the detection process.

Graphical representation of rate of missed and identified P2P and percentage this rate is described as follows:

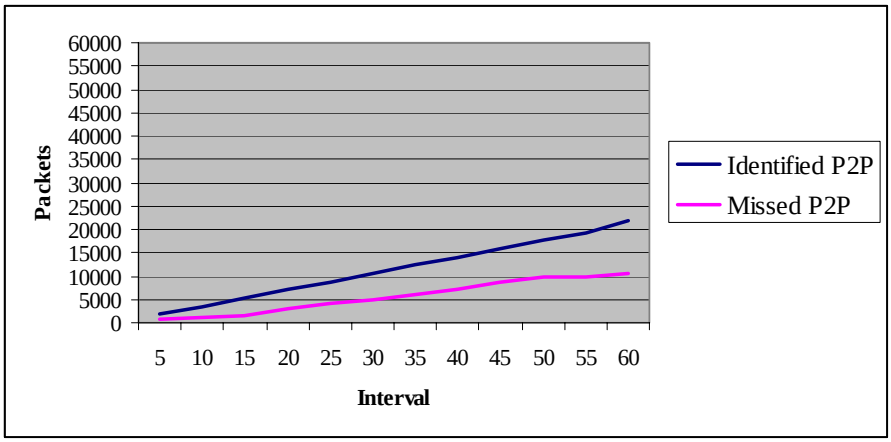

Figure 4-4: Identified and missed P2P when some signatures are unknown

Table 4-2: Percentage of Identified and missed P2P

Payload-based Detection method	**Identified P2P**	**Missed P2P**
Case I: Known Signatures	96.3%	2.9%
Case II: Known and Unknown Signatures	67.65%	32.34%

4.1.3 Heuristic based detection

It evidently differs from the payload based detection technique as it does not examine packet payloads. Payload based method cannot be used if payload information is not available.i.e.if encryption techniques are applied on the payload and it cannot identify

unknown classes of traffic as well.

Moreover, examining the payload in real time is not feasible due to its high overhead and computational complexity whereas in case of non-payload based methodology processing overhead is much lesser. Following graphical representation shows the rate of processing delay in the proposed methodology.

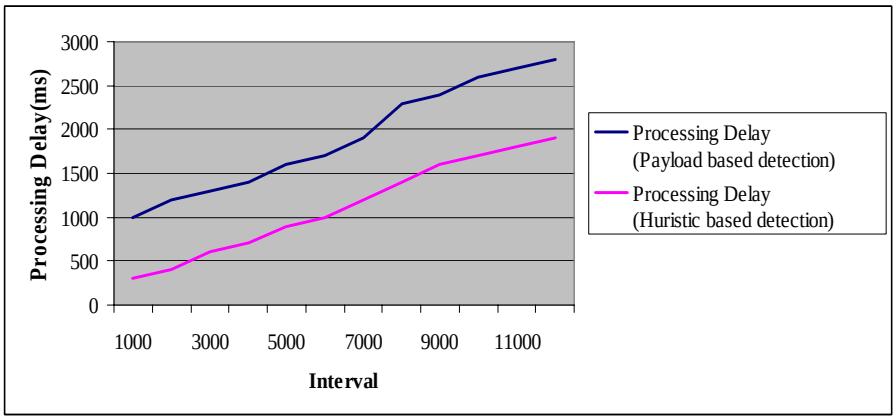

Figure 4-5: Comparison of Processing delay in Payload and Heuristic analysis

Packet data of any traffic is not public, privacy issues are associated with it. So, data inspection in payload analysis causes privacy and legal alarms. Non payload methodology here provides good solution to this problem by not examining the data rather only connection patterns of peer-to-peer traffic.

The new proposed methodology based on use of heuristics overcomes the limitations faced by port based and payload based methodologies.

Following graphical representations show that identification rate for peer-to-peer traffic by Heuristic based methodology is more as compared to the other techniques.

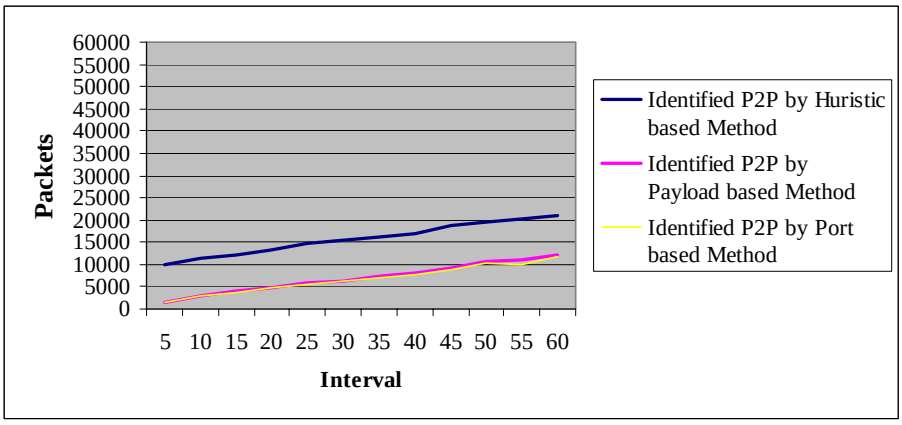

Figure 4-6: Comparison of detection techniques

When a P2P application uses arbitrary port, its traffic is missed by port based technique but it is detected by the new methodology.

Traffic belonging to P2P applications with unknown signatures is missed by payload methodology and detected by the new methodology. Processing time in this case, is much lesser than that in payload analysis as this technique does not involve payload inspection.

Inspection of payloads in payload analysis is not feasible as privacy issues are involved so non-payload based method provides a real good solution for this problem. It can detect traffic as P2P and non P2P when traffic is encrypted as well.

Payload method can detect the existence of protocols when information about them is already known or previously determined. The proposed methodology of non-payload based detection does not require any previous knowledge so can determine unknown peer-to-peer traffic.

Following table shows the results drawn from the graphical representations.

Table 4-3: Percentages of P2P Identified by detection techniques

Peer-to-peer traffic Identification Methods	**Identified P2P**
Port-based detection	47.5%
Payload-based detection	50%
Heuristic-based detection	77%

Port-based method detected 47.5% of P2P traffic correctly, Payload-based method detected 50% and Heuristic-based method detected 77%.

4.2 Analysis of Heuristic-based methodology

For detailed analysis of the proposed methodology, packets from different P2P applications are captured. Two popular peer-to-peer applications: Emule and Shareaze are included in them. These applications are installed and peer-to-peer traffic detector is run for identification techniques to work on captured packets**.**

4.2.1 Known Peer-to-peer applications

Shareaza peer-to-peer application is one of the known peer-to-peer softwares for searching and sharing files. It establishes connection with Gnutella2 network and uses port 6346/6347 and 'Gnutella' protocol. Searching for some keywords is started in Shareaza and downloading is also performed during search process. Peer-to-Peer traffic

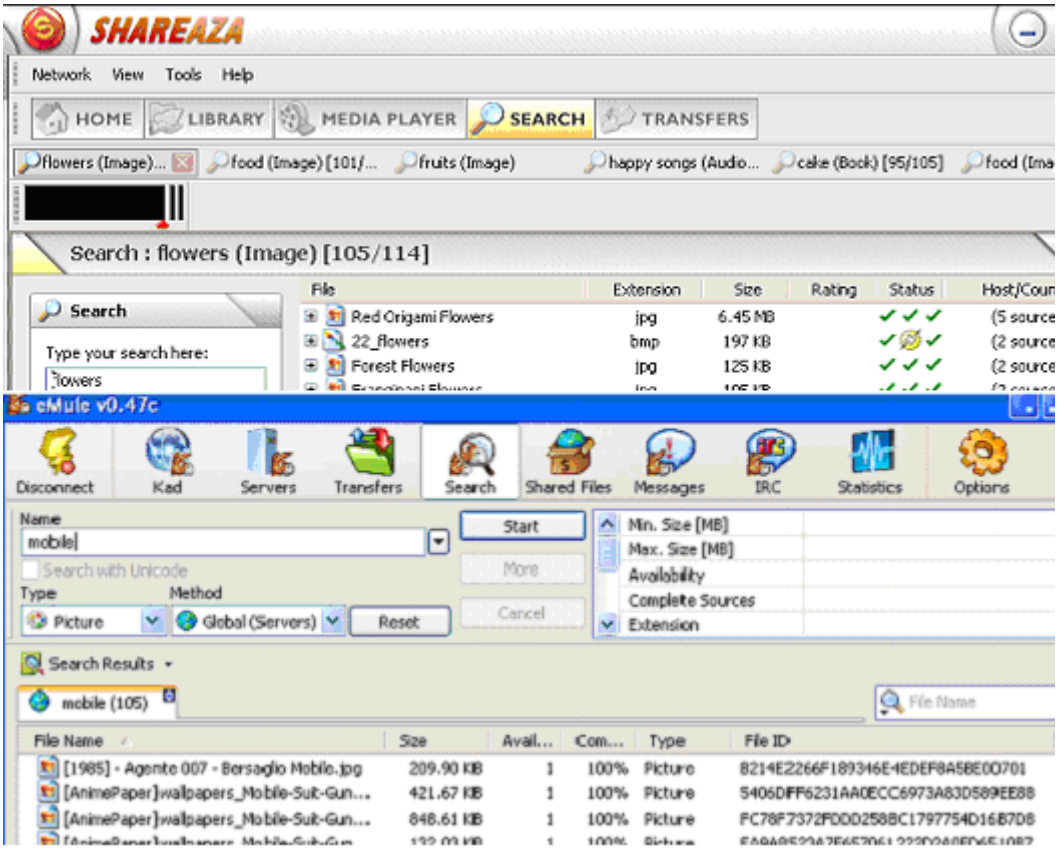

Figure 4-7: Peer-to-peer applications

detector is run for Shareaza application for every five minute interval. Port based identification method captures the packets at ports specific to Shareaza i.e. 6346/6347.Payload based identification method inspects the packet payload and allows only those packets in that have ‘Gnutella’ and ‘GND’ signature.

Emule peer-to-peer application connects at IP: 64.34.193.218, e2DK server. Port based identification method captures the packets of port number 4362 specific to Emule.Payload based identification method inspects the packet payload and allows only those packets in that have specific bit pattern.

Tcp/Udp and IP/Port pair heuristics capture IPs that the three applications use, on the basis of stated rules. Important findings are highlighted below.

- IPs missed by port based and signature based identification are captured by TCP/UDP heuristic and IP-Port pair heuristic.
- False positives are IPs that show behavior of IP/Port heuristic but do not actually belong to peer-to-peer traffic. In the current research, DNS and malware traffic IPs are false positives. They are detected by peer-to-peer traffic detector and added in the list of 'Rejected pairs'.
- True negatives are IPs that do not belong to peer-to-peer traffic. They are detected during the identification process.

Rate of detected P2P IPs, positives and true negatives is depicted through following graphical representation and table.

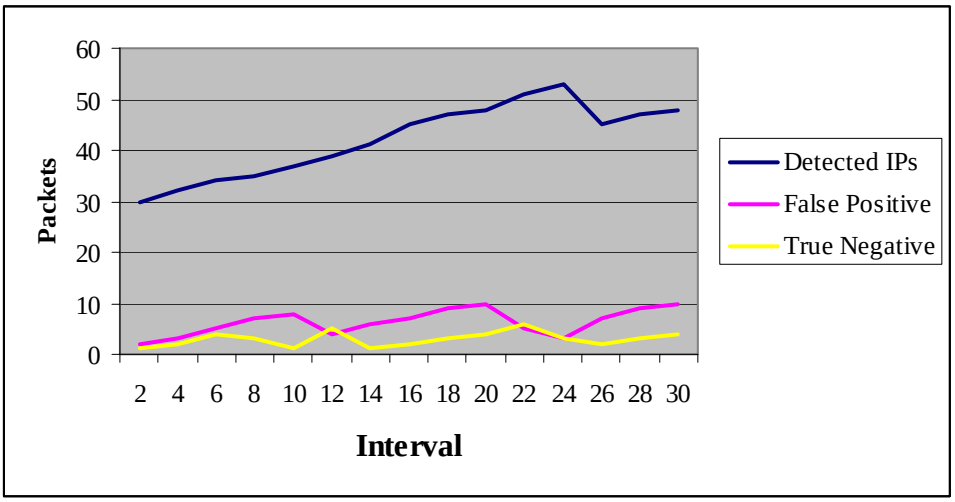

Figure 4-8: Analysis Heuristic based detection technique

Table 4-4: Percentage of detection by Heuristic based detection

Heuristic-based detection	
Detected P2P IPs	82%
False Positives	12%
True Negatives	6%

4.2.2 Unknown Peer-to-peer application

This test is performed to verify the efficiency of Heuristic based approach. Morpheus is peer-to-peer software. It uses random ports during communication. Its protocol is not analyzed so signatures are not predetermined.

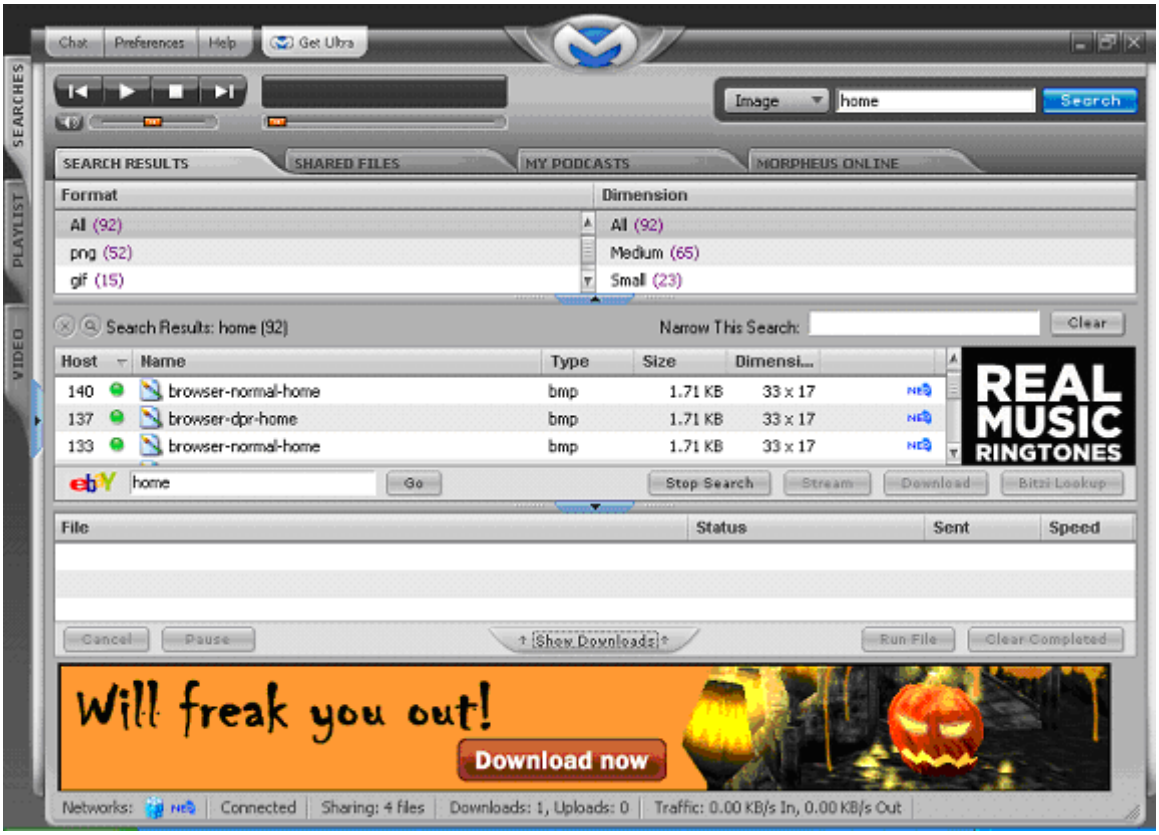

Figure 4-9:Morpheus peer-to-peer application

Searching and downloading of files is performed in Morpheus software. Peer-to-peer traffic detector runs and detection process starts.

Following findings are obtained:

- Port based method fails to detect the Morpheus application as it used arbitrary ports.
- Payload based detection method also fails as signatures for the application were not predetermined.
- Heuristic based detection methodology enlists the TCP/UDP and UP/Port pairs based on the stated rules. These IPs are specific to the morpheus application that it uses at connection establishment and during searching and downloading.

Hence, the proposed methodology overcome the limitations of previous techniques.

CONCLUSION

5 CONCLUSION

5.1 Overview

Peer-to-peer traffic is a growing component of internet traffic. It has been gaining popularity specifically because peer-to-peer applications are used for file-sharing over internet. Current research focused on monitoring and detection of this traffic using different identification techniques. Detailed analysis for the results is performed .Following description provides conclusion based on findings explored during the research.

5.2 Limitations of Detection techniques

Port based identification method uses default or standard port numbers for detecting peer-to-peer traffic. Port matching is very simple in practice, but its limitations are obvious. Most peer-to-peer applications allow users to change the default port numbers by manually selecting whatever port(s) they like. Additionally, many newer peer-to-peer applications are more inclined to use random ports, thus making the ports unpredictable. This issues makes port based analysis less effective.

Using the Signature or Payload based identification, the result is normally more accurate, but it still has some shortcomings. Peer-to-peer applications are evolving continuously, and therefore signatures can change. Static signature based matching requires new signatures to be effective when these changes occur. Signature-based identification means that the product should read and process all network traffic, which brings up the issue of how to maintain network stability in a large network. The product may burden network equipment heavily or even cause network failures. No payload or encrypted data

means no detection at all. Moreover, there is a great processing overhead involved in this method of identification.

Non payload based identification method examines the packet header to detect peer-to-peer flows and does not in any way examines user payload. It uses heuristics to apply this technique. The implemented peer-to-peer traffic detector program gives much better results. It can identify peer-to-peer flows using randomize port numbers that port based method fails to detect. Signature based method can only detect traffic when signatures are known while the proposed new methodology does not require such previous knowledge for detection. Through peer-to-peer traffic detector, it is possible to detect known and unknown peer-to-peer applications. Moreover, it rejects the IPs that show behavior similar to heuristics applied but are actually non peer-to-peer.

Heuristics based detection method is able to detect 82% of the peer-to-peer traffic. Detection percentage is quite satisfactory. Results show 12% false positives and 6 %. Moreover, detection percentage for the new methodology is more than the previous techniques.

5.3 Future Enhancements

With the introduction of new methodology, it is important to focus on modifications that can be made in it to improve its efficiency and accuracy. Following description highlights the future enhancements.

- In order to reduce the rate of false positives, it is required to do more analysis and research and modify the heuristics to meet this requirement.
- Peer-to-peer traffic is growing day by day. So, efficient techniques are required for detection. For that, heuristics may be developed considering packet size information.

- Much of the peer-to-peer traffic use is unmonitored. Since the computers running the peer-to-peer applications are usually connected to a network, they can be used to spread malware, viruses and trojans. So, the term malicious traffic is common in the context of peer-to-peer traffic. Therefore, one of important future goals is to detect malicious peer-to-peer traffic after detection of peer-to-peer class of service.
- The final aim is to perform network management tasks such as flow prioritization, traffic shaping/policing, security and diagnostic monitoring after detection of peer-to-peer traffic using intelligent heuristics.

BIBLIOGRAPHY

6 BIBLIOGRAPHY

1. V. Muthusamy, “An Introduction to Peer-to-Peer Networks Presentation for MIE456 - Information Systems Infrastructure II”, 2003
2. http://www.hightech-guides.net/p2p/p2pgens.html
3. http://www.hightech-guides.net/p2p/apps.html
4. http://www.hightech-guides.net/p2p/p2pworks.html
5. URL:www.srdc.metu.edu.tr/webpage/projects/artemis/documents/D3.1.1.3SOAv1.1_P2P - 2.doc
6. S. Kim, Y.H. Kang, Y. I. Eom, “An efficient contents discovery mechanism in pure p2p environments”, *In GCC (1)*, pp. 420–427, 2003.
7. www.wikipedia.com
8. http://www.jxta.org.
9. D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, D.Werthimer, “An experiment in public-resource computing”, Commun. ACM; 45(11): pp. 56–61, 2002.
10. J. Kangasharju, K. W. Ross, E. Al, “Adaptive content management in structured p2p communities”.
11. URL: http://www.hardwarecentral.com/hardwarecentral/reviews/
12. Constantinou, F. Mavrommatis, P. MIT, MA, “Identifying Known and Unknown Peer-to-Peer Traffic”, In *Network Computing and Applications, NCA*. Fifth IEEE International Symposium.,2006
13. A. Oram, “Peer-To-Peer: Harnessing the Power of Disruptive Technology”, First Edition, O’Reilly, March 2001.
14. http://www.gnutella.com, 2005.
15. www.Gnutella.com

16. www.kazaa.com

17. www.bittorent.com

18. S. Saroiu, P. Gummadi and S. D. Gribble, “Measurement study of peer-to peer file sharing systems”, *Multimedia Computing and Networking 2002*, 2002.

19. S. Sen and J. Wang, “Analyzing Peer-To-Peer Traffic Across Large Networks”, *IEEE/ACM Trans. on Networking*, Vol. 12, No. 2, pp. 219–232, 2004.

20. M. Kim, H. Kang and J. W. Hong, “Towards Peer-to-Peer Traffic Analysis Using Flows”, *Proc. of 14h IFIP/IEEE Workshop Distributed Systems: Operations and Management*, 2003.

21. WinMX, http://www.winmx.com/

22. Winny, http://www.nynode.info/

23. E.Jeffrey, A.Martin, M. Anirban, “Traffic Classification Using Clustering Algorithms University of Calgary”, 2500 University Drive NW, Calgary, AB, Canada .

24. A. McGregor, M. Hall, P. Lorier, and J. Brunskill, “Flow Clustering Using Machine Learning Techniques”. In *PAM 2004*, Antibes Juan-les-Pins, France, April 19-20, 2004.

25. S. Zander, T. Nguyen, and G. Armitage, “Automated Traffic Classification and Application Identification using Machine Learning”. In *LCN’05*, Sydney, Australia, Nov 15-17, 2005 .

26. T. Karagiannis, K. Papagiannaki, and M. Faloutsos, “BLINK: Multilevel Traffic Classification in the Dark”. In *SIGCOMM’05*, Philadelphia, USA, August 21-26,05.

27. A. W. Moore and K. Papagiannaki, “Toward the Accurate Identification of Network Applications”. In *PAM 2005*, Boston, USA, March 31-April 1, 2005.

28. S. Sen, O. Spatscheck, and D. Wang. Accurate, “Scalable In-Network Identification of P2P Traffic using ApplicationSignatures” , *Proceedings of the 13th International World WideWeb Conference*, pp. 512-521, NY, USA, May 04.

29. C. Dewes, A. Wichmann and A. Feldmann, “An Analysis of Internet Chat Sys- tems,” *Proc. of ACM SIGCOMM Internet Measurement Workshop 2003,* pp. 51–64, 2003.
30. K. P. Gummadi, R. J. Dunn and S. Saroiu, “Measurement, Modeling and Analysis of a Peer-to-Peer File-Sharing Workload,” *Proc. of ACM SOSP’03 2003,* pp. 314–329, 2003.
31. Snort, http://www.snort.org/
32. P. Barford, J. Kline, D. Plonka and A. Ron, “A Signal Analysis of Network Traffic Anomalies,” *Proc. of ACM IMW’02,* pp. 71–82, 2002.
33. S. Sen O. Spatscheck and D. Wang, “Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures,” *Proc. of ACM WWW’04,* 2004.
34. Subhabrata Sen, Oliver Spatscheck, Dongmei Wang, “Accurate, Scalable InNetwork Identification of P2P Traffic Using Application Signatures”, Florham Park, 2004.
35. A. W. Moore and D. Zuev, “Internet Traffic Classification Using Bayesian Analysis Techniques”. In *SIGMETRIC’05*, Banff, Canada, June 6-10, 2005.
36. M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, “Class-of-Service Mapping for QoS: A Statistical Signature-based Approach to IP Traffic Classification”. In *IMC’04*, Taormina, Italy, October 25-27, 2004.
37. A. McGregor, M. Hall, P. Lorier, and J. Brunskill, “Flow Clustering Using Machine Learning Techniques”. In *PAM, 2004*, Identification of Network Applications,2004.
38. F. Hernandez-Campos, A. B. Nobel, F. D. Smith, and K. Je_ay, “Statistical Clustering of Internet Communication Patterns”. Computing Science and Statistics, 35, July 2003.
39. M. Perényi, T. D. Dang, A. Gefferth, S. Molnár, “Identification and Analysis of Peer-to-Peer Traffic”, Budapest University of Technology & Economics, Budapest, Hungary.
40. S. Sen, O. Spatscheck, and D. Wang, “Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures”. In *WWW2005*, New York, USA, May 17-22, 2004.

41. Dews, A. Wichmann, and A. Feldmann, “An analysis of internet chat systems”, In *IMC’03*, Miami Beach, USA, Oct 27-29, 2003.
42. Haffner, S. Sen, O. Spatscheck, and D. Wang, “ACAS: Automated Construction of Application Signatures”, In *SIGCOMM’05 MineNet Workshop*, Philadelphia, USA, August 22-26, 2005.
43. A. W. Moore and D. Zuev, “Internet Traffic Classification Using Bayesian Analysis Techniques”. In *SIGMETRIC’05*, Banff, Canada, June 6-10, 2005.
44. Dempster, N. Paird, and D. Rubin, “Maximum likelihood from incomeplete data via the EM algorithm,” *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.
45. Zander, T. Nguyen, and G. Armitage, “Self-Learning IP Traffic Classification Based on Statistical Flow Characteristics,” in *PAM 2005*, Boston, USA, March 31-April 1, 2005.
46. “Automated Traffic Classification and Application Identification using Machine Learning”, In *LCN’05*, Sydney, Australia, November 15-17, 2005.
47. P. Cheeseman and J. Strutz, “Bayesian Classification (AutoClass):Theory and Results.” In *Advances in Knowledge Discovery and Data Mining, AAI/MIT Press, USA*, 1996.
48. Roughan, S. Sen, O. Spatscheck, and N. Duffield, “Class-of-Service Mapping for QoS: A Statistical Signature-based Approach to IP Traffic Classification,” in *IMC’04*, Taormina, Italy, October 25-27, 2004.
49. Moore and D. Zuev, “Internet Traffic Classification Using Bayesian Analysis Techniques,” in *SIGMETRICS’05*, Banff, Canada, June 6-10,2005. (
50. O. Satoshi, T. Yoichi, Kawashima k, “A Traffic Identification Method and evaluations for a Pure P2P Application”, Tokyo University of Agriculture and Technology,

51. R.Michael, C. Michael, “Finding Peer-To-Peer File-sharing Using Coarse Network Behaviors”, CERT/Network Situational Awareness, Software Engineering Institute.
52. Project Leder: N.Jenq, “Flow-based Peer-to-peer Identification”.
53. O. DJ, K.Hun, K.Jinoh, “Transport Layer Identification of P2P Super nodes”.
54. L. Hui, F.Wenfeng, “A Peer-To-Peer Traffic Identification Method Using Machine Learning”, Networking, Architecture, and Storage, 2007. NAS 2007. International Conference on Volume, Issue, 29-31 July 2007 Page(s):155 – 160
55. E.Jeffrey, M. Anirban, A. Martin, W. Carey, “Identifying and Discriminating between Web and Peer-to-Peer Traffic in the Network Core”.
56. H. Lukas, “P2P Population Tracking and traffic characterization of Current P2P File-sharing Systems”, ¨ Master Thesis,2004.
57. Karagiannis, A. Broido, M. Faloutsos, and K. claffy, “Transport Layer Identification of P2P Traffic”. In *IMC’04*, Taormina, Italy, October 25-27, 2004.