# A NOVEL METHOD FOR THE DETECTION OF P2P TRAFFIC IN THE NETWORK BACKBONE INSPIRED BY INTRUSION DETECTION SYSTEMS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

MURAT SOYSAL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING
JUNE 2006

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. İsmet Erkmen
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Dr. Şenan Ece (Güran) Schmidt
Supervisor

**Examining Committee Members**

| | | |
|---|---|---|
| Prof. Dr. Semih Bilgen | (METU-EE) | _____ |
| Dr. Şenan Ece (Güran) Schmidt | (METU-EE) | _____ |
| Prof. Dr. Hasan Güran | (METU-EE) | _____ |
| Prof. Dr. Cem Saraç | (ULAKBIM) | _____ |
| Assoc. Prof. Dr. Cüneyt Bazlamaçcı | (METU-EE) | _____ |

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last name : Murat SOYSAL

Signature :

# ABSTRACT

A NOVEL METHOD FOR THE DETECTION OF P2P TRAFFIC IN THE NETWORK BACKBONE INSPIRED BY INTRUSION DETECTION SYSTEMS

SOYSAL, Murat
M.Sc., Department of Electrical and Electronics Engineering
Supervisor: Şenan Ece (Güran) Schmidt
June 2006, 83 pages

The share of peer-to-peer (P2P) protocol in the total network traffic grows day-by-day in the Turkish Academic Network (UlakNet) similar to the other networks in the world. This growth is mostly because of the popularity of the shared content and the great enhancement in the P2P protocol since it first came out with Napster. The shared files are generally both large and copyrighted. Motivated by the problems of UlakNet with the P2P traffic, we propose a novel method for P2P traffic detection in the network backbone in this thesis. Observing the similarity between detecting traffic that belongs to a specific protocol and detecting an intrusion in a computer system, we adopt an Intrusion Detection System (IDS) technique to detect P2P traffic. Our method is a passive detection procedure that uses traffic flows gathered from border routers. Hence,

it is scalable and does not have the problems of other approaches that rely on packet payload data or transport layer ports.

Keywords: Academic Network, anomaly based detection, backbone, Bayesian Networks, Intrusion Detection Systems, P2P peer-to-peer, traffic characterization, ULAKBIM, UlakNet.

# ÖZ

## AĞ OMURGASINDA EŞLER ARASI İLETİŞİMİN TESPİT EDİLMESİ İÇİN SALDIRI TESPİT SİSTEMLERİNDEN ESİNLENİLMİŞ YENİ BİR YÖNTEM

Soysal, Murat
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Danışmanı: Şenan Ece (Güran) Schmidt
Haziran 2006, 83 sayfa

Eşler arası (peer-to-peer-P2P) protokolüne ait trafiğin toplam ağ trafiği içindeki payı diğer tüm ağlarda olduğu gibi Türkiye Ulusal Akademik Ağı (UlakNet) içinde de hızla artmaktadır. Bu artışın en önemli sebepleri P2P ağlarında paylaşılan dosyaların popülerliği ve Napster tecrübesinden sonra P2P protokollerinde yaşanan gelişmelerdir. Bu tip ağlarda paylaşılan dosyaların çoğunluğu telif hakkına tabi dosyalardır ve bu dosyalar bit bazında büyük boyutlara sahiptir. Bu çalışmada, eşler arası iletişim protokollerine ait trafiğin, akademik ağ omurgasında ölçeklendirilebilir tespiti için saldırı tespit sistemlerinden (intrusion detection systems-IDS) esinlenerek bir yaklaşım geliştirilmiştir. Bilgisayar ağlarına yapılan saldırıları tespit için geliştirilmiş sistemlerin çalışma prensipleri ile bir ağ trafiğindeki çeşitli protokollerden birinin (P2P) ayıklanması arasındaki benzerlikler incelenmiş ve daha önceki çalışmalar sonucu ortaya çıkan bir saldırı tespit sistemi P2P trafiği tespit etme problemine adapte edilmiştir. Bu çalışma sonucu ortaya çıkan metot, akademik ağ omurgasında bulunan omurga yönlendiricilerinin sağladık trafik akış izlerini kullanmaktadır. Böylece, P2P trafiğini

belirlemek için daha önce önerilen, 4. tabaka iletişim portu tabanlı ve ağ trafiği paketleri açılarak yapılan imza tabanlı tespit yöntemlerinin yaşadığı problemlerin üstesinden gelinmiştir. Ayrıca bu yöntem tüm P2P protokollerine ve ağ omurgalarına uygulanabilir bir yapı içermektedir.

Anahtar Kelimeler: Bayesian ağları, eşler arası, P2P, Saldırı Tespit Sistemleri, trafik tanımlama, ULAKBIM, UlakNet, Ulusal Akademik Ağ.

# ACKNOWLEDGMENTS

I would like to thank Dr. Şenan Ece (Güran) Schmidt for her creative ideas, valuable supervision and patience. Her support on this thesis work increased my motivation to the top level and her guidance encouraged me to complete this thesis.

I want to thank my managers in ULAKBIM for their support on this research. I also want to thank my colleagues at ULAKBIM for their continuous assistance.

I would also like to thank my parents and my brother for giving me encouragement and patience during this thesis and all kind of supports during my whole education. In addition, I wish to thank my grandparents for their best wishes and financial support for my computer.

Finally, I have very special thanks for my dear wife Neşe Soysal, for her precious support and patience during this thesis.

# TABLE OF CONTENTS

PLAGIARISM ..... iii
ABSTRACT ..... iv
ÖZ ..... vi
ACKNOWLEDGMENTS ..... viii
TABLE OF CONTENTS ..... ix
LIST OF TABLES ..... xi
LIST OF FIGURES ..... xii
LIST OF ABBREVIATIONS ..... xiv
CHAPTERS
1. INTRODUCTION ..... 1
2. BACKGROUND ..... 6
2.1 ULAKBIM and UlakNet ..... 6
2.2 P2P Communication ..... 9
2.2.1 P2P Networks ..... 10
2.2.2 P2P Traffic Detection ..... 14
2.3 Intrusion Detection Systems ..... 19
2.3.1 Signature Based IDS ..... 20
2.3.1 Anomaly Based IDS ..... 21
2.3.1 Hybrid IDS ..... 24
2.4 Bayesian Networks ..... 24
3. PROPOSED METHODOLOGY FOR P2P TRAFFIC DETECTION ..... 27
3.1 Overview of Our Method ..... 28

3.2 Differentiating Feature Discovery ..... 30
3.2.1 Border Routers ..... 31
3.2.2 The Inspector ..... 32
3.2.2.1 Signature Based P2P Detection Tool ..... 33
3.2.2.2 Storing and Processing the NetFlow Data ..... 37
3.3 Composition of Conditional Probability Tables ..... 45
3.4 Realization of Bayesian Networks ..... 51
4. RESULTS AND EVALUATION ..... 52
5. CONCLUSION AND FUTURE WORK ..... 56
REFERENCES ..... 60
APPENDICES
A. ULAKBIM Acceptable User Policy ..... 64
B. List of Popular P2P Protocols ..... 67
C. A Sample Flow-stat Output ..... 69

# LIST OF TABLES

**Table I** P2P Clients and Transport Layer Ports ..... 14
**Table II** CPU Usage Time (t) Distribution ..... 22
**Table III** Conditional Probability Table of Positive ..... 26
**Table IV** IpP2P Options and Detection Rating ..... 34
**Table V** Comparison of our method with signature based detection techniques ..... 58
**Table VI** List of Popular P2P Programs ..... 67

# LIST OF FIGURES

**Figure 1** UlakNet Topology Map ..... 7
**Figure 2** UlakNet Weather Map ..... 9
**Figure 3** Centralize P2P Network Topology ..... 11
**Figure 4** Decentralized P2P Network Topology ..... 12
**Figure 5** Hybrid P2P Network Topology ..... 13
**Figure 6** An Example Bayesian Network ..... 26
**Figure 7** Testbed used for feature discovery and verification ..... 31
**Figure 8** IpTables Packet Journey ..... 34
**Figure 9** IP Packet Size Distribution ..... 39
**Figure 10** Packets per Flow Distribution ..... 39
**Figure 11** Octets per Flow Distribution ..... 40
**Figure 12** Flow Time Distribution ..... 40
**Figure 13** Differentiating Feature Discovery ..... 44
**Figure 14** Conditional Probability Table $d_1$ ..... 46
**Figure 15** Conditional Probability Table $d_2$ ..... 47
**Figure 16** Conditional Probability Table $d_3$ ..... 47
**Figure 17** Conditional Probability Table $d_4$ ..... 48
**Figure 18** Conditional Probability Table $d_5$ ..... 48
**Figure 19** Conditional Probability Table $d_6$ ..... 49
**Figure 20** Conditional Probability Table $d_7$ ..... 49
**Figure 21** Conditional Probability Table $d_8$ ..... 50
**Figure 22** Conditional Probability Table $d_9$ ..... 50
**Figure 23** Proposed Bayesian Network for P2P Traffic detection ..... 51
**Figure 24** Success rate of P2P Contributor decision ..... 53
**Figure 25** Success rate of not P2P Contributor decision ..... 53
**Figure 26** Success rates of Differentiating Features ..... 54

# LIST OF ABBREVIATIONS

| | |
|---|---|
| AS | Autonomous System |
| ATM | Asynchronous Transfer Mode |
| AUP | Acceptable Use Policy |
| CGI | Common Gateway Interface |
| CPT | Conditional Probability Table |
| CPU | Central Processing Unit |
| DAG | Directed Acyclic Graph |
| IDS | Intrusion Detection Systems |
| ISP | Internet Service Providers |
| NAT | Network Address Translation |
| NBAR | Network Based Application Recognition |
| NOC | Network Operating Center |
| NRENs | National Research and Educational Networks |
| PoP | Point of Precedence |
| Pps | Packet per Second |
| QoS | Quality of Service |
| RAM | Random Access Memory |
| SMS | Short Message Service |
| TTL | Time to Live |
| TÜBİTAK | The Scientific and Technical Research Council of Turkey |
| ULAKBIM | Turkish Academic Network and Information Center |
| ULAKNET | Turkish National Academic Network |
| VoIP | Voice over IP |

# CHAPTER I

# INTRODUCTION

Before 1990's the dominating network application architecture was the client-server architecture. The traditional client-server communication includes a dedicated server and a number of clients. Every client sends a request to the server for the given service (files, web pages, etc) and then the server responses the request. Peer-to-peer (P2P) application architecture which came around 1990s has a different paradigm. P2P communication is the sharing of the computer resources by direct exchange. Every host in the network both uses the resources of the other hosts and shares its own sources which enable every P2P client to function as a server. The resources shared in P2P communication can be the storage area, the processor power and the files stored in the memory. To be involved in P2P communication, a user installs P2P application software, which can be downloaded from the Internet for free, to his computer and makes search queries according to his interest. In addition, since the aim of P2P communication is direct exchange, advanced protocols are developed to enable faster and efficient exchange such as downloading a file simultaneously from multiple sources. The easy procedure for joining the P2P network, searching a file and faster downloading protocols made the P2P communication most popular file searching and sharing protocol.

On the other hand, the bandwidth consuming characteristic of P2P communication and the contents of the shared files are the main concern from the Internet Service Providers' (ISP) point of view. P2P protocols evolve for enhancing the searching and downloading features which maximizes the bandwidth usage. In today's Internet, peer-to-peer (P2P) communication has the largest bandwidth share which is still

increasing [1], [2]. The growth of P2P communication share is a problem for the links with a limited capacity. The greedy behavior of P2P squeezes the other applications into small link capacities. In addition, the content shared by P2P protocols contains popular files most of which are under copyright protection. These files include music files, movies, and software such as operating systems and games. These files can have sizes up to hundreds of megabytes [3], [4]. Sharing such kind of files is forbidden by law (5846 Sayılı Fikir ve Sanat Eserleri Kanunu).

As the manager of the Turkish National Academic Network (UlakNet), Turkish Academic Network and Information Center (ULAKBIM) [6] also faces these problems with P2P mentioned above. The limited capacity of UlakNet's uplinks to the global internet and the links of nodes to the UlakNet backbone are mostly used by P2P communication traffic. The WeatherMap (see Figure 2) of UlakNet backbone (see Figure 1) shows that the uplink capacities are utilized over 90% percent in the working hours which hardens the usage of the network for academic purposes. In addition, delay intolerant applications such as Voice over IP (VoIP), video conference are negatively affected by the congestion in crowded links. The Acceptable Use Policy (AUP) of ULAKBIM prevents user of UlakNet from generating high disruptive traffic patterns, which affect the quality of service on the UlakNet. (see Appendix A). Therefore, it is necessary for ULAKBIM to detect the P2P communication traffic to limit it to a certain bandwidth and maintain the Quality of Service (QoS) rules. The transmission of the materials (including texts, articles, books, films, music) that infringes the copyright of another person is also listed as the unacceptable use in the AUP. It should also be possible for ULAKBIM to identify the P2P traffic contributors to meet the legal responsibilities in case of a lawsuit**.**

Since the introduction of Napster [8], which is the first P2P application in the current sense, analyzing and detecting P2P traffic has been an important issue for the network traffic engineers to provide insights for ISPs about engineering their network traffic and planning the capacity accordingly [4]. The detection techniques

followed the improvements in the P2P protocols. The detection techniques can be grouped into two: Port based detection and Signature based detection.

Port based detection is mainly the identification of Transport Layer Ports of the P2P applications [1], [2]. Either the IP packet headers or the flow traces supplied by the routers are used in the analysis of Port Based Detection. Signature based detection is the investigation of the IP packet payload to find any matches for specific strings which characterize the P2P protocol [18], [19]. The network traffic is either intercepted or mirrored to another link to investigate the payloads of network traffic packets.

Both of these two detection techniques have weak points. The port based detection technique became useless by the adaptation of dynamic port usage feature to the P2P applications. New versions of P2P applications can use transport layer ports of well known services such as 21(File Transport Protocol), 25 (Simple Mail Transport Protocol) or 80 (Hyper Text T Protocol). Despite the accurate detection capability, signature based detection has technical and ethical disadvantages. The backbone and global Internet connections reach 10 Gbps capacities. At these speeds, the traffic analysis of packet payloads by intercepting the traffic forces the limits of the buffers and memories of computer architectures with current capacities. Mirroring the traffic to another link, and storing it for further analysis of the packet payload requires vast amount of storage area. For example for a 622 Mbps link, it requires a 400 TeraBytes storage area for a 1 hour data when the link is 100% utilized. The success rate of the signature based detection encouraged the P2P application programmers to hide the payload of the P2P packets from man-in-the-middle. A beta version of Azuerus [23] was released in which end-to-end encryption feature is embedded. The discovery of the signatures and the analysis procedure of signature based detection become useless with the encrypted packet payload. Finally, the ethical disadvantage of signature based detection is the violation of the user data privacy by inspecting the user data packet payloads.

The limitations to the current P2P traffic detection techniques for backbone usage and the upcoming threats drive further research on P2P traffic detection. Similar to other ISPs, the limited uplink capacities of UlakNet and the high prices of bandwidth in telecommunication market in Turkey, urge the researches on the P2P traffic and its contributor's detection. Our research is motivated by the concerns of ULAKBIM for the increasing volume of the P2P traffic in their network.

In this thesis, we propose a novel method for the detection of P2P traffic observing the similarities between detecting the P2P traffic in the entire network traffic and detecting intrusions in a computer system. Intrusion detection systems (IDS) are "intrusion alarms" of the computer security field where an intrusion is the action which a user takes it illegally. Our proposed method adopts Bayesian network based intrusion detection systems. We use the traffic flow information gathered from border routers of the National Academic Network of Turkey (UlakNet).

We tested our method on two nodes of UlakNet with link capacities 34 and 155 Mbps and achieved successful results. In the tests performed, our method has detected P2P traffic contributors with 10% false negatives and 17% false positives.
Among many problems of P2P traffic, bandwidth consumption is the most important one for ULAKBIM. Hence, we test our model to detect the "heavy hitters" of P2P traffic who download and upload large amount of data. However, our methodology is entirely general and can be applied to any other network. Our method is scalable to high link speeds and a broader range of P2P protocols including the ones that support end-to-end encryption.

The thesis is organized as follows:
In Chapter 2, first, an introduction of ULAKBIM and UlakNet is given. Then, the P2P communication protocol is introduced. Next, the developed P2P detection techniques are reviewed in detail. Brief information about intrusion detection systems is supplied in the next section. Finally, the Bayesian Networks are introduced.

In Chapter 3, we present our method. In the first part, the topology of our test bed and the details of separation of P2P and non-P2P flows are given. Identification of differentiating features, composition of Conditional Probability Tables and realization of Bayesian Network procedures are explained in detail.

The results of our method analyzed on the traffic of UlakNet are given in Chapter 4. A conclusion and a discussion on future work are also included.

# CHAPTER II

# BACKGROUND

Our research is motivated by the concerns of ULAKBIM for the increasing volume of the P2P traffic in the National Academic Backbone. We used the flow traces gathered from border routers of UlakNet to compose and test our P2P detection model. Hence, we first introduce ULAKBIM and UlakNet. Then, we present an overview of the P2P communication and a literature survey on the P2P detection. Later, we present a literature survey on IDS techniques. Finally, we introduce the Bayesian Networks.

## 2.1 ULAKBIM and UlakNet

Turkish Academic Network and Information Center (ULAKBIM) [6] manages the academic network in Turkey. It was founded as a service unit, in association with the Scientific and Technological Research Council of Turkey (TUBITAK) [7], in 1996. ULAKBIM fulfills the functions of providing information technology support to the national innovation system. This support includes carrying out research and development work in the field of information technologies and enabling the information services over the national academic network. ULAKBIM is composed of two units: UlakNet and Cahit Arf Information Center. The Turkish Academic Network (UlakNet) is an interactive system which uses new technologies and connects the innovation centers to each other in the national scale. Cahit Arf Information Center has been providing nationwide information and document delivery services using traditional and electronic means in order to meet the

information needs of universities, public and industrial sectors, and to contribute to the production of academic information.

The nodes connected to UlakNet are;

- Universities
- Research and Development Organizations
- Governmental Organizations
- Military and Police Academies

There are 110 distinct units connected to UlakNet. The total number of the nodes reached to 650 with the connection of Higher Institutes of Technology, Faculties, Graduate Schools, Schools of Higher Education, Conservatories, Vocational Schools and Research Centers of the universities. Over 80.000 lecturers and 1.700.000 students are using UlakNet in these nodes.

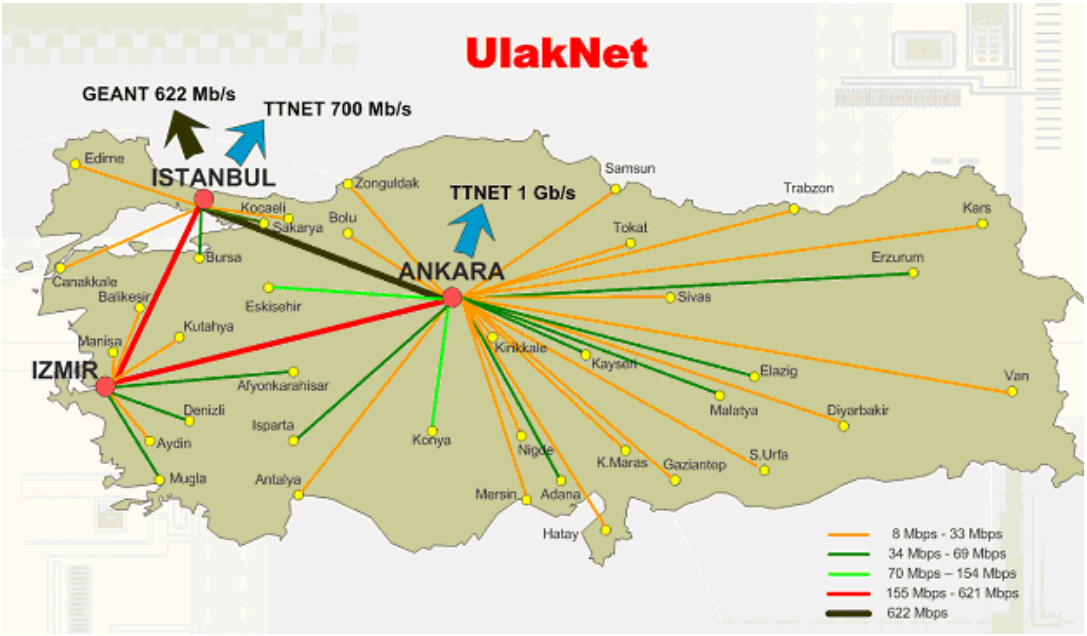


**Figure 1** UlakNet Topology

The connections of the nodes are realized by the lines rented from Turkish Telco using technologies including Frame-Relay, Asynchronous Transfer Mode (ATM) and Leased Line. UlakNet backbone is composed of three Point-of-Presence (PoP) located in Ankara, Istanbul and Izmir. The PoP in Ankara is in the ULAKBIM building, the one in Istanbul is in Istanbul Technical University and the one in Izmir is located in Dokuz Eylül University. The backbone connection between Ankara and Istanbul is a 622 Mbps ATM line. Other two backbone connections are 155 Mbps ATM links.

UlakNet has two global Internet connections. One is from Ankara with 1 Gbps and the other is from Istanbul with 700 Mbps. These links are supplied by Turkish Telco with Metro Ethernet technology. In addition, Turkish Academic Network has an uplink to the European Academic Network. This uplink is between Istanbul and Athens and has a capacity of 622 Mbps (see Figure 1 for a detailed backbone map).

Capacities of the backbone connections, the global Internet uplinks and the connections to the nodes are lower when compared to the other National Research and Education Networks (NRENs) in Europe. The main reason of these low capacities is the high prices in Turkish Telecom market. Since the monopoly in this area came to an end in 27.06.2003, a discount in the prices are being expected in several years. On the other hand, bandwidth usages in global Internet uplinks are saturated currently (see Figure 2). These heavily loaded links of UlakNet are the main concerns from network engineering point of view.

ULAKBIM signed an Acceptable Use Policy (AUP) (Appendix A) with the heads of the nodes as a step in the connection procedure. This policy is signed by the president of the university or the director of the Research and Development units. Acceptable Use Policy defines the rules applied to all the users of "UlakNet". The complete policy is supplied in the Appendix A. The main topic

of this policy related to our work is the “Unacceptable Use” part, especially the entry 9.7 which is “The transmission of material (including, without limitation; the works in the form of texts, articles, books, films, music) such that this infringes the copyright of another person” and the entry 9.3. “Generating highly disruptive traffic patterns, which affect the quality of service on the UlakNet” are the highly related ones to the P2P communication. This relation is discussed in detail in the next section of this chapter.

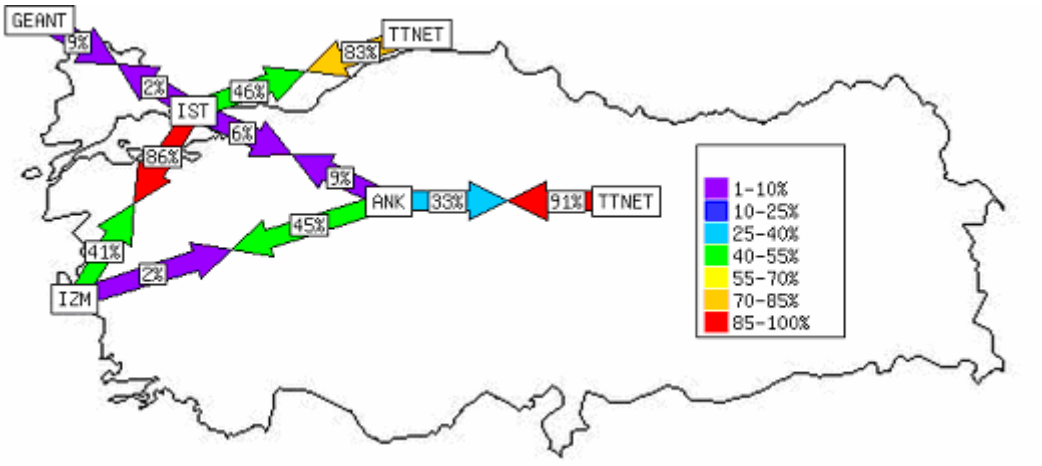


**Figure 2** UlakNet Weathermap (Link Utilizations)

## 2.2 P2P Communication

P2P is the sharing of the computer resources by direct exchange as opposed to the traditional client/server network architecture in which one or more computers are dedicated as servers. This includes giving the serving ability for all clients in the network. Every P2P client both gets service from the P2P network and serves to the network at the same time. The shared resources in P2P networks can be the processor power, the storage area or the files stored in computers.

The two terms, P2P networks and P2P clients should be defined for the sake of better understanding of the P2P communication. A P2P network is a protocol including specific rules that bring various P2P clients together. A P2P client is simply a computer application which interacts with other clients through the network. Two types of packets are exchanged in P2P communication: control and data packets [17]. The packets used for registering to the network, making search queries and reporting search results are the control packets. The packets containing the parts of the shared file traveling from one client to another client are the data packets.

### 2.2.1 P2P Networks

The P2P communication as in the current sense became a popular application with the introduction of Napster [8] in 1999 by Shawn Fanning. Napster's infrastructure is based around centralized index servers that maintained a database of all the content on the network and clients currently logged on at any time. When a user wants to find a file, he simply searches Napster. The request for a file is directed to the central server by the help of Napster client. The server checks the “known files list” and provides the internet location of the users who have the file if they exist to the requester user. The Napster Network had its peak usage in February 2001 with 29.4 million registered users and 2.79 billion shared files [9]. This rapid spread of Napster brought the complaints of Metallica, Dr.Dre and Recording Industry Association of America. Consequently, the servers in Napster were shutdown in late 2001. Although the usage of centralized servers facilitated the indexing and searching algorithms, it also enabled lawyers to shutdown the Napster network easily. The lawyers arrived to the server room and turned off the servers, which was the end of Napster Network.

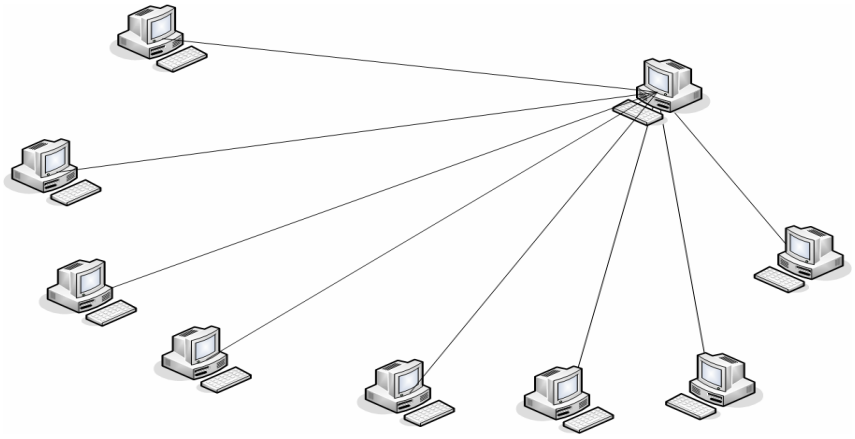

**Figure 3** Centralized P2P Networks Topology

The second generation of P2P protocols, Gnutella (with clients BearShare, Morpheus, Mutella, Phex) [10], was created to overcome the design flaw in Napster that led to easy shutdown. In the basic sense, Gnutella works by connecting the clients directly to other clients resulting in a distributed/decentralized architecture. When a user starts Gnutella client, he connects to a certain number of users whom also connected to a certain number of clients. In order to search a file, Gnutella client asks for the file to the directly connected clients. When a client receives a file request, it searches its own database for the file and replies the request if the search is successful. If the search is unsuccessful in its own file database, the client forwards the request to the directly connected nodes except the requester. This spreading search procedure continues whether the file is found or a Time-to-Live (TTL) value expires. Shutting down a Gnutella network is not as easy as shutting down Napster because it requires disabling all of the communicating clients. However, Gnutella suffers from a poor search algorithm which generates excessive amount of control traffic.

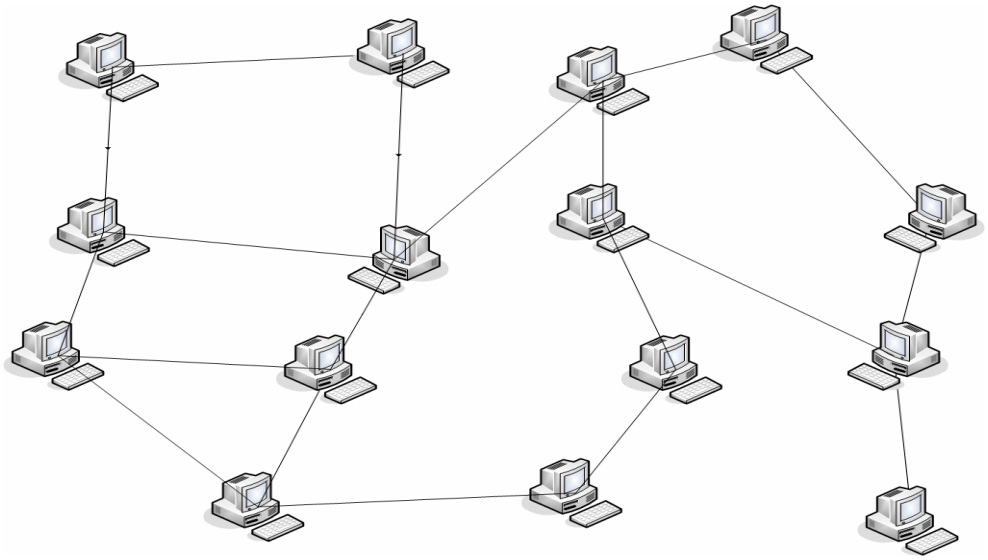

**Figure 4** Decentralized P2P Networks Topology

The inefficiency of the distributed/decentralized architecture forced further improvements on P2P network topologies resulting in the third generation of P2P protocols Fast Track (with clients Grokster [11], Kazaa [12] and iMesh [13]). Fast Track combines the benefits of the centralized and distributed/decentralized topologies. This hybrid network is spanned by a set of Super Nodes which act similar to the Napster index servers. These Super Nodes are selected from the users running Fast Track clients and have a high capacity link to the Internet. These nodes periodically index the file databases of the users connected to them and share this information with other Super Nodes which reduce the amount of the control traffic carried in the network and increase the efficiency of the search queries. A similar protocol Direct Connect [14] adds the feature of chatting with the users connected to the same super node which makes the protocol more community oriented.

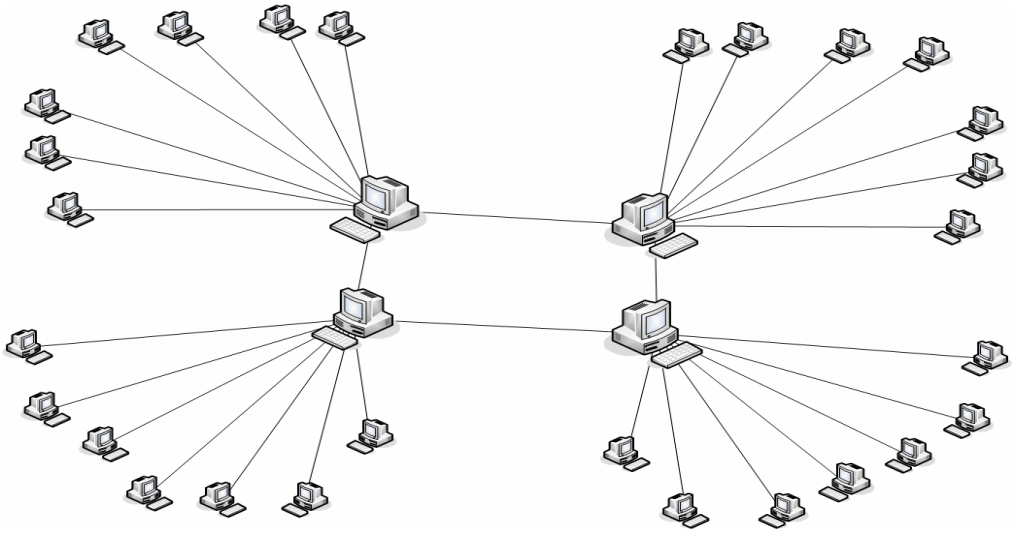

**Figure 5** Hybrid P2P Networks Topology

The next step in the P2P evolution is BitTorrent [15] which is mainly designed to discourage the leeches who only download files from P2P network without supplying any. The BitTorrent network uses a principle called *tit-for-tat* which means giving files in order to get files. This is guaranteed by arranging the download speeds of clients according to the number of files shared. In addition, the searching procedure of BitTorrent is very different than its ancestors. It uses web sites (announce sites) to distribute ".torrent" files. These ".torrent" files have information about BitTorrent users which share all or a part of a specific file. A user issues a web search to find the ".torrent" file of a file he is interested in. After downloading the ".torrent" file, which has an approximate size of 50 kb, the user opens it by a BitTorrent client program. BitTorrent client uses the ".torrent" file to contact to a computer called *tracker* which hosts the information about the other clients who have the entire interested file (*seeds*) or some of it. The tracker composes a *swarm* which is the network including all the clients downloading or uploading the same file. Tracker regulates trading the pieces of file inside the swarm. Web search facility and tit-for-tat principle has rapidly increased the use of BitTorrent Network.

### 2.2.2 P2P Traffic Detection

The efforts to detect the P2P traffic followed the development of the P2P applications. Legal action was successfully taken to close Napster. In addition to that, the P2P contributors could easily be detected by simply inspecting the connections to the Napster servers from traffic flows.

Although the inefficient search procedure caused second generation P2P networks to disappear suddenly, the notion of using decentralized topologies was inherited to the next generation. After the introduction of hybrid topologies in the third generation P2P Networks, P2P communication's packet exchange has gone under further investigation to identify P2P packets. Those investigations resulted into the Transport Layer port number based detection techniques. The flooding of all of the control packets and some data packets are performed on specific and static transport layer ports (Table I). P2P users are detected by identifying those ports [1], [2]. Most of the firewalls are equipped with rules denying the traffic on the identified P2P ports. P2P client programmers responded by adding dynamic port feature in the new versions which made the port based detection useless. In the new versions, when a user starts the P2P client, it tries to communicate with the Super Nodes by using the default port. If the connection is unsuccessful client changes the port in a random manner. In addition, the user can also set the operating port manually.

**Table I** P2P Clients and Transport Layer Ports

| P2P Client | Transport Layer Port(s) |
|---|---|
| eDonkey200 | 4661-4665 |
| FastTrack | 1214 |
| BitTorrent | 6881-6889 |
| Gnutella | 41170 |
| DirectConnect | 411-412 |

Port based P2P detection lost most of its use by the invention of dynamic port assignment property in P2P clients. The default ports for P2P clients supply clues instead of acting individually in P2P detection.

The next move in P2P detection efforts was to discover the *signatures* of P2P client programs. It was accomplished by the investigation of data and control packets of P2P communication to find any matches for specific strings which characterize the P2P protocol [16], [17], [18], and [19]. Every P2P client program inserts specific query or response commands for other clients to understand the concept of communication. The payload of every single packet is examined to look for these special strings. Therefore, the network traffic to be examined is intercepted or cloned to a separate location for further investigation in signature based P2P detection techniques.

A detailed investigation on P2P client program signatures is supplied in [17]. The signatures used in the data packets are identified and listed in their work. Next, we present some of their identified signatures to show how they look like.

- **Gnutella Network**

The TCP connection creation assumes following format:

*GNUTELLA CONNECT/<protocol version string>\n\n*

And the response for this request in TCP connection is:

*GNUTEALLA OK\n\n*

And also there is a handshake session within each content download. The session starts with:

*GET /get/<File Index>/<File Name>*
*/HTTP/1.0 \r \n*
*Connection: Keep-Alive\r\n*
*Range: byte=0-\r\n*
*User-Agent: <Name>\r\n*

*\r\n*

The response in handshake session is:

*HTTP 200 OK\r\n*

*Server: <Name>\r\n*

*Content-type: \r\n*

*Content-length: \r\n*

*\r\n*

- **eDonkey Network**

After investigating the data and control packets, the following pattern is found directly after the TCP header:

```
1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
+-+-+-+-+-+-+-+-+
| Marker |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| packet Length (4 Bytes) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Message type |
+-+-+-+-+-+-+-+-+
```

where the marker value is always 0x3e in hex representation.

- **DirectConnect Network**

Client-to-client and server-to-client communication use TCP in DirectConnect Network. The TCP commands are in the form of:

*$command_type field1 field2 … |*

The list of valid TCP command is:

*MyNick,*

*Lock, Key, Direction, GetListLen, ListLen, MaxedOut, Error,*

*Send, Get, FileLength, Canceled, HubName, ValidateNick, ValidateDenide,*

*GetPass, Mypass, BadPass, Version, Hello, Logedin,*

*MyINFO, GetINFO, GetNickList, NickList, OpList, To, Connect-*

*ToMe, MultiConnectToMe, RevConnectToMe, Search, MultiSearch,*

*SR, Kick, OpForceMove, ForceMove, Quit*

- **BitTorrent Network**

The client-to-client communication starts with a handshake in BitTorrent Networks. The BitTorrent header of the handshake packets is:

*<a character(1 byte)><a string(19 byte)>*

The first byte is a fixed character with value '19' and the string is 'BitTorrent Protocol'.

- **Kazaa Network**

The request from a client to a Super Node contains the following header:

*GET /.files HTTP/1.1\r\n*
*Host: IP address/port\r\n*
*UserAgent: KazaaClient\r\n*
*X-Kazaa-Username: \r\n*
*X-Kazaa-Network: KaZaA\r\n*
*X-Kazaa-IP: \r\n*
*X-Kazaa-SupernodeIP: \r\n*

The response has the following header:

*HTTP/1.1 200 OK\r\n*
*Content-Length: \r\n*
*Server: KazaaClient\r\n*
*X-Kazaa-Username: \r\n*
*X-Kazaa-Network: \r\n*
*X-Kazaa-IP: \r\n*
*X-Kazaa-SupernodeIP: \r\n*
*Content-Type: \r\n*

Both in the signature discovery and the detection phases of signature based P2P traffic detection, the payloads of the network traffic packets have to be inspected. The methods based on the payload investigation have a list of disadvantages including continuous updating of the signatures, violating the user data privacy by payload inspection, and technical problems with intercepting the traffic in the backbone or the huge storage space required for mirroring the traffic. Packet

inspection becomes hard and inefficient as the links are reaching 10 Gbps capacities. At these speeds, packet per second (pps) level is very high for the packet processing capability and the buffers of the current packet analyzing programs. A considerable amount of delay can be introduced which is a problem for "delay intolerant" applications such as voice over IP. As an example, Cisco [21] enables Network Based Application Recognition (NBAR) [22], for P2P detection on the edge routers but not on the core routers.

Another problem about applying the signature based detection techniques, which is in fact the most threatening one, is the usage of “end-to-end” encryption in P2P communication. The success of signature based detection techniques in edge applications motivated the P2P application programmers to hide the P2P packets from the man-in-the-middle. A BitTorrent client Azureus [23] released a beta version which includes “end-to-end” encryption of BitTorrent handshake to test the efficiency issues. The results are successful as the programmers of Azureus clarified. They also announce that the 2.4 version of the program will include the encryption feature in a stable manner which will render the signature based detection useless.

Another detection technique for the P2P traffic is using a crawler which is a computer running a P2P client and gathering information about the P2P network by that client's connections [24]. Some popular files are loaded in the crawler machine and shared, so too many clients from the P2P network will start file download from this specific computer. The IP numbers of the clients can easily be recorded by investigating the traffic flows. Although this technique has few false positives, it cannot identify all P2P users in the network. The users in the P2P networks to which the client on crawler does not belong and the users that never download any file from the crawler are not identified. On the other hand, data gathered from the crawler can be combined with the other techniques to improve the performance.

There exist some researches on identification of P2P traffic from network traffic flow traces. The general problem in using flow traces to detect P2P traffic is the

unsuccessful description of P2P behavior from flow traces. We give the details of the information included in flow traces in section 3.2.1. The main problem of designing a P2P traffic detection model is that there are no major differences between P2P and non-P2P traffic characteristics. Previous researches, [2], [16] reveal some traffic characteristics that are different for P2P and non-P2P traffic such as usage of TCP and UDP ports at the same time, connection duration, and IP packet size distributions. On the other hand, these characteristics can not be used individually to perform the P2P traffic detection. In addition, inspecting more than one characteristic brings the problem of result aggregation. Generally a separate model is used for analyzing each characteristic. The outputs of each model are summed up in the decision phase and the result is compared to a threshold.

Aggregation of models by summing up the outputs and comparing the result to a threshold can mislead the technique. The threshold must be small enough to catch a traffic flow which results in a variation in only one characteristic's behavior. On the other hand, small variations of all characteristics in a non-P2P flow can be identified as P2P since the sum of variations will exceed the threshold.

## 2.3 Intrusion Detection Systems

Intrusion detection can be defined as the process of identifying malicious behavior that targets a network and its resources [5]. Intrusion Detection Systems (IDS) are groups of software, hardware and rules which periodically monitor the network elements and generate an alarm incase of an intrusion. The network element under inspection can be an application running on a computer, the operating system on the computer, a group of computers or all of the computers in a network. According to the type of the inspected element, IDS analyze the related logs such as access logs and error logs and compare them to the previously defined rules. If a match to an intrusion rule occurs, an alarm is generated. The alarm can be in several forms such as a line in the logs, an email to the system administrator or a Short Message Service (SMS) to the Network Operating Center (NOC) members' cellular phone.

The wide range of elements brings a variety to IDS applications. On the other hand, all intrusion detection systems can be grouped into three according to the rule formation procedure.

### 2.3.1 Signature Based (Misuse) IDS

Signature based detection is analyzing the events to detect predefined patterns of intrusions. The known intrusion techniques are characterized and a signature is composed for every attack type. Later, logs and monitoring results are inspected to find a match with an attack signature [26]. The logs can belong to the network traffic [26], [27], to the operating system [28], [29] or to an application [30]. For example a Smurf attack is described as:
*icmp number (receive( )) ≥m* and *addr ( ) = (host,broadcast)*
which means, the number of packets that a host received with "network broadcast address" as the destination address is equal or more than a threshold m [31].

The ***known*** attack types are converted into signatures such as the one of Smurf attack and then the related logs are investigated in the signature based IDS. To detect a Smurf, the IDS has a rule for logging the traffic hits to the network broadcast address and then compares the lines in the log to a threshold.

Another example of signature involves Common Gateway Interface (CGI) scripts. A popular open source IDS SNORT [20] is used to inspect the packet payloads of the network traffic and search for specific strings. An exploit is used by hackers that probe the target Web server for known CGI bugs. For example, the *phf exploit* allows an attacker to return any file instead of the proper web page. This is achieved by a request which includes the following string in packets:

*GET /cgi-bin/phf?*

If the network hosting the target web server includes a SNORT IDS, the request of the attacker will be detected by the IDS and the preprogrammed actions (dropping

the packets, generating an alarm …) will be taken. A similar CGI attack signature in the packet payloads is as follows:

*cgi-bin/phf?Qalias=x%0a/bin/cat%20/etc/passwd*

This attack becomes dangerous if the following line is appended to the web server's access log.

*GET /cgi-bin/phf?Qalias x%0a/bin/cat%20/etc/passwd HTTP/1.0" 200 267*

Another example of the signature based IDS is the spam filtering programs [32], [33]. They use header and text analysis, DNS block lists, and collaborative filtering databases. Most of them issue a scaling for an email to identify it as spam or not. Every hit for the predefined signatures in the header and text analysis advances the score of the mail. The signatures in this phase can be the usage of words such as Viagra, free, download, etc. and including an HTTP link in the mail body. The signatures are both preprogrammed and also learned from the email that the users mark as spam.

The advantage of the signature based IDS is the minimal number of false positives achieved by the complete characterization of harmful actions. On the other hand, whether the inspected element is a network or a host, this technique is not useful for the undiscovered attacks. Since this technique is not an adaptable one by its nature, the number of false negatives is high because of the undiscovered attacks.

### 2.3.2 Anomaly Based IDS

Anomaly based techniques follow a totally different approach with respect to misuse detection techniques. This technique is based on composition of models or profiles of the "*normal use*". This normal use can belong to the users of a system [34], [35], to an application [36], [37] or the network resources [38], [39], [40]. After forming

the profile, every use is compared to the profile and the deviations are recorded as attack. Similar concepts are being used to detect telecom and credit card frauds [54].

An example of composing a profile for normal usage is given in [35]. A user's normal profile is formed by the help of *individual measures*. These are file accesses, CPU time usage and the terminals used to log on. By observing the values of individual measures over many audit records and selecting appropriate intervals, a categorization is formed which is called *the frequency distribution*. All of the individual measures are calculated and compared to the frequency distribution for an audit data. The variations are compared to a threshold to identify the event as normal or abnormal.

An example distribution for CPU usage time is as follows:

**Table II** CPU Usage Time (t) Distribution

| t (msec) | Percentage |
|---|---|
| $0 \leq t < 1$ | 0.5 % |
| $1 \leq t < 2$ | 7 % |
| $2 \leq t < 4$ | 15 % |
| $4 \leq t < 8$ | 42 % |
| $8 \leq t < 16$ | 12 % |
| $16 \leq t$ | 23.5 % |

In [37], a normal usage characterization of programs is composed. These programs include the programs that run as a daemon and do not, the programs that vary widely in size and complexity and different kind of intrusions (Trojan programs, denial-of-service and buffer overflows). Only the ones that run with privilege are inspected

since they have the greatest potential for harm to the system. A *trace* is defined as the list of system calls issued by a single process from the beginning of its execution to the end. As an example, the *Inetd* program is inspected. Inetd maintains passive sockets on a variety of these well-known ports. When a new connection is created, Inetd starts a program to handle the connection, based on a configuration table. In this way, one program can handle incoming connections for a variety of services. It is started as a foreground process which initiates a daemon process to run at the background and then exits. Then the daemon process initiates some child process to perform initializations. These child processes are nearly identical. A dataset including the traces of startup process, daemon process and a representative child process is used in the Inetd. An intrusion against Inetd is a denial-of-service attack which ties up network connection resources. Another dataset is composed of the same traces at an attack instant. The result of the comparison between two datasets is only a deviation in daemon process traces. Therefore, for Inetd attacks a signature is composed which looks for the daemon a process variation.

A successful incident for anomaly based IDS occurred in the detection of W32.Blaster [25]. The inspection on the source and destination ports of the network traffic revealed the increasing activity on Windows ports. This abnormal activity resulted in an alarm. Further investigations on network logs showed that many IPs were communicating on port 135 to outside. Those IPs were sending SYN packets to a considerable amount of different hosts and same pattern was observed in the traffic to the inside hosts. Later on, a definition of W32.Blaster worm was released on security forums, which was similar to the abnormal network traffic caught by the IDS.

The adaptable behavior of anomaly based IDS enable them to detect undiscovered attack types such as in the W32.Blaster case. On the other hand, the difficulties in the normal usage characterization result in too many false positives. Using more than one individual measure in composing the normal usage profile brings the result aggregation problem. This aggregation is mainly achieved by summing up the

outputs. Comparing the result of this summation to a threshold can mislead the technique. This procedure is the main contributor of the positives since high variation in one measure can lead to an intrusion decision where the other measures do not vary and actually the event is not an intrusion.

Instead of using summation, Bayesian networks are used to aggregate the model outputs and to decrease the amount of false positives. [5], [41], [42], and [43] present IDS which are based on Bayesian Networks. Each different symptom of intrusion is analyzed by individual models and these models are combined in a Bayesian network. The details of Bayesian networks are given in the next section.

### 2.3.3 Hybrid IDS

The hybrid IDS include both signature based and anomaly based IDS characteristics. Currently some proposed hybrid systems exist, but they are still in the research and development phase. Checking signatures as well as the variations from normal profile, results in a more robust defense of the network. In this manner, both the already known type and the undiscovered attacks will be able to be detected, so hybrid models can be much more successful than its counterparts. On the other hand, they bring the excessive investigation burdens by issuing the two techniques together. More research efforts are needed on hybrid IDS.

## 2.4 Bayesian Networks

Bayesian Networks are used to make decision by considering the evidences supplied by various factors. A Bayesian network is a directed acyclic graph (DAG) and models an uncertainty domain. Each node of the DAG represents a discrete random variable and includes the states of the random variable with the conditional probability table (CPT). The relation between the nodes is a parent-child relationship which indicates a causal dependency of the variable of the child node to the variable

of the parent node. The CPT of a node indicates the probability of the node being in a state given the states of its parent nodes.

A Bayesian Network includes two main kinds of variables. They are the information variables whose states can be measured in a straight forward manner and the hypothesis variables whose states can not be obtained directly. The aim of a Bayesian Network is to gather the probabilities of hypothesis variable being in a state according to the evidences supplied by the information variables.

An example Bayesian Network is presented in [5]. There is a bottle of milk that can be either infected or clean. Also there is a test that can determine whether the milk is infected or not. This situation is represented with a Bayesian Network (see Figure 6) including two random variables *positive* and *infected*. The variable *infected* is true when the milk is actually infected and false when the milk is not infected. The variable *positive* is true when the test result indicates that the milk is infected and false otherwise. The conditional probability table for the variable positive is given by test which represents the probability of a positive result given that the milk is infected and the probability of the positive result given that the milk is clean. These two variables are represented as the nodes of the Bayesian Network. The arrow from infected to positive indicates a casual relationship between the variables.

*Prob { positive=1 | infected=1} = 0.99*
*Prob { positive=1 | infected=0} = 0.01*
*Prob { positive=0 | infected=1 } = 0.01*
*Prob { positive=0 | infected=0} = 0.99*

In this network, positive is a information variable whose states can be measured in a straight forward way. On the other hand, infected is the hypothesis variable whose states are derived from the evidences supplied by the information variable (positive) and the corresponding conditional probability table. The probability that the infected

is true can be derived by giving the evidence (positive is true or not) into the Bayesian Network.

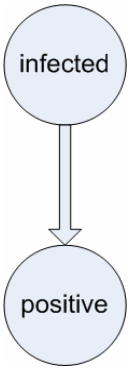


**Figure 6** A simple Bayesian Network [5]

**Table III** Conditional Probability Table of Positive

| | | Infected | |
|---|---|---|---|
| | | True | False |
| Positive | True | 0.99 | 0.01 |
| | False | 0.01 | 0.99 |

# CHAPTER III

# PROPOSED METHODOLOGY FOR P2P TRAFFIC DETECTION

Detecting P2P traffic in the network backbone is not an easy task. The detection techniques other than signature based detection are not accurate. However, signature based P2P detection techniques are useless in backbone application because of the restrictions introduced by the payload inspection. These restrictions include the overhead of intercepting and mirroring the network traffic to investigate the packet payloads at the backbone link speed, and violation of user data privacy. The most significant problem of signature based detection technique is the increasing tendency on the usage of end-to-end encryption in P2P client applications.

Intrusion detection for computer systems and P2P traffic detection for computer networks are very similar problems. Intrusion detection systems incorporate techniques such as collecting system logs and searching for specific patterns through these logs. These techniques can also be used in the detection of P2P traffic.

There are no main differences in P2P and non-P2P traffic characteristics. Therefore, we first characterize the P2P and non-P2P traffic by using the information gathered from flow traces. Since the collection of traffic flow traces is not based on payload analysis, our method overcomes the restrictions of signature based techniques. Then, we identify the variations in P2P and non-P2P traffic characteristics and use them in the detection of P2P contributors.

Our survey on IDS shows that, characterization of P2P and non-P2P traffic is similar to the normal usage profile generation of anomaly based techniques. In addition,

analyzing the features in the characteristics and aggregating the results of these analyses are also similar. Therefore we use a Bayesian Network for aggregating the analyses of the features in P2P and non-P2P characteristics.

## 3.1 Overview of Our Approach

We first characterize the P2P and non-P2P traffic behaviors to detect the P2P contributors through the complete network traffic. The main problem in composing the profiles of P2P and non-P2P traffic is that there are no major differences in P2P and non-P2P traffic characteristics. We define the non-P2P traffic as the normal usage and the P2P traffic as the abnormal one. There exist previous researches that reveal some traffic characteristics which are different for P2P and non-P2P traffic [2], [16]. However, a complete set of differentiating features for P2P and non-P2P traffic does not exist. Some of these discovered characteristics are usage of TCP and UDP ports at the same time, connection duration, and IP packet size distributions. In addition to them, we also look at traffic characteristics such as flow time and IP packet size distribution.

We define *Differentiating Features* as the value or the range of values of these characteristics for which the respective distribution has significantly different values for P2P and non-P2P traffic. The first step in our method is the differentiating feature discovery phase to identify the differentiation features. In this phase, the P2P traffic and non-P2P traffic are distinguished by a signature based P2P detection technique. The output of the signature based technique is a log file which includes the source/destination IP pairs of the P2P contributors. This log file is used to identify flow traces as P2P and non-P2P. The flow traces of both sets are analyzed in detail and the variations in the P2P and non-P2P flows are identified as the differentiating features.

This differentiating feature discovery phase tailors our detection method for a specific network which also contributes to the performance. However, note that the

approach is entirely general and can be applied to any backbone network. It is also possible to start using our method with a given set of differentiating features which could be obtained from another network and then tune these features gradually or even add or subtract features according to the specific traffic of the network. The Differentiating Feature Discovery phase is discussed in detail in Section 3.2.

The differentiating features represent the normal profile as a whole where inspecting more than one characteristic brings the problem of result aggregation. This is also another concern revealed from analysis on the anomaly based IDS. Instead of using summation, Bayesian networks are used to aggregate the outputs of the feature analysis. Some examples of IDS based on Bayesian Networks are presented in [5], [41], [42], and [43]. We also use a Bayesian Network to overcome the disadvantages faced in the result aggregation phase of the anomaly based techniques.

Bayesian networks consist of informational nodes and a hypothesis node. This hypothesis node uses the evidence supplied by the informational nodes to compute the probability of the hypothesis being correct. The informational nodes of our Bayesian Network for P2P traffic detection are the *feature analyzers*. We use a hypothesis node called P2P contributor in the Bayesian network which denotes a traffic flow as P2P or not. The next step is analyzing the differentiating features in the collected P2P and non-P2P data sets to compose the Conditional Probability Tables (CPT) of the informational nodes. The details of CPT compositions are presented in Section 3.3.

In the last phase, we construct our Bayesian P2P Traffic Detection Model with informational nodes and their CPTs. Construction of the model and the resulting Bayesian Network are presented in part 3.4.

Our IDS inspired approach for P2P traffic detection which includes the usage of flow traces is completely novel. In [19], the authors reprogram an intrusion detection software tool SNORT [20] to detect a particular P2P protocol. Their approach is

based on payload investigations and SNORT has to be reprogrammed for every P2P protocol. Our model is an anomaly based detection technique and it is applicable to all P2P protocols. We use network traffic flow traces, so our method overcomes the restrictions introduced by payload analysis. In addition, the Bayesian Network in our model decreases the number of false positives caused by the nature of anomaly based techniques.

In the following sections, we explain the details of the three phases of our approach.

## 3.2 Differentiating Feature Discovery

We set up the test bed in Figure 7 to discover the differentiating features. The link between the Inspected Network's router and the border router is intercepted by a computer with hostname Inspector. In the border router, NetFlow feature [44] is enabled which exports flow information about the routed traffic gathered from IP headers of the traveling packets. The NetFlow data is directed to the Inspector and a General Public License program Flow-Tools [50] is setup in Inspector to capture and store the directed data for further processing. In addition, we setup a signature based P2P detection tool IpP2P [48] which is published under GNU GPL (General Public License) on Inspector to separate out the packets of intercepted traffic with P2P traffic and non-P2P traffic and log their IP addresses in a file. We then parse this log file with awk [49] to get source and destination IP address pairs. These IP address pairs will be used to identify IP flow traces exported by Netflow as P2P or non- P2P. We use Flow-Tools to process the identified flow data and obtain the characteristic distribution. Finally, we extract the differentiating features from these distributions.

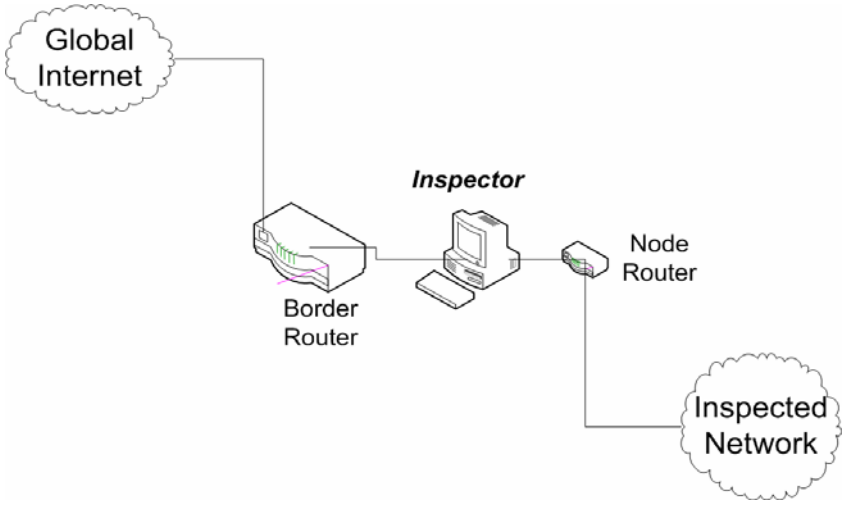


**Figure 7** Testbed used for feature discovery and verification

The link between the UlakNet border router and the Inspected Network's router is intercepted by a computer called *Inspector*. Only the border router and the Inspector are actively included in the feature discovery phase, so the specifications and configurations of these two elements are presented below.

### 3.2.1 Border Router

The border router is of Cisco 12008 series and running 12.0(26) S2 version IOS on it. More than 50 interfaces are configured on this router including Fast Ethernet, Gigabit Ethernet, Asynchronous Transfer Mode (ATM) and Packet over SONET/SDH (POS) technologies. All of the interfaces are configured to export the flow traces which are supplied by NetFlow [44]. NetFlow is a Cisco Internetworking Operating System (IOS) application that provides statistics on packets flowing through the routing devices in the network. It is emerging as a primary network accounting and security technology. NetFlow identifies packet flows for both ingress and egress IP packets. NetFlow does not require any change externally either to the packets themselves or to any networking device. NetFlow is completely transparent to the existing network,

including end stations, application software and network devices such as LAN switches. Also, capturing and exporting the flow traces are performed independently on each internetworking device by NetFlow. NetFlow need not be operational on each router in the network.

A *NetFlow network flow* is defined as a unidirectional stream of packets between a given source and destination. The source and destination are each defined by a network-layer IP address, transport layer source and destination port numbers. Specifically, a flow is defined by the combination of the following seven key fields:

- IP source address
- IP destination address
- Source port
- Destination port
- Layer 3 protocol type
- Class of Service
- Router or switch interface

In the configuration of border router, the IP address of the Inspector is defined as the destination of the exported Netflow data. Therefore, the border router continuously exports the flow traces of the traffic packets to the Inspector.

### 3.2.2 The Inspector

The computer with hostname *Inspector* has a Central Processing Unit (CPU) of Pentium 4 running with 2400 Mhz. The Inspector has 240 Gbyte storage area, 1 Gbyte Random Access Memory (RAM) and two gigabit Ethernet cards installed on it. The Inspector is running Debian [45] on 2.6 Linux kernel as the operating system.

The Inspector is used for two different purposes in our experiment. The Inspector stores the NetFlow data and processes it as a first job. Secondly, it employs the

signature based P2P detection tool to identify the P2P and non-P2P flows. The details of these two purposes are given below.

#### 3.2.2.1 Signature Based P2P Detection Tool

The Inspector functions as a signature based P2P detection tool. A deep packet inspector IpTables [46] is setup on the Inspector. The IpTables is developed in the Netfilter [47] packet filtering framework. The available usages of Netfilter are:

- building internet firewalls based on stateless and stateful packet filtering
- using Network Address Translation (NAT) and masquerading for sharing internet access if you don't have enough public IP addresses
- using NAT to implement transparent proxies
- aid the traffic control (tc) and iproute2 systems used to build sophisticated QoS and policy routers
- do further packet manipulation (mangling) such as altering the TOS/DSCP/ECN bits of the IP header

The IpTables/Netfilter architecture is extended to identify P2P data in the IP traffic by another project called IpP2P [48]. IpP2P uses suitable search patterns to identify P2P traffic thus allowing the reliable identification of traffic belonging to many P2P networks. Once identified, one may handle P2P traffic in different ways - dropping such traffic, putting into low priority classes or shaping to a given bandwidth limit is possible.

**Table IV** IpP2P Options and Detection Rating [48]

| Option | P2P network | Protocol | Quality |
|---|---|---|---|
| --edk | eDonkey, eMule, Kademlia | TCP and UDP | very good |
| --kazaa | KaZaA, FastTrack | TCP and UDP | good |
| --gnu | Gnutella | TCP and UDP | good |
| --dc | DirectConnect | TCP only | good |
| --bit | BitTorrent | TCP and UDP | good |
| --apple | AppleJuice | TCP only | (need feedback) |

Since we intercept the target network traffic, we easily apply IpP2P which is published under GNU GPL (General Public License) to detect the P2P traffic contributors (see Table IV). The Inspector is functioning as a bridge in our experiment. There exist three sets of rules, called chains, in the "filter table" of a kernel. The filter table is set of rules which are applied to the packets processed by the kernel. You can add rules to these chains by using the IpTables. The journey of a packet after entering from one Ethernet card till leaving from the other Ethernet card is given in Figure 8 for a general IpTables application.

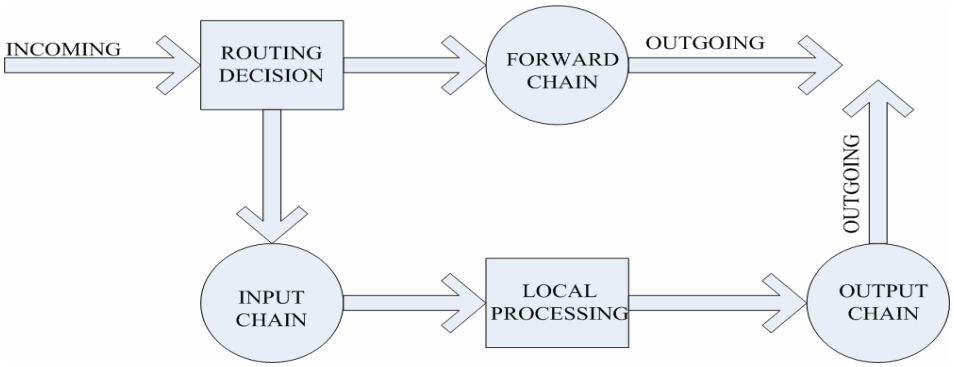


**Figure 8** IpTables Packet Journey

1. When a packet comes in (for example through the Ethernet card) the kernel first looks at the destination of the packet: this is called `routing'.
2. If the packet is destined for this box, the packet passes downwards in the diagram, to the INPUT chain. If it passes this, any processes waiting for that packet will receive it.
3. Otherwise, if the kernel does not have forwarding enabled, or it doesn't know how to forward the packet, the packet is dropped. If forwarding is enabled, and the packet is destined for another network interface (if you have another one), then the packet goes rightwards on our diagram to the FORWARD chain. If it is ACCEPTed, it will be sent out.
4. Finally, a program running on the box can send network packets. These packets pass through the OUTPUT chain immediately: if it says ACCEPT, then the packet continues out to whatever interfaces it is destined for.

We do not use the input and output chains of IpTables since no packets exist for the box Inspector (Step 2). Therefore, all of the packets traverse the FORWARD chain in the Inspector. In addition, to enable the usage of IpP2P module in IpTables two more chains are defined: Pre-routing traversed before the forward chain and post-routing traversed after the forward chain. When IpP2P detects a P2P pattern in the packet payload in pre-routing chain, the packets can be accepted, dropped or marked for later actions. In our experiment, we mark the P2P packets in the pre-routing chain by the following configuration:

***1#*** *iptables -t mangle -A PREROUTING -j CONNMARK --restore-mark*
***2#*** *iptables -t mangle -A PREROUTING -m mark ! --mark 0 -j ACCEPT*
***3#*** *iptables -t mangle -A PREROUTING -m ipp2p --edk -j MARK --set-mark 1*
***4#*** *iptables -t mangle -A PREROUTING -m ipp2p --dc -j MARK --set-mark 2*
***5#*** *iptables -t mangle -A PREROUTING -m ipp2p --gnu -j MARK --set-mark 3*
***6#*** *iptables -t mangle -A PREROUTING -m ipp2p --kazaa -j MARK --set-mark 4*
***7#*** *iptables -t mangle -A PREROUTING -m ipp2p --bit -j MARK --set-mark 5*
***8#*** *iptables -t mangle -A PREROUTING -j CONNMARK --save-mark*

First all of the packets entered to this chain are marked as 0 (Step 1). The second step ensures that an already marked packet won't get marked again. Then the P2P packets are marked with the corresponding protocols identifier (Steps 3-6). After the analysis of the packet payloads is finished, then the "later action" is defined:

***9#*** *iptables -t mangle -A POSTROUTING -m mark --mark 1 –j LOG*
***10#*** *iptables -t mangle -A POSTROUTING -m mark --mark 2 -j LOG*
***11#*** *iptables -t mangle -A POSTROUTING -m mark --mark 3 -j LOG*
***12#*** *iptables -t mangle -A POSTROUTING -m mark --mark 4 -j LOG*
***13#*** *iptables -t mangle -A POSTROUTING -m mark --mark 5 -j LOG*

The marked packets are logged to the default kernel log file /var/log/messages with the configuration in steps 9 – 13. After intercepting the target networks traffic with the above configuration we have lines in the log file as follows:

***March 22 10:04:36*** *p2p kernel: p2p searchIN=br0 OUT=br0 PHYSIN=eth1 PHYSOUT=eth0 SRC=**aaa.aaa.aaa** DST= **xxx.xxx.xxx.xxx** LEN=341 TOS=0x00 PREC=0x00 TTL=126 ID=21435 DF PROTO=TCP SPT=1130 DPT=6349 WINDOW=16620 RES=0x00 ACK PSH URGP=0*

***March 22 10:04:36*** *p2p kernel: p2p dataIN=br0 OUT=br0 PHYSIN=eth1 PHYSOUT=eth0 SRC=**bbb.bbb.bbb.bbb** DST= **yyy.yyy.yyy.yyy** LEN=341 TOS=0x00 P REC=0x00 TTL=126 ID=21435 DF PROTO=TCP SPT=1130 DPT=6349 WINDOW=16620 RES=0x00 ACK PSH URGP=0*

***March 22 10:04:49*** *p2p kernel: p2p searchIN=br0 OUT=br0 PHYSIN=eth1 PHYSOUT=eth0 SRC=**ccc.ccc.ccc.ccc** DST=**zzz.zzz.zzz.zzz** LEN=413 TOS=0x00 PREC=0x00 TTL=126 ID=21657 DF PROTO=TCP SPT=1130 DPT=6349 WINDOW=16817 RES=0x00 ACK PSH URGP=0*

The log file includes the IP Header information of P2P packets. We don't drop the P2P packets, just log them since we want the P2P traffic to be realized and cause flow traces in the border router. The source/destination IP pair and the time of the traffic information will be enough to find the corresponding flow trace of P2P traffic.

Therefore we process the huge log file with a simple command line awk [49] script and pick the required information.

```
cat /var/log/messages" | grep p2p | awk ' {print $3"\t"$11"\t"$12}' | sort -u >> OUTPUT
```

The output of the awk script is used in the processing of the NetFlow data to distinguish the P2P and non-P2P flows.

#### 3.2.2.2 Storing and Processing the Netflow Data

As mentioned at the beginning of this chapter, the border router is configured to export the NetFlow data to the Inspector by specifying its IP address. An "*Open Source Initiative (OSI) Approved – Berkeley Source Distribution (BSD)"* licensed software, Flow-Tools [50] is used to store and process the NetFlow data exported by the border router. The Inspector is programmed to store the NetFlow data into files including the traces of 5-minute traffic. The flow traces aggregated into a file every five minute by the flow-capture command. Every 5-minute flow data has approximately 30 megabytes size in day time and 20 megabytes size in night time.

The processing of the flow data is also done by using flow-tools commands. *Flow-cat* enables us to aggregate the 5-minute flow files for further processing. *Flow-filter* filters the aggregated data according to source or destination IP addresses, Autonomous System (AS) numbers, interface identifiers, Layer 4 port numbers and some other criteria. *Flow-stat* gives 20 different types of statistics about the aggregated and filtered data. Mainly these three commands are used for the processing of the NetFlow data in this work.

We end up with a log file including the source and destination IP pairs and the time of the P2P traffic after inspecting the target network traffic by IpP2P for a week. Later, we aggregate the NetFlow data corresponding to that week with flow-cat. The aggregated data is then filtered by flow-filter. The filtering is done according to the

interface identifier of the target network, since the NetFlow data also includes the traces belonging to other networks. We also filter the aggregated data with two access-lists.

The first access-list is used to identify the P2P traffic flows. Only the IP pairs which exchanged a P2P packet and were logged by IpP2P are permitted in the access list. This access-list is called *P2P.acl*.

The other access-list is used to identify the non-P2P flows. For this access-list we made an assumption. *None of the source-destination IP pairs involved in a P2P communication also involve in a non-P2P communication.* A non-P2P communication forces source or destination host to be traditional servers. Since setting up a P2P client on a traditional server such as FTP, SMTP or HTTP can occur rarely, we define all the traffic between an IpP2P logged IP pair as P2P. Therefore in the second access-list we deny all the traffic between logged IP pairs and permit others. This access-list is called *non-P2P.acl*.

We gather statistics about the flows after filtering them as P2P and non-P2P by flow-stat. The flow summary option of the flow-stat command includes the following distributions:

- *Packet size distribution (ps)*
- *Packets per flow distribution (pf)*
- *Octets per flow distribution (of)*
- *Flow Time distribution (ft)*

The listed distributions can be gathered with a single processing of the flow traces. A sample output of flow-stat flow summary is given in Appendix C which shows the various distributions for a group of flows.

These distributions are gathered for the target network's whole traffic flow traces, P2P traffic flow traces and non-P2P traffic flow traces. The four distributions are

plotted in to graphs to visualize the variations of P2P and non-P2P traffic characteristics. The resulting graphs are given in the Figures 9 - 12.

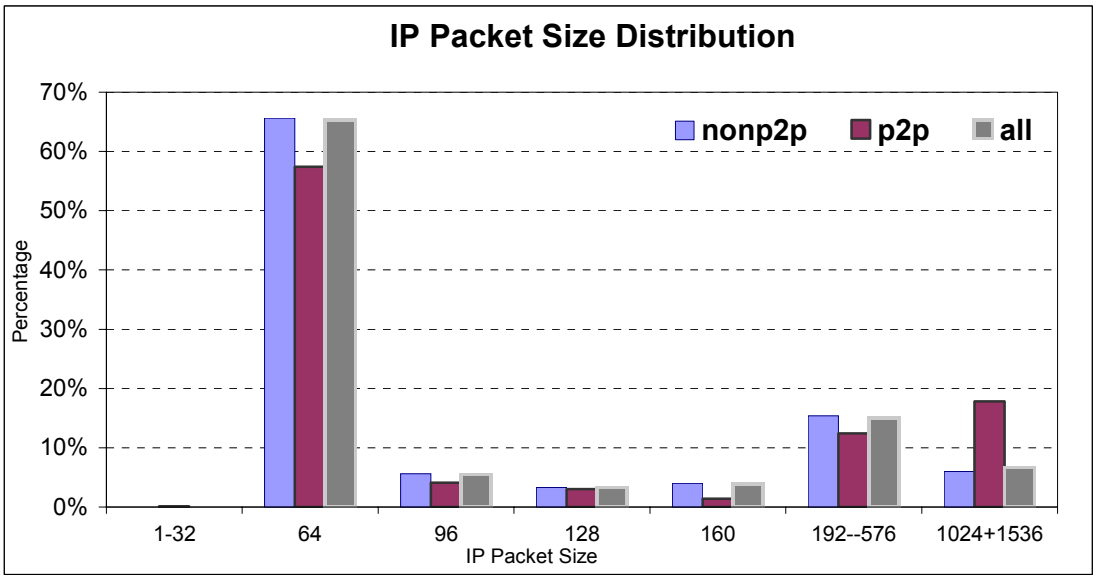


**Figure 9** IP Packet Size Distribution

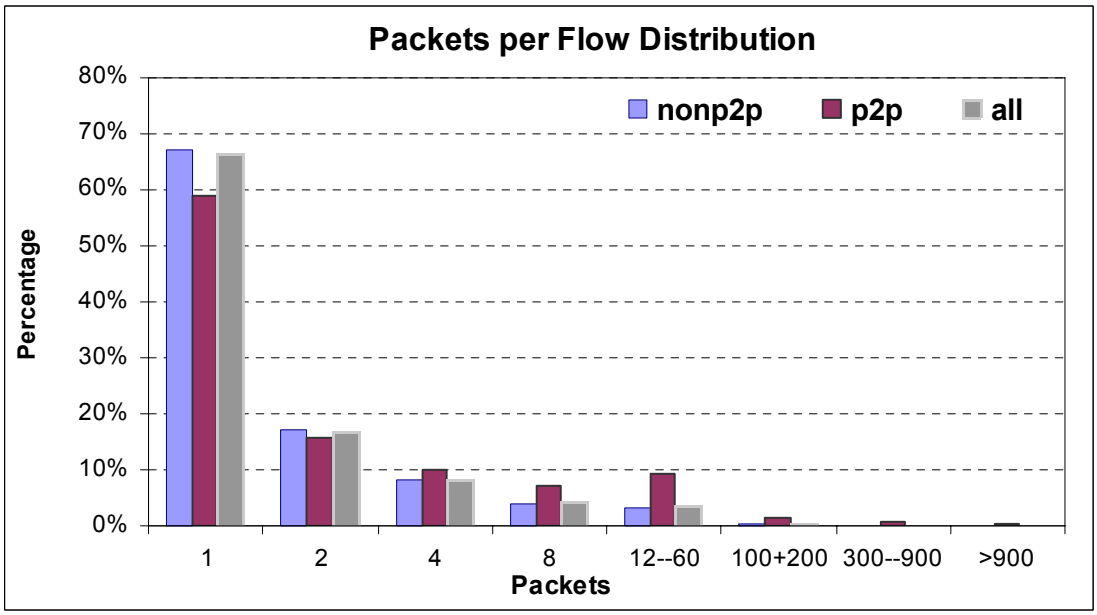


**Figure 10** Packets per Flow Distribution

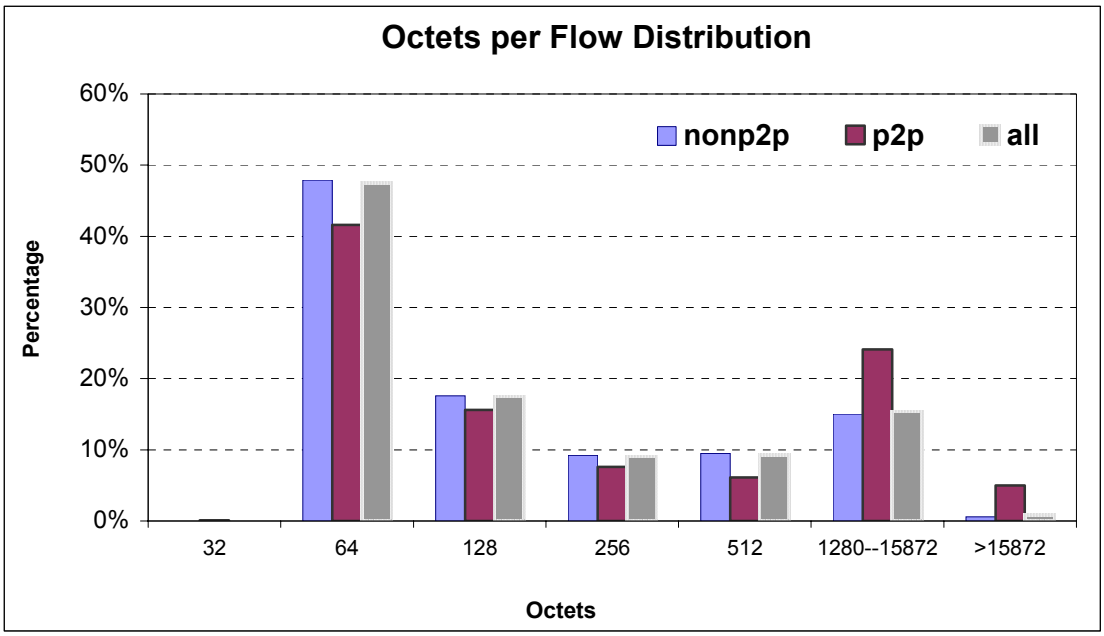


**Figure 11** Octets per Flow Distribution

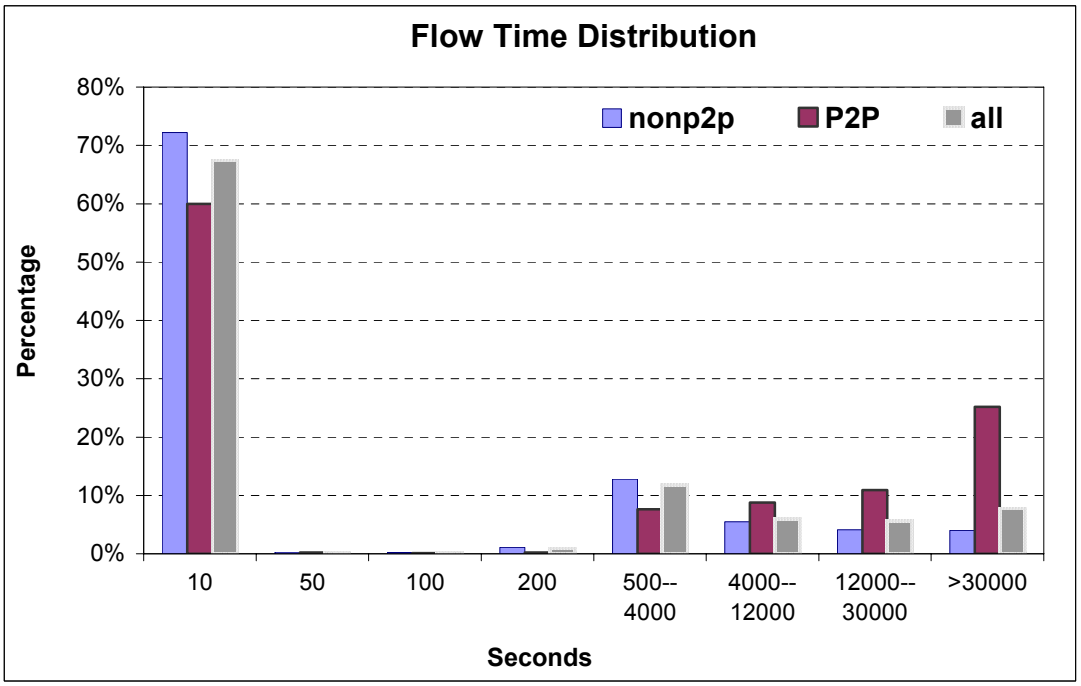


**Figure 12** Flow Time Distribution

Some of the values in the distributions are directly included in the graphs, some are represented with intervals and some include value set. As an example, the values 64, 94 and 128 bytes in the IP packet size distribution are directly included in the graph. 192—576 represent an interval of packet size values between 192 and 576. Finally 1024+1536 represents a set including the packet size values 1024 and 1536 for IP packet size distribution.

We calculate the differences of the percentages of P2P and non-P2P traffic for each value in the distributions and this is repeated for the all four distributions. The difference for a value of a distribution is compared to the differences of other values in the distribution and the values that differ significantly between P2P and non-P2P are selected as differentiating features. The percentage of flows with the number of packets between 12 and 60, and the percentage of the packets with sizes 1024 and 1536 are examples for differentiating features. The variation in the percentage of flows with the number of packets between 12 and 60 is 6% for P2P and non-P2P traffic where the differences for the other values of distribution of packets per flow are between 1.4% and 3%. Similarly, difference of percentage of packets with sizes 1024 and 1536 are 11.8% where the differences for the other values in distribution of number of packets are between 0% and 4%.

We do not look at the absolute difference between the values of the characteristics for only three values in three different distributions. They are the percentage of flows with one packet, the percentage of flows with flow time less than 10 milliseconds and the percentage of the packets with size 64 Bytes. The differences of P2P and non-P2P traffics are 8%, 12% and 8% respectively. The selected differentiating values for these distributions have 7%, 21% and 11.8% where the other values are between 0% to 3%, 0% to 5% and between 0% to 4% respectively. On the other hand, these three values occupy the 60% to 70% of the flows for P2P and non-P2P traffic where all the other values have a 40% to 30% gap to vary. Therefore, we do not select these values of the distributions as differentiating features.

The following values or range of values are discovered as the variations of P2P and non-P2P traffic characteristics:

$d_1$: ***ps*** = 1024 *and* ***ps*** = 1536
$d_2$: 12 ≤ ***pf*** ≤ 60
$d_3$: 1280 ≤ ***of*** ≤ 15872
$d_4$: ***of*** ≥ 15872
$d_5$: ***ft*** ≥ 30000

These five characteristics are identified as the *differentiating features* by the analysis of the graphs of distributions. These features are mostly *transferred data size* oriented and easily analyzable. The *transferred data size* approach serves for detecting the heavy hitters of P2P traffic. Note that we focus on detecting the heavy hitters because of the bandwidth consumption concerns.

As previously mentioned, there exist previous efforts on characterizing the P2P traffic. We analyzed the revealed features and used two of them in our method which need more analysis on flow data. First one is *both TCP/UDP ports usage* (**tu**) [16]. The Domain Name queries (UDP port 53) are not counted as a hit to the UDP ports since every host generates such traffic. Next, we consider the number of IP addresses communicated for each host in the network. Rather than using this number directly, we use the ratio of the distinct IP addresses connected to each host to the total number of IP addresses connected by all of the hosts in the network. We call this differentiating feature *connection metric* (**cm**). These features are also considered in [2] and [16]. Note that, we look at the traffic volume characteristics in more detail than [2].

$d_6$: ***tu***
$d_7$: ***cm***

In addition, we also include the *Transport Layer port* (**tp**) information of in our method. Usage of the well known P2P ports and well known service ports [52] also listed as the *differentiating features.*

$d_8$: ***tp*** $\in$ {21, 25, 80, 443, 8080}
$d_9$: ***tp*** $\in$ {411, 1214, 4672, 6699, [4660–4669], 5661, 5662, [6881–6889]}

A flow diagram which represents the Differentiating Feature Discovery phase is given in Figure 13.

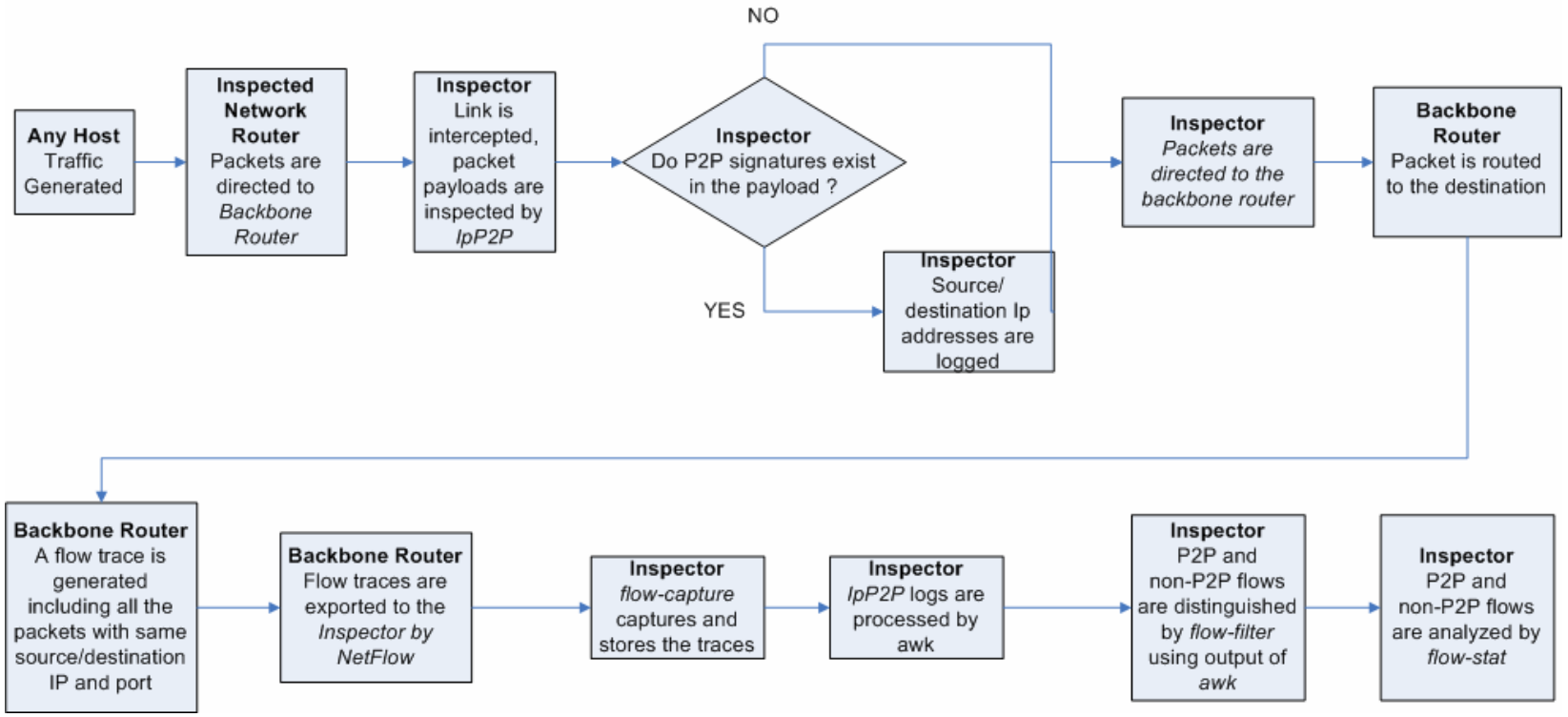


**Figure 13 Differentiating Feature Discovery**

## 3.3 Composition of Conditional Probability Tables

The composition of CPTs is achieved by statistical analysis. The nine differentiating features are analyzed for a group of P2P and non-P2P traffic contributors. The distributions of the values of the differentiating features are used to compose the Conditional Probability Tables.

The *on-time* value of hosts is defined as the average connection duration to a P2P network. Previous work on on-time values [2] shows that around 60% of P2P contributors do not stay longer than 10 minutes in P2P network. In addition, cumulative distributions of IP addresses show similar behavior for 20 and 30 minute investigations. This shows that the traffic characteristics of a P2P contributor host become stable after 20 minutes. We assume that a maximum of one hour is feasible to identify an IP address as P2P contributor or not. Consequently we compose our dataset from 5-minute, 15-minute, 30-minute and 60-minute traces. In addition, heavy hitters are on line for 24 hours and P2P networks have users from all over the world from different time zones. Therefore, the selected flow traces are collected at different times of the day and distributed in a uniform manner. We use flow traces of the network traffic gathered during a week.

200 different IP addresses are selected from the uniformly composed datasets. All of these IP addresses are picked from the *heavy hitters list*. The heavy hitters list includes the top 20% IP addresses ordered according to the volume of the traffic they generate. There are 50 flow traces from each time period type including 5-minute, 15-minute, 30-minute and 60-minute traces which sum up to a total of 200 flow traces in the dataset. One single IP is selected from every trace. As a result, the nine differentiating features are analyzed for 200 unique IP addresses for the corresponding flow traces.

After identifying the datasets we perform the following procedure to compose the CPTS:

For a given differentiating feature $d_i$ we compute the conditional probability values:

$Prob\ \{\ flow\ is\ P2P\ |\ Prob\ \{d_i\} \in S_j\} = P_{P2Pij}$

$Prob\ \{\ flow\ is\ non\text{-}P2P\ |\ Prob\ \{d_i\} \in S_j\} = P_{nP2Pij}$

where;

$I = 1,..., 9,\ \ S_j = [\ S_{JL},\ S_{JH}\ ],\ 0 \le S_{JL} \le S_{JH} \le 1$

CPT for $d_i$ obtained by arranging $P_{P2Pij}$ and $P_{nP2Pij}$ for a set of mutually exclusive $S_j$ where;

$j = 1,\ldots,k\ S_{1L} = 0,\ S_{KH} = 1.$

The resulting conditional probability tables are given in the Figures 14 – 22.

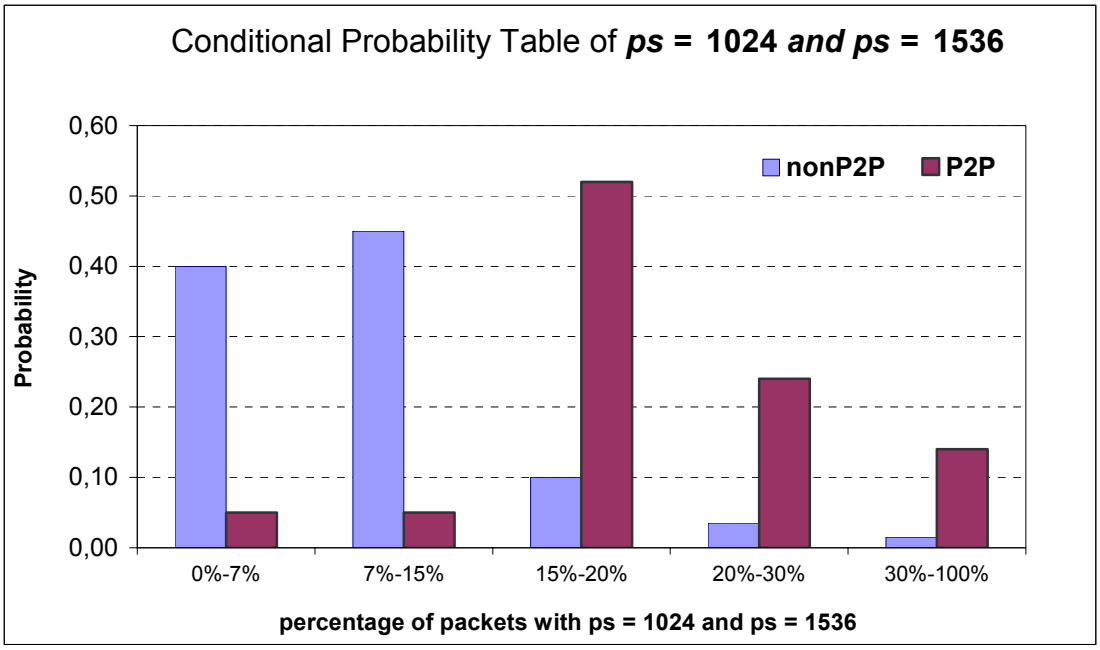


**Figure 14** CPT of **ps = 1024** and **ps = 1536**

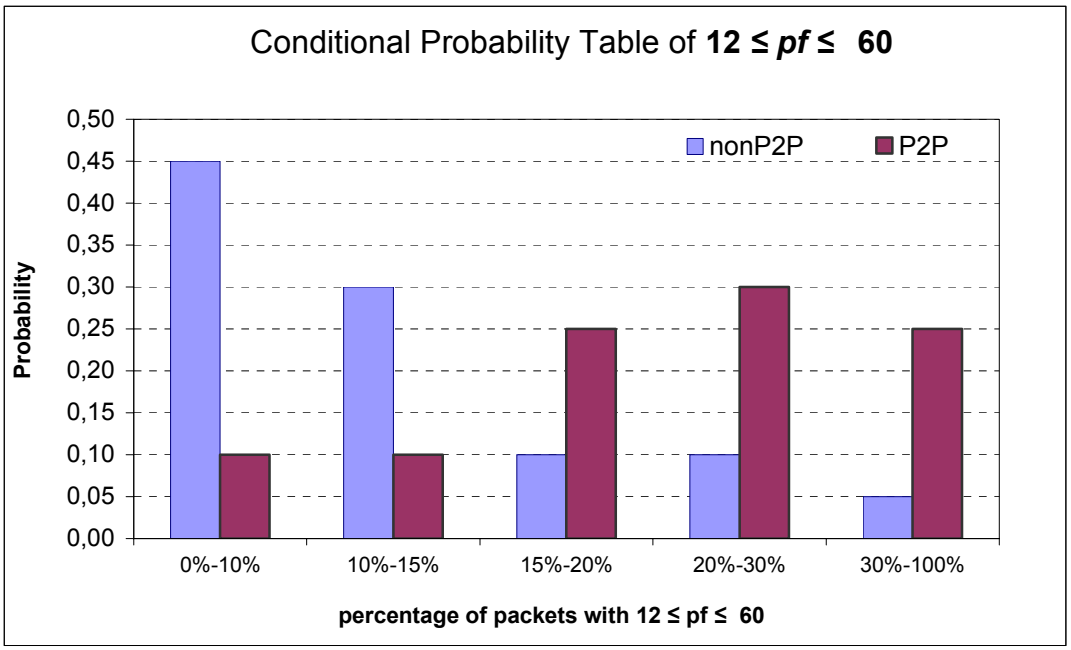


**Figure 15** CPT of **12 ≤ pf ≤ 60**

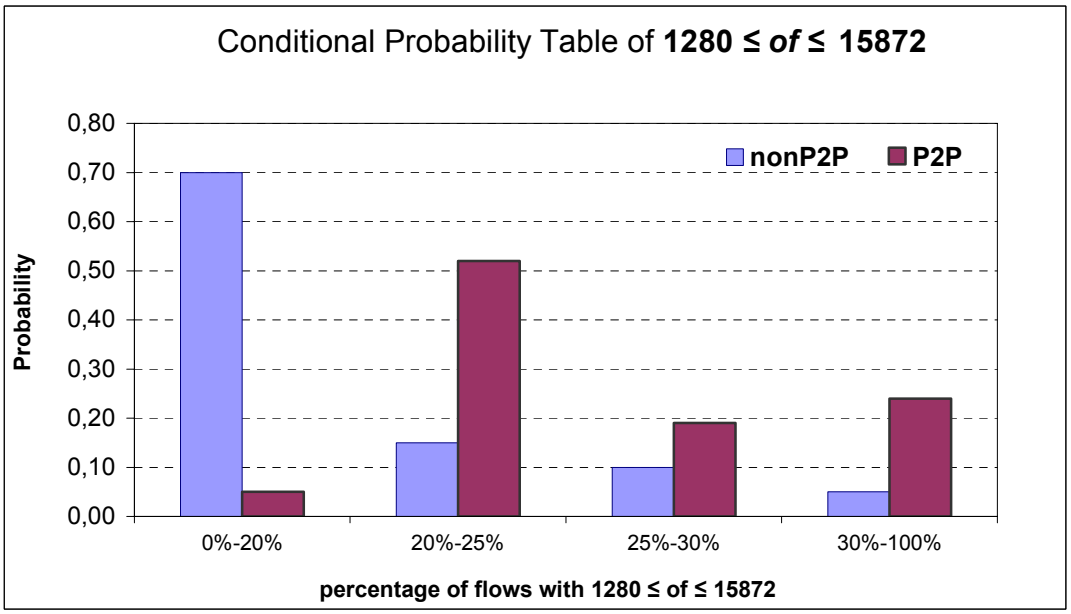


**Figure 16** CPT of **1280 ≤ of ≤ 15872**

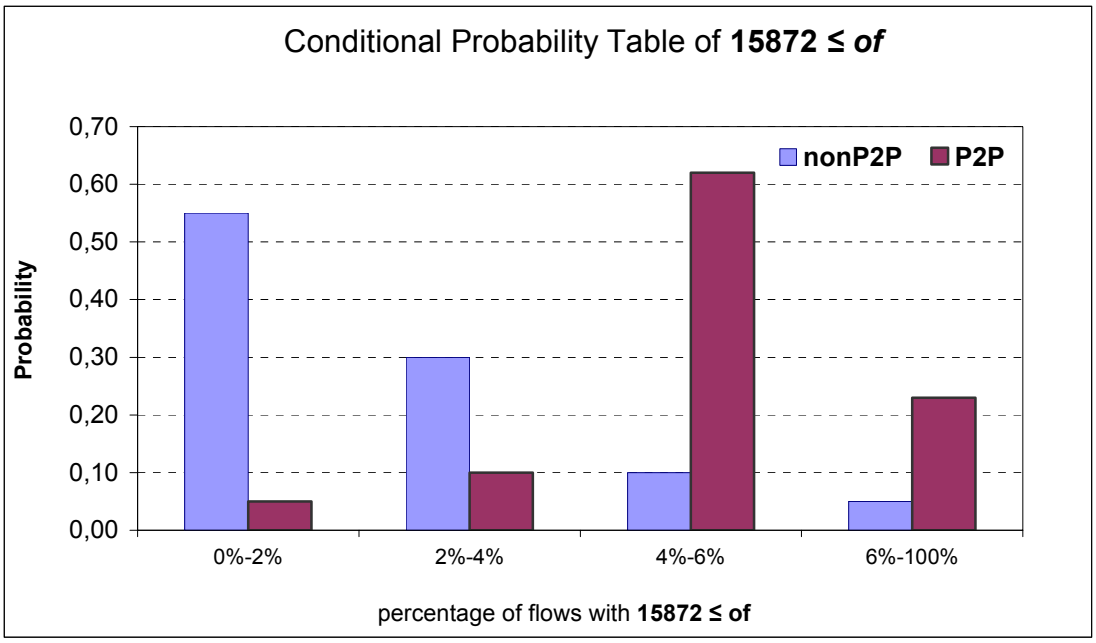


**Figure 17** CPT of **of ≥ 15872**

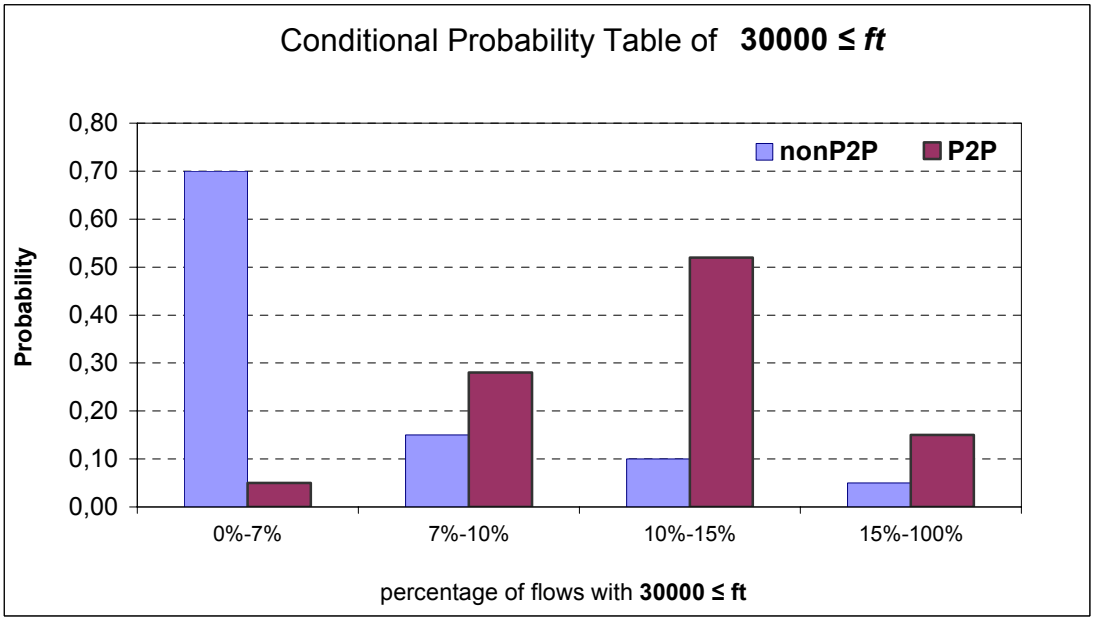


**Figure 18** CPT of **ft ≥ 30000**

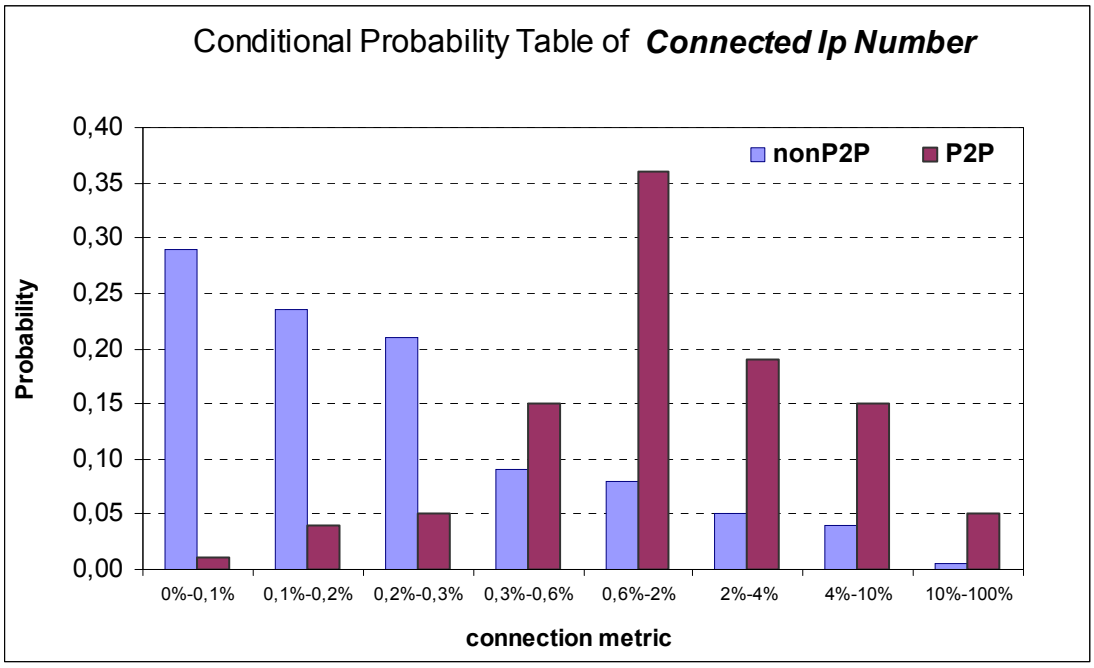


**Figure 19** CPT of **cm**

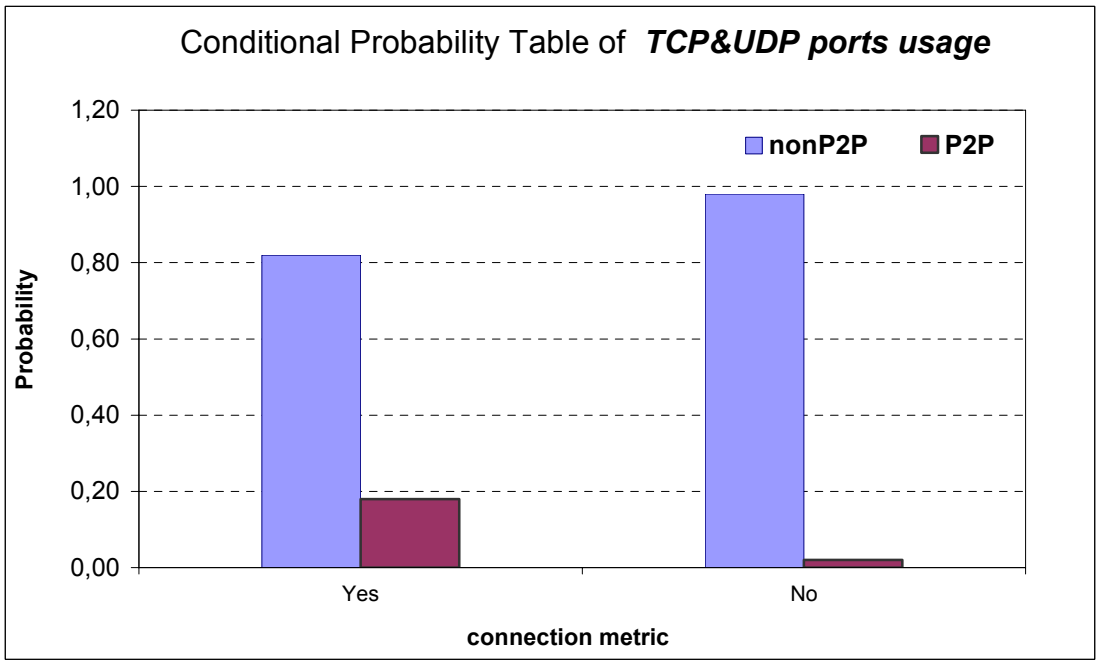


**Figure 20** CPT of **tu**

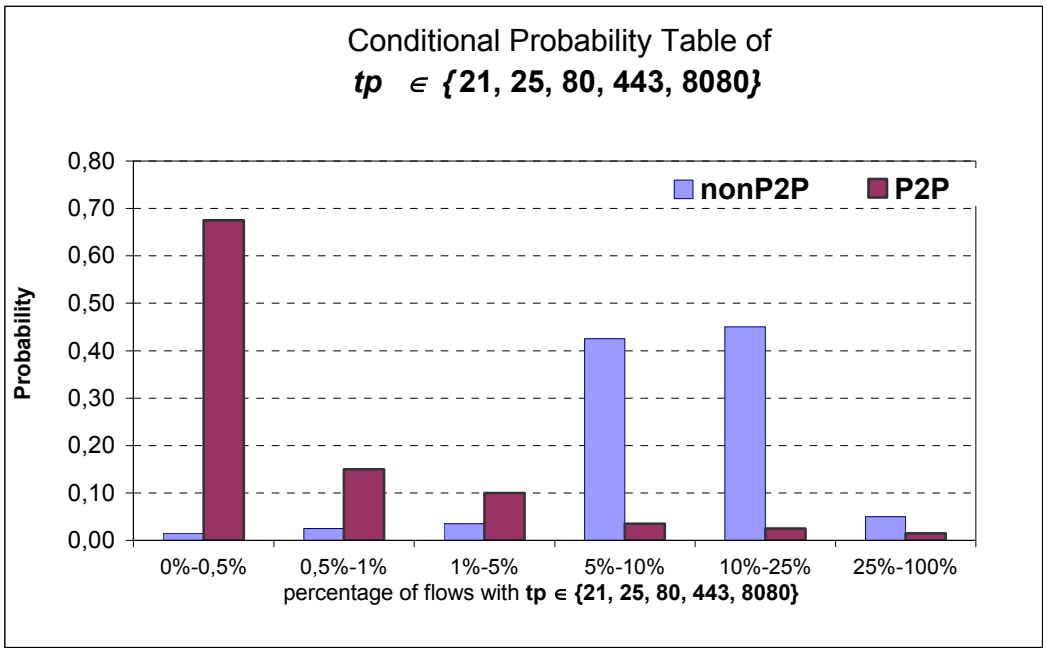


**Figure 21** CPT of **tp ∈ {21, 25, 80, 443, 8080}**

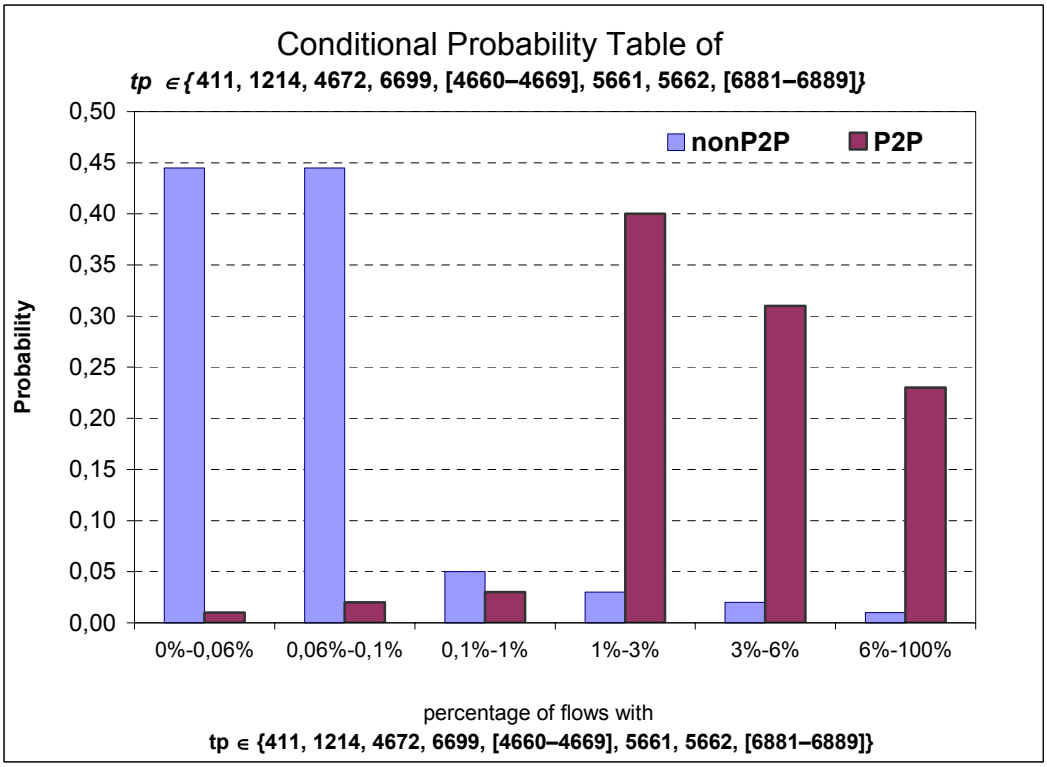


**Figure 22** CPT of **tp ∈ {411, 1214, 4672, 6699, [4660–4669], 5661, 5662, [6881–6889]}**

## 3.4 Realization of the Bayesian Network

After deciding the informational nodes and composing their Conditional Probability Tables, the next step is to setup the Bayesian Network. We use Microsoft Bayesian Network Editor and Toolkit (MSBNx) [51] to realize our network. MSBNx is a component-based Windows application for creating, assessing, and evaluating Bayesian Networks, created at Microsoft Research.

All the identified differentiating features are represented as informational nodes. The CPTs' values are entered into MSBNx. For an investigated IP address, the differentiating feature presented by an informational node is analyzed by Flow-Tools from the flow traces for an interested time period. The resultant distribution value or range values are fed into the Bayesian network. This procedure is repeated for all of the informational nodes. The hypothesis node, P2P Contributor outputs the probability that the given IP address is involved in P2P communication in the investigated time period. The resulting Bayesian Network is given in the Figure 23. Part of this research has been presented in [53].

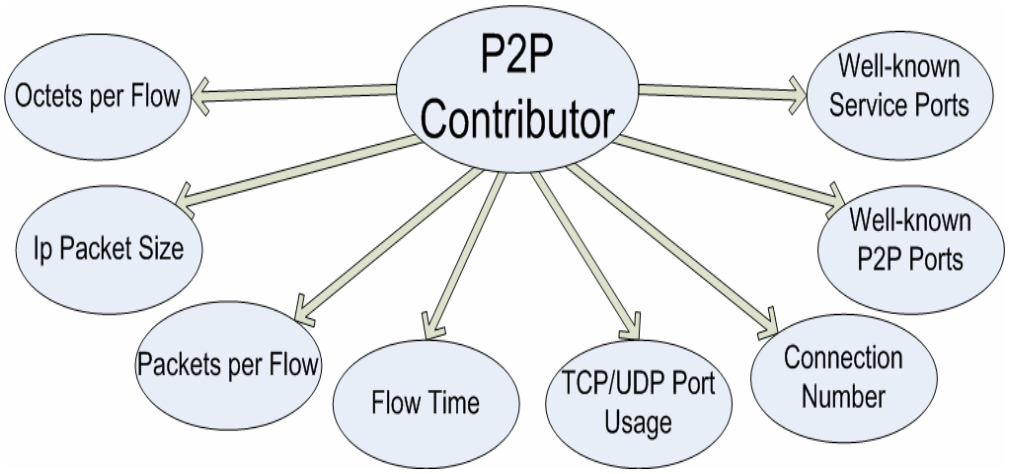


**Figure 23** Proposed Bayesian Network for P2P Traffic detection

# CHAPTER IV

# RESULTS AND EVALUATION

We apply our Bayesian Network model for detecting P2P traffic contributors in two of the nodes of Turkish Academic Network. We used flow traces collected from a border router for a week. The size of this data is 60 GigaBytes. It takes 30 seconds to process 5 minutes flow data with our filtering rules in Flows-Tools. These filtering rules include port, source-destination IP pairs and interface identifiers. Among many problems of P2P traffic, bandwidth consumption is the most important one for ULAKBIM since the global internet uplinks of UlakNet are heavily utilized as seen in Figure 2. Hence, we order the IP addresses according to the volume of traffic they generate and pick the top 20% to find the ”heavy hitters”. Then we identify these “heavy hitters” as P2P contributor or not by using our Bayesian Network.

We use signature based detection techniques for verification issues in both nodes. The first node is the same one which is included in the test bed composed to develop our method. The target network is still under inspection of the IpP2P embedded IpTables system. The IpTables output log file includes the IP addresses of the P2P contributors and is used to verify our method. The data used for verification is not the same one used for composition of the conditional probability tables. We gathered another one week data to verify our method.

The second node is a university network which uses Cisco’s NBAR application to detect P2P contributors. The P2P packets are marked at the node’s router by NBAR. We log the marked packets by an access control list at backbone router. This log is similar to the output of IpP2P and includes the P2P contributor IP addresses.

We compare the decisions of our Bayesian Network for every tested IP address to the log files which are the outputs of signature based technique. Results show that the model detects P2P traffic with 10% false negatives and 17% false positives (see Figures 24-25). The output of the hypothesis node in our Bayesian Network shows the probability of the tested flows belonging to a P2P communication. This probability is over 90% percent in the successful detections of our method.

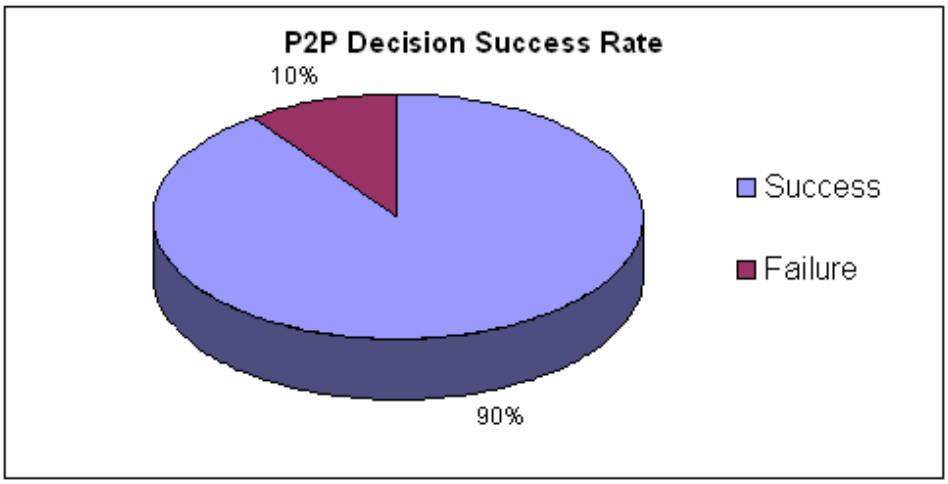


**Figure 24** Success rate of P2P Contributor decision

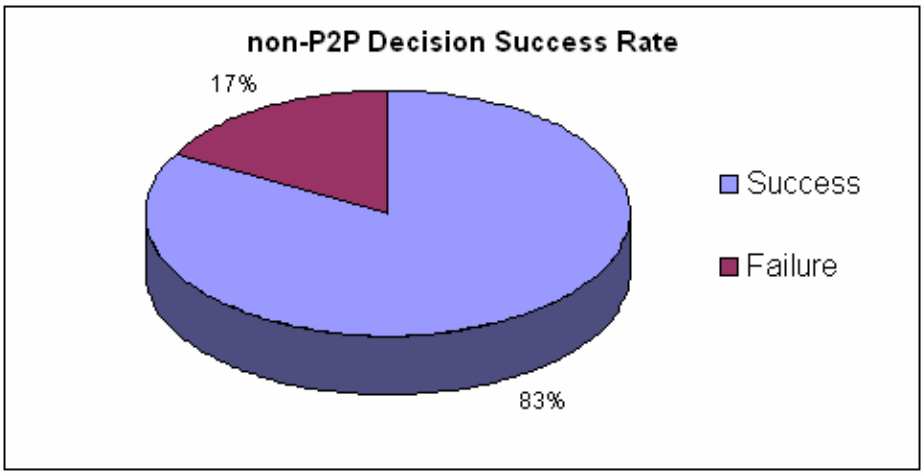


**Figure 25** Success rate of not P2P Contributor decision

We performed an analysis on the false negatives and false positives of our method. As we described previously, the differentiating features (informational nodes) supply evidence to produce the decision of the hypothesis node. Hence, for a given flow each differentiating feature contributes to the final decision. For each differentiating feature, we found the ratio of the IP flows for which the supplied evidence correctly identifies them as P2P contributors or not (See Figure 26).

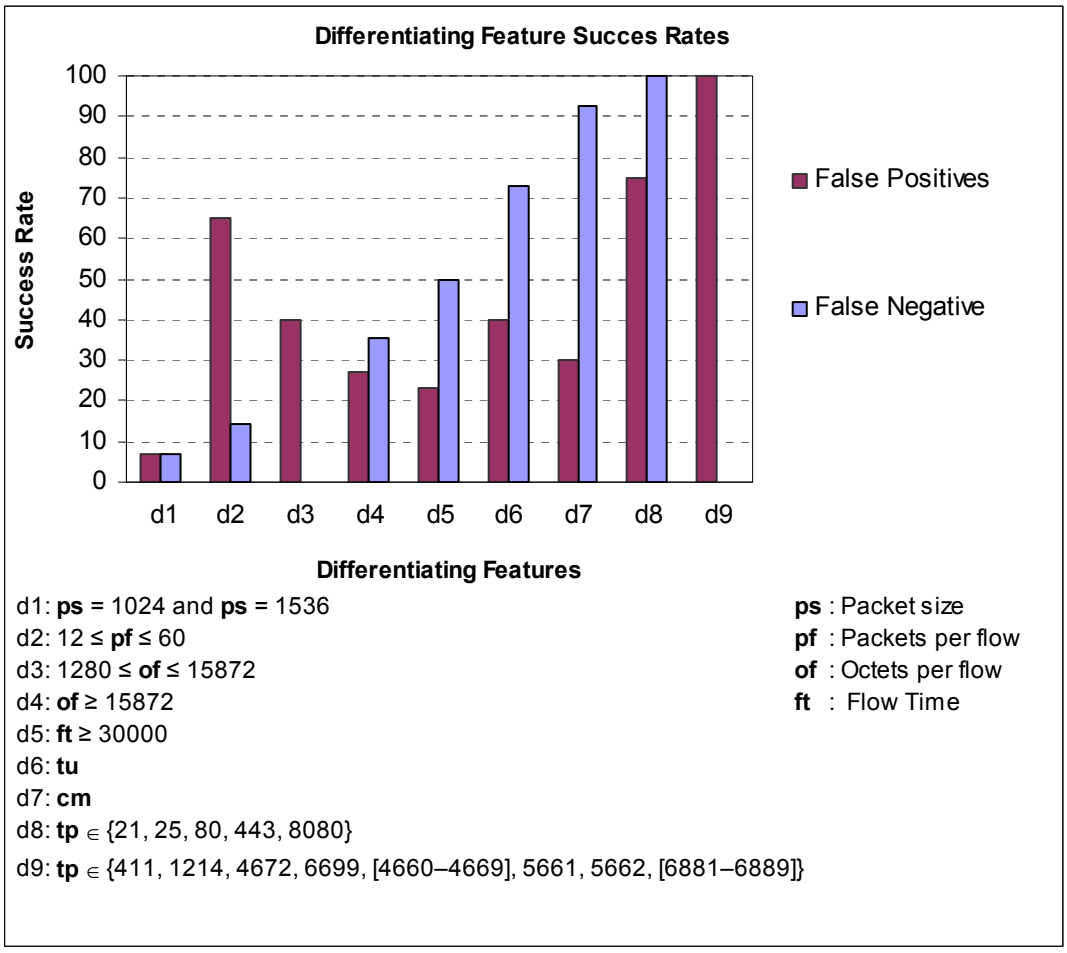


**Figure 26** Success rates of *Differentiating Features*

As seen in Figure 26 , $d_3$ and $d_9$ **could not detect** any of the missed P2P contributors. And also the success rates of $d_1$ and $d_2$ are very low. On the other hand, $d_5$ and $d_8$ successfully identify the P2P contributors.

14% of our missed P2P contributors have byte transfer less than 2% of the average of the entire flow set. The number of packets they transfer is less than 15% of the average of the entire flow set. These IPs are not actual heavy hitters, however, they are producing high enough amount of traffic with respect to the low network load at the moment. Hence, our model designed to detect the heavy-hitters fail to detect these IPs. These non-actual heavy hitters are the main reason for the failure of the $d_1$, $d_2$ and $d_3$ which are mainly based on the high amount of packets and data transferred. The reason for the failure $d_9$ is the dynamic port usage of the P2P applications which is expected.

The success rates of the *differentiating features* for the incorrect P2P contributor decisions (false positives) are also given in the Figure 26. The success rates of $d_1$, $d_4$, $d_5$ and $d_7$ are relatively low which means that these features of the incorrectly identified flows are similar to the P2P traffic behavior.

$d_1$, $d_4$, and $d_5$ are the main features that represent large amount of data transfers which is the main reason for the failure of these characteristics in false positive detections. The protocols resulting in high amount of data transfers and not listed as the well-known protocols such as grid applications and online games can mislead these features. Same protocols mislead $d_7$ since the data transferring hosts open considerable amount of connections.

# CHAPTER V

# CONCLUSION AND FUTURE WORK

In this thesis, we present our intrusion detection inspired approach for the detection of P2P traffic in the backbone. Currently, the most accurate P2P detection methods look for specific P2P application signatures in the IP packet payload. The signature based detection methods have scalability problems to work at backbone rates. Also, inspection of payloads violates the user data privacy. More importantly, these methods are useless for the encrypted data.

We use flow traces collected from the routers and use the variations in the traffic characteristics of P2P and non-P2P traffic, so our method does not suffer from the limitations of signature based methods. Previous researches also included analysis of flow traces for P2P detection to some degree. We use the features revealed from the previous researches and also discover our own features to differentiate P2P and non-P2P traffic.

We use a signature based P2P detection tool to mark and distinguish P2P and non-P2P flows for further investigations. This tool is used only once in the development phase of our method. After distinguishing the flows, the variations in characteristics of P2P and non-P2P traffic are identified as *Differentiating Features*. These features are analyzed from traffic flow traces for an IP address under investigation. The results of the analyses on the discovered features are aggregated by a Bayesian Network. The Bayesian Network detects P2P contributors from flow traces with 10% false negative 17% false positives.

Our P2P traffic detection method is scalable; it can be applied to any backbone and detects all of the current P2P protocols. We use an anomaly based technique, so it is an adaptable one which will detect unknown P2P protocols. On the other hand, major changes in P2P communication resulting in variations of the traffic characteristics will decrease the success rate of our method. However, constructing the Bayesian Network with the differentiating features of specific network identified by using up-to-date P2P protocols would improve our method's performance.

We used network data collected from two nodes of National Academic Network not an artificial test data. Therefore, we have a good performance on characterizing P2P and non-P2P traffic. In addition, using the network data collected from an academic network enhanced our method. The users in an academic network are more curious and they have high level skills in information sciences when compared to the user of any other ISP. Therefore, it is strongly probable that there exist traffic flows belonging to all of the current P2P protocols in the network traffic of academic networks.

In addition to the technical problems of intercepting the backbone links, our method has various advantages compared to the signature based techniques which intercept the network traffic. First, our method does not introduce any delay to the packets where inspecting the packet payload does. Second, any service fault on the host accommodating the signature based tool can result into traffic hit. Finally parallel computation methods can be applied to analyze flow traces which speeds up our method. On the other hand, payload analysis can not be performed by parallel computation with interception method. Some signature based techniques need mirroring of the traffic. The huge storage for this mirroring is the main limitation for those techniques. Our method overcomes this limitation since we use the flow traces supplied by the routers. A comparison of our method with signature based detection techniques is given in Table V.

**Table V** Comparison of our method with signature based detection techniques

| | **Our Method** | **Signature Based Methods** |
|---|---|---|
| **Evidence** | Flow Traces | Packet Payload |
| **Evidence Collection** | Supplied by routers | Either traffic is intercepted or mirrored |
| **Delay introduction to traffic** | None | Delay is introduced during payload analysis in interception method. |
| **Traffic hit probability** | None | In interception method any fault in signature based tool causes traffic hit |
| **Parallel Computation** | Grid applications can be used on flow analysis | Interception method can not use grid applications.<br><br>Mirrored traffics can be analyzed in parallel |
| **Processed data size** | Around 15 Mega byte flow traces for 90% utilized 1 Gbps link for a 5-minute traffic | 33,75 Giga byte traffic for 90% utilized 1 Gbps link for a 5-minute traffic |
| **Detection of encrypted P2P traffic** | Yes | No |
| **Adaptability** | Training part can be run again | New signatures must be identified for every protocol change |
| **Ethical Issues** | Only the packet header information is used | Packet payloads are inspected which violates the user data privacy |

Our technique can be used to limit P2P contributors to certain bandwidths and reserve some bandwidth for the other protocols. As a result service quality of Turkish Academic Network will be increased.

We used a signature based P2P detection to distinguish the P2P and non-P2P traffic flows, so the performance of our method is directly related to the performance of the signature based method. Although we made a detailed analysis in selecting the signature based tool, concatenating two or more signature based tools as a future work can improve our method. This will also allow us to find the success rate of our method for individual P2P protocols. In addition, constructing the Bayesian Network with the absence of a specific P2P protocol and then applying the method to detect that protocol can also be done with the concatenated signature based methods. The signature based tool which is located to the nearest position to the source of the traffic has to be configured to drop the packets of a specific protocol (e.g. Kazaa) for this purpose. The next signature based method will be used to log the P2P contributors and the other steps will be same with our method. The only difference is that the drop rule in the first signature based method should be removed to test our method.

Our future work includes building a more mathematical approach of feature discovery phase. Since we identify the differentiating features by the significant variations in P2P and non-P2P traffic characteristics' distribution graphs, a more complex comparison on those characteristics can reveal some other features. And also, a correlation analysis on the differentiating features is needed, since the effects of P2P traffic characteristics on the distributions we analyze are similar. For example, flows with small packet numbers will also have little time.

A very important contribution of our approach is demonstrating that the solutions proposed for intrusion detection problem can be adopted to P2P traffic detection problem. Intrusion Detection Systems have been studied for a time period much longer than the existence of P2P traffic. Further improvements in P2P traffic detection can be achieved by using the experiences in intrusion detection research.

## REFERENCES


**[1]** A. Gerber, J. Houle, H. Nguyen, M. Roughan, and S. Sen, "P2P The gorilla in the cable," *in Proc. National Cable \& Telecommunications Association (NCTA) 2003 National Show*, 2003.

**[2]** S. Sen, and J. Wang, "Analyzing peer-to-peer traffic across large networks," *IEEE/ACM Trans. Networking*, vol. 12, pp. 219-232, April 2004.

**[3]** Sai Ho Kwok , S. M. Lui , Ricky Cheung , Sally Chan , Christopher C. Yang, "Searching Behavior in Peer-to-Peer Communities," *Proceedings of the International Conference on Information Technology: Computers and Communications*, p.130, April 28-30, 2003

**[4]** Mauro Andreolini, Riccardo Lancellotti, Philip S. Yu, "Analysis of Peer-to-Peer Systems: Workload Characterization and Effects on Traffic Cache ability," *12th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'04)*, mascots, pp. 95-104, 2004.

**[5]** C. Kruegel, D. Mutz, W. Robertson, and F. Valeur, "Bayesian event classification for intrusion detection," *in Proc. 19th IEEE Annual Computer Security Applications Conference*, pp. 14-23, 2003.

**[6]** ULAKBIM, http://www.ulakbim.gov.tr. , 06.06.06

**[7]** TÜBİTAK, http://www.tubitak.gov.tr, 06.06.06

**[8]** Napster, http://www.napster.com, 06.06.06

**[9]** Cachelogic, http://www.cachelogic.com, 06.06.06

**[10]** Gnutella, http://www.gnutella.com/, 06.06.06

**[11]** Grokster, http://www.grokster.com/, 06.06.06

**[12]** Kazaa, http://www.kazaa.com, 06.06.06

**[13]** Imesh, http://www.imesh.com, 06.06.06

**[14]** DirectConnect, http://www.dcplusplus.sourceforge.net, 06.06.06

**[15]** BitTorrent, http://www.bittorrent.com, 06.06.06

**[16]** T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy, "Transport layer identification of P2P traffic," *in Proc. 4th ACM SIGCOMM conference on Internet measurement*, pp.121-134, 2004.

**[17]** S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of P2P traffic using application signatures," *in Proc. 13th international conference on World Wide Web*, pp.512-521, 2004.

**[18]** T. Karagiannis, A. Broido, N. Brownlee, kc claffy, and M. Faloutsos, "File-sharing in the Internet: A characterization of P2P traffic in the backbone," *Technical report*, 2004. http://www.cs.ucr.edu/~tkarag.

**[19]** A. Spognardi, A. Lucarelli, and R. Di Pietro, "A methodology for P2P file-sharing traffic detection," *in Proc. International Workshop on Hot Topics in Peer-to-Peer Systems*, pp.52-61, 2005.

**[20]** Snort, http://www.snort.com, 06.06.06

**[21]** Cisco, http://www.cisco.com, 06.06.06

**[22]** NBAR, http://www.cisco.com/go/nbar, 06.06.06

**[23]** Azureus, http://azureus.aelitis.com/wiki/index.php/Avoid_traffic_shaping, 06.06.06

**[24]** T. Haniudu, K. Chujo, T. Chujo, and X. Yung, "Peer-to-Peer traffic in metro networks: Analysis, modeling, and policies," *in Proc. NOMS 2004 - IEEE/IFIP Network Operations and Management Symposium*, pp. 425-438, 2004.

**[25]** L. Nistor "Rules Definition for Anomaly Based Detection," *A Technical Report* 2002.

**[26]** G. Vigna, R. A. Kemmerer, "NetSTAT: A network-based intrusion detection system," *Journal of Computer Security*, 7(1): 37-71, 1999.

**[27]** M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *USENIX Lisa 99*, 1999.

**[28]** K. Ilgun. USTAT: A Real-time Intrusion Detection System for UNIX. In *Proceedings of the IEEE Symposium on Research on Security and Privacy*, Oakland, CA, May 1993.

**[29]** RealSecure, http://www.iss.net/products_services/enterprise_protection, 06.06.06

**[30]** Swatch: Simple Watchdog, http://swatch.sourceforge.net, 06.06.06

**[31]** Zheng Shan, Peng Chen, Ying Xu, Ke Xu, "A Network State Based Intrusion Detection Model," *iccnmc*, *International Conference on Computer Networks and Mobile Computing (ICCNMC'01)*, p. 481 2001.

**[32]** SpamAssassin, http://spamassassin.apache.org/, 06.06.06

**[33]** AMaVIS, http://www.amavis.org/, 06.06.06

**[34]** D. Denning. An Intrusion Detection Model. *IEEE Transactions on Software Engineering*, 13(2):222–232, Feb. 1987.

**[35]** H. S. Javitz and A. Valdes. The SRI IDES Statistical Anomaly Detector. In *Proceedings of the IEEE Symposium on Security and Privacy*, May 1991

**[36]** S. Forrest. A Sense of Self for UNIX Processes. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 120–128, Oakland, CA, May 1996

**[37]** C. Warrender, S. Forrest, and B. Pearlmutter. Detecting intrusions using system calls: Alternative data models. In *IEEE Symposium on Security and Privacy*, pages 133–145, 1999

**[38]** C. Kruegel, T. Toth, and E. Kirda. Service Specific Anomaly Detection for Network Intrusion Detection. In *Symposium on Applied Computing (SAC)*. ACM Scientific Press, March 2002.

**[39]** P. Porras and P. Neumann. EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances. In *Proceedings of the 1997 National Information Systems Security Conference*, October 1997.

**[40]** P. A. Porras and A. Valdes. Live traffic analysis of TCP/IP gateways. In *Proceedings of the 1998 ISOC Symposium on Network and Distributed System Security (NDSS'98)*, San Diego, CA, 1998

**[41]** A. Valdes and K. Skinner, "Adaptive, model-based monitoring for cyber attack detection," *In Proc. of RAID 2000*, Tolouse, France, October 2000

**[42]** A. A. Sebyala, T. Olukemi, and L. Sacks, "Active platform security through intrusion detection using naive Bayesian Network for anomaly detection," *In London Communications Symposium*, 2002.

**[43]** F. Jensen "Bayesian networks and decision graphs," *Springer*, New York, USA, 2001.

**[44]** Netflow, http://www.cisco.com/go/netflow, 06.06.06

**[45]** Debian, http://www.debian.org/, 06.06.06

**[46]** IpTables, http://www.netfilter.org/, 06.06.06

**[47]** Netfilter, http://www.netfilter.org/, 06.06.06

**[48]** ipP2P, http://rnvs.informatik.uni-leipzig.de/ipP2P, 06.06.06

**[49]** Gawk, http://www.gnu.org/software/gawk, 06.06.06

**[50]** Flow-tools, http://www.splintered.net/sw/flow-tools, 06.06.06.

**[51]** MSBNx, http://research.microsoft.com/adapt/MSBNx, 06.06.06

**[52]** Iana, http://www.iana.org/assignments/port-numbers, 06.06.06

**[53]** Ş. E. (Güran) Schmidt, M. Soysal, "An Intrusion Detection Based Approach for the Scalable Detection of P2P Traffic in the National Academic Network Backbone," *7th International Symposium On Computer Networks (ISCN*06*)*, Istanbul, July 2006 (Accepted).

**[54]** Jaakko Hollmén. User Profiling and Classification for fraud detection in mobile communications networks. PhD thesis 2000, Helsinki University of Technology

**[55]** Richard Quinn, http://richard-quinn.com/quinn-pages/essays/p2p/peer-to-peer.html, 06.06.06

## APPENDIX A

### Turkish Academic Network and Information Center (ULAKBIM)
### Acceptable Use Policy (AUP)

**Background and Definitions**

- 1. The hereby Acceptable Use Policy defines the rules applied to all the users of "UlakNet".
- 2. User organizations are universities and research institutions; users are the students, faculty, researchers, and other personnel working in these organizations.
- 3. "UlakNet" is the name given to the collection of networking services and facilities which support the communication requirements of the users via a network operating with TCP/IP protocol.
- 4. UlakNet is maintained by ULAKBIM, an institute of the Scientific and Technical Research Council of Turkey (TUBITAK), in accordance with the TUBITAK regulations.
- 5. UlakNet is a network comprising the principles and obligations stated in this document, of which the Signatory is informed and is fully responsible.
- 6. This Policy applies in the first instance to any organization authorized to use UlakNet. It is the responsibility of User Organizations to ensure that members of their own user communities use UlakNet services in an acceptable manner and in accordance with current legislation, and take the necessary precautions.
- 7. It is therefore recommended that each User Organization establishes its own statement of acceptable use to be signed by its users.

**Acceptable Use Policy**

- 8. The network's good use and security calls for a good cooperation among the different users. This cooperation is especially based on the Signatory's commitment on behalf of the sites' users he is responsible for, directly or indirectly connected with UlakNet, to adhere to the following rules:

    - 8.1. to use UlakNet network for strictly professional purposes: education, scientific research, technical development, transfer of technologies, diffusion of scientific, technical & cultural information;
    - 8.2. to transport and give access to the Network only for licit data, according to the appropriate legislation;
    - 8.3. not to give access, as commercial or not, under payment or not, to the UlakNet network to any non-authorized third party;
    - 8.4. to implement technical and human resources in order to ensure a permanent security level, and to prevent any eventual acts of intrusion from or through his site(s);
    - 8.5. their own mail servers should be closed to relay mail.

**Unacceptable Use**

- 9. Violations of system or network security are prohibited, and may result in criminal and civil liability. ULAKBIM will investigate incidents involving such violations and will cooperate with law enforcement if a criminal violation is suspected. UlakNet may not be used for any of the following:
    - 9.1. sending unsolicited mail messages including commercial advertising or informational announcements (SPAM mail);
    - 9.2. using another user's mail server to relay mail without the express permission of the site;
    - 9.3. generating highly disruptive traffic patterns, which affect the quality of service on the UlakNet;
    - 9.4. the creation and transmission (other than for properly supervised and lawful research purposes) of any offensive, obscene or indecent images, data or other material;
    - 9.5. the creation and transmission of material which is likely to cause annoyance, inconvenience or needless anxiety;
    - 9.6. the creation and transmission of defamatory material;
    - 9.7. the transmission of material (including, without limitation; the works in the form of text, article, book, film, music) such that this infringes the copyright of another person;
    - 9.8. the deliberate unauthorized access to national or international services accessible via UlakNet;
    - 9.9. deliberate activities with any of the following characteristics:
        - 9.9.1. corrupting or destroying other users' data;
        - 9.9.2. violating the privacy of other users;
        - 9.9.3. disrupting the work of other users;
        - 9.9.4. creating traffic over UlakNet in a way that denies service to other users;
        - 9.9.5. continuing to use a software after ULAKBIM has announced a notice requesting the cease of its use because it is causing unnecessary traffic and disrupting the correct functioning of UlakNet.

**Compliance**

- 10. The Signatory of this Acceptable Use Policy is informed and expressly accepts that the ULAKBIM has the ability to control the correct use of the network, in accordance with the rules stated in items 8 and 9, provided that the user's personal rights are kept intact.
- 11. The Signatory accepts that ULAKBIM could take emergency measures, including the decision to limit or interrupt temporarily the access to Ulaknet of his site(s) at a national or international level, in order to preserve security in case of any troubling incident ULAKBIM would be aware of. However, these measures will be taken respecting the best terms of time and after

communication with the concerned site(s), unless this situation affects the general functioning and security of the network.

- 12. In the case that the Users are harmed by the repeated hostile behavior of another User; ULAKBIM may take measures upon the request of the Signatory or any other related User, with conditions described above.
- 13. The Signatory is informed and expressly accepts that ULAKBIM, mainly in order to take into account legal evolution which might occur in this sector, can modify this Acceptable Use Policy; ULAKBIM reserves the right to modify the policy and the agreement at anytime. The modified agreement becomes effective upon posting of it to the URL address below: http://www.ulakbim.gov.tr/ulaknet/basvuru/kullanimpolitika.uhtml

The Signatory agrees to be fully acquainted with the Acceptable Use Policy of ULAKBIM and abides by these commitments.

The Signatory
name, surname, title

The Signatory
name, surname, title

Signature:

**Table 6 List of Popular P2P Programs**

| Application | Protocol | Type | Major Use | Licensing Model | Revenue Model | Tech Details |
|---|---|---|---|---|---|---|
| **BCDC++** | Direct Connect | Hybrid, hubs and supernodes | File Sharing | ? | ? | C++ |
| **BearShare** | Gnutella | Hybrid, hubs | File Sharing | Closed Source | Advertising | Windows Only |
| **BitTorrent** | BitTorrent | Decentralized, mixed roles | File Sharing, RSS Feed sharing | Open Source, MIT License | Voluntary - donations, some advertising | Python with xWindos client. |
| **DC++** | Direct Connect | Hybrid, hubs and supernodes | File Sharing | Open Source, GNU GPL | Voluntary | C++, Client only |
| **eDonkey** | eDonkey | Decentralized | File Sharing | Closed Source | Voluntary, not for profit ?? | |
| **eMule** | eDonkey | Decentralized | File Sharing | Closed Source | Voluntary, not for profit | Windows only |
| **Freenet** | Freenet | Hybrid, nodes | Information dispersal, Anonymous | Open Source, GNU GPL | Voluntary | Java |
| **GnouGat** | JXTA | Decentralized | File Sharing | Open Source, Sun Project JXTA Software License | Voluntary | Java only (JINI) |
| **GnuNet-GTK** | GnuNet | Decentralized, anonymous | File Sharing | Open Source, GNU GPL | Voluntary | Source only, UNIX GTK, HTTP, SMTP, |
| **Grokster** | FastTrack | Hybrid | File Sharing | Closed Source | Advertising, Content distribution fee | Windows Only |
| **Groove** | Groove | Decentralized | Groupware | Closed Source | Fee Based | Fee Based |

**Table 6 Continued**

| Application | Protocol | Type | Major Use | Licensing Model | Revenue Model | Tech Details |
|---|---|---|---|---|---|---|
| **iMesh** | FastTrack | Decentralized | File Sharing | Closed Source , freeware | Advertising | Windows Client |
| **KaZaa** | FastTrack | Hybrid | File Sharing | Closed source | Advertising, Spyware, Malware | |
| **Morpheus** | FastTrack, Gnutela2, Direct Connect | Hybrid | File Sharing | Closed Source | Advertising | Windows only |
| **Napster** | Napster | Napster is now defunct as a P2P network | | | | |
| **NeoModus** | Direct Connect | Hybrid, hubs and supernodes | File Sharing | Closed Source | Advertising | C++, C# |
| **OverNet** | Overnet, eDonkey | Decentralized | File Sharing (large files) | Closed Source | Voluntary, some advertising ?? | Windows client only |
| **Piolet / Blubster** | Manolito, MP2P | Decentralized | File Sharing, MP3 only | Closed Source | ?? | Windows Only |
| **Rebol** | Rebol IOS | Decentralized | Internet Operating System | Closed Source | Commercial | Over 40 OSs |
| **ShareAza** | BitTorrent, eDonkey, Gnutella, Gnutella2 | Hybrid | File Sharing | Closed Source, freeware | Advertising | Windows Client |
| **Soulseek** | Soulseek | Centralized | MP3 File Sharing | Closed Source | Contribution based | |
| **WinMX** | OpenNap, WPNP | Hybrid | File Sharing | ? | ? | Windows Client |

## APPENDIX C

## A SAMPLE OUTPUT OF *flow-stat –f0*

```
#  --- ---- ---- Report Information --- --- ---
#
# Fields:    Total
# Symbols:   Disabled
# Sorting:   None
# Name:      Overall Summary
#
# Args:      flow-stat -f0
#
Total Flows                 : 23546
Total Octets                : 363744391
Total Packets               : 320386
Total Time (1/1000 secs) (flows): 131592068
Duration of data  (realtime)    : 86379
Duration of data (1/1000 secs)  : 4294952808
Average flow time (1/1000 secs) : 5588.0000
Average packet size (octets)    : 1135.0000
Average flow size (octets)      : 15448.0000
Average packets per flow        : 13.0000
Average flows / second (flow)   : 0.0055
Average flows / second (real)   : 0.2726
Average Kbits / second (flow)   : 0.6775
Average Kbits / second (real)   : 33.6882
```

**IP packet size distribution:**

```
  1-32   64   96  128  160  192  224  256  288  320  352  384  416  448  480
  .000 .411 .024 .042 .074 .016 .012 .015 .010 .011 .008 .006 .007 .006 .010
```

512 544 576 1024 1536 2048 2560 3072 3584 4096 4608
.006 .008 .014 .077 .244 .000 .000 .000 .000 .000 .000

**Packets per flow distribution:**

1 2 4 8 12 16 20 24 28 32 36 40 44 48 52
.667 .130 .100 .054 .017 .007 .005 .003 .002 .002 .001 .001 .001 .001 .001

60 100 200 300 400 500 600 700 800 900 >900
.001 .003 .002 .001 .001 .000 .000 .000 .000 .000 .003

**Octets per flow distribution:**

32 64 128 256 512 1280 2048 2816 3584 4352 5120 5888 6656 7424 8192
.000 .368 .082 .108 .058 .058 .124 .020 .052 .010 .030 .007 .017 .005 .008

8960 9728 10496 11264 12032 12800 13568 14336 15104 15872 >15872
.003 .005 .003 .003 .003 .002 .003 .001 .003 .001 .026

**Flow time distribution:**

10 50 100 200 500 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000
.696 .039 .016 .020 .031 .028 .026 .017 .012 .009 .009 .010 .006 .006 .006

12000 14000 16000 18000 20000 22000 24000 26000 28000 30000 >30000
.009 .008 .009 .007 .004 .004 .003 .002 .001 .002 .020