CERIAS Tech Report 2011-14

MEASUREMENT-DRIVEN CHARACTERIZATION OF EMERGING TRENDS IN INTERNET CONTENT DELIVERY

by Ruben Torres
Center for Education and Research
Information Assurance and Security
Purdue University, West Lafayette, IN 47907-2086

MEASUREMENT-DRIVEN CHARACTERIZATION OF EMERGING TRENDS IN INTERNET CONTENT DELIVERY

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Ruben Dario Torres Guerra

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2011

Purdue University

West Lafayette, Indiana

To my parents, Ruben and Carmen, and to my brother, Daniel, whom have supported me from the beginning of this long journey.

ACKNOWLEDGMENTS

Special thanks to Dr. Sanjay Rao, my thesis advisor, for all his guidance and encouragement throughout the six years that I have been his student. Working for such a dedicated and kind professor has made my graduate school experience a very pleasant one. Thanks to Marco Mellia and Maurizio Munafo from Politecnico di Torino, for all their help in collecting and analyzing the datasets that were a key component of my dissertation. Thanks to the Purdue IT Security team for facilitating the collection of datasets from Purdue. To my colleagues at the Internet Systems Lab, specially Eric, Mohammad and Shankar, for bringing life to the office when it was most needed. To the members of my thesis committee, Cristina Nita-Rotaru, Charlie Hu and Vijay Pai, for their valuable comments to improve my dissertation work. Finally, I thank all my friends, specially to Salvador Acuna and Gaspar Modelo-Howard, for all the fun and moral support provided over the years.

TABLE OF CONTENTS

Page

LIST OF TABLES x

LIST OF FIGURES xi

ABSTRACT xv

1 INTRODUCTION 1

1.1 Detecting and Preventing Undesirable Behavior of P2P Clients . . . 2

1.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems 3

1.3 Dissecting Video Server Selection Strategies in the YouTube CDN . 4

1.4 Research Methodology 4

1.5 Contributions 5

1.5.1 Detecting and Preventing Undesirable Behavior of P2P clients 5

1.5.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems 6

1.5.3 Dissecting Video Server Selection Strategies in the YouTube CDN 7

1.6 Thesis Roadmap 8

2 INFERRING UNDESIRABLE BEHAVIOR FROM P2P TRAFFIC ANALYSIS 9

2.1 Methodology Overview 11

2.2 Datasets 13

2.2.1 Setup and Trace Collection 14

2.2.2 Description of P2P Systems 16

2.2.3 Preliminary Traffic Analysis 17

2.3 Traffic Classification 19

2.3.1 Selecting Flows of Interest 19

Page

2.3.2 Aggregating Flows into Host Samples 19
2.4 Metrics 20
2.4.1 Metrics Independent of Flow Initiator 21
2.4.2 Metrics Dependent on Flow Initiator 22
2.5 Identifying Unwanted Behavior 23
2.5.1 Density-based Clustering 24
2.5.2 Interesting Region Selection 27
2.5.3 Correlation Across Interesting Samples 28
2.6 Results 29
2.6.1 High Level Characteristics of Systems 29
2.6.2 Interesting Region Selection 30
2.6.3 Host Distribution in Interesting Region 35
2.7 Key Findings 35
2.7.1 DDoS Attacks Exploiting P2P Systems 36
2.7.2 Unnecessary P2P Traffic 37
2.7.3 Malicious P2P Servers 39
2.7.4 Other Interesting Findings 41
2.8 Generalizing to other Systems 43
2.9 Related Work 44
2.10 Conclusions and Discussion 45
3 PREVENTING DDOS ATTACKS ON INTERNET SERVERS EXPLOITING P2P SYSTEMS 47
3.1 Introduction 47
3.2 DDoS Attacks by Exploiting P2P Systems 49
3.2.1 Systems Background 50
3.2.2 Attacks 51
3.2.3 Potential Evidence for Real Attacks in the Wild 54
3.3 Eliminating Attack Amplification 55

Page

3.3.1 Classifying Attacks by Amplification Causes 55
3.3.2 Principles for Robust Design 57
3.4 Enhancing DDoS Resilience 58
3.4.1 Approaches for Validating Membership Information 59
3.4.2 Validating Peers through Active Probing 60
3.4.3 Preventing DDoS Exploiting Validation Probes 61
3.4.4 Avoiding Validation Failures with NATs 63
3.4.5 Illustrating the Scheme 64
3.5 Analysis 65
3.5.1 DDoS Attacks 65
3.5.2 Attacks on Destination-throttling 70
3.6 Evaluation Methodology 72
3.6.1 Evaluation Goals 72
3.6.2 Performance Metrics 73
3.6.3 Methodology 73
3.7 Parameterizing the Validation Framework 75
3.7.1 File Sharing Application 75
3.7.2 Video Broadcasting Application 82
3.7.3 Discussion 83
3.8 Evaluation under Attack 84
3.8.1 File Sharing Application 85
3.8.2 Video Broadcasting Application 88
3.9 Interactions with P2P Developers 89
3.10 Related Work 90
3.11 Conclusions 92
4 UNCOVERING CLOSED COMMUNITY BASED P2P SYSTEMS . . . 94
4.1 Introduction 94
4.2 Peer-to-Peer Communities 97

Page

4.2.1 P2P Community in an ISP Network 97
4.2.2 P2P Community in a Campus Network 98
4.3 Evaluation Goals and Methodology 99
4.3.1 Goals 99
4.3.2 Trace Collection Tool 100
4.3.3 Datasets 100
4.3.4 Comparing P2P Communities with Generic P2P Systems . . 101
4.4 Characterizing Community Usage 102
4.4.1 Prevalence of Communities 102
4.4.2 User Reliance on Communities 104
4.5 User and System Performance 105
4.5.1 Throughput 106
4.5.2 Delay 109
4.6 Traffic matrix for P2P communities 110
4.6.1 Validation 112
4.7 Impact of Communities on the Network 114
4.7.1 Modeling Approach 115
4.7.2 Predicting Traffic Matrix Changes 116
4.7.3 Results 117
4.8 Discussion and Implications 118
4.9 Related Work 121
4.10 Conclusions 122
5 DISSECTING VIDEO SERVER SELECTION STRATEGIES IN THE YOUTUBE CDN 124
5.1 Introduction 124
5.2 YouTube Basics 126
5.3 Methodology 127
5.3.1 Collection Tool 127

Page

5.3.2 Datasets 128
5.4 AS Location of YouTube Servers 130
5.5 Server Geolocation 131
5.6 Evaluating YouTube's Server Selection Algorithm 134
5.6.1 Video Flows and Sessions 134
5.6.2 Understanding Server Selection Strategy 137
5.6.3 Mechanisms Resulting in Accesses to Non-preferred Data Centers 139
5.7 Causes Underlying Non-preferred Data Center Accesses 141
5.7.1 DNS-level Load Balancing 141
5.7.2 Variations Across DNS Servers in a Network 143
5.7.3 Investigating Redirection at the Application Layer 144
5.8 Impact of Redirections on User Performance 148
5.8.1 Analyzing the Startup Delay 149
5.8.2 Analyzing the Ratio of Download Rate to Playback Rate . . 150
5.9 Discussion 152
5.10 Related Work 154
5.11 Conclusion 155
6 CONCLUSIONS AND FUTURE WORK 157
6.1 Summary of Contributions 157
6.1.1 Detecting and Preventing Undesirable Behavior of P2P Clients 157
6.1.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems 158
6.1.3 Dissecting Video Server Selection Strategies in the YouTube CDN 159
6.2 Future Work 159
6.2.1 Video Content Distribution in Mobile Devices 160
6.2.2 Modeling "What-if" Scenarios of Changes in Video Distribution Systems 160

Page
LIST OF REFERENCES 162
VITA 169

LIST OF TABLES

Table		Page
2.1	General Statistics of P2P Traffic	18
2.2	List of Metrics	23
2.3	DBScan Sensitivity	27
2.4	Metrics with Multiple Clusters - Kad	31
2.5	Metrics with Multiple Clusters - KadU	32
3.1	Classification of DDoS Attacks by their Major Source of Amplification	56
3.2	Destination-throttling: Percentage of contacted prefixes blocked with the particular search rate and for various combinations of (D_{prefix}, F_{prefix}). Measured in Kad.	79
4.1	Traffic Summary for the *ISP-FTTH* Trace	103
4.2	Traffic Summary for the *ISP-ADSL* Trace	103
4.3	Traffic Summary for the Campus Trace	103
4.4	Access Technology of *ISP-Community* Peers	107
5.1	Traffic Summary for the Datasets	129
5.2	Percentage of Servers and Bytes Received per AS	131
5.3	Google Servers per Continent on each Dataset	134
5.4	Fraction of Sessions with Redirections	151

LIST OF FIGURES

Figure Page

2.1 Schematic overview of the proposed methodology. 12

2.2 Schematic representation of possible connections from hosts in the MiniPop versus other hosts. 13

2.3 Traffic volume shares in the MiniPoP. The top plot reports the inbound traffic, while the bottom plot reports the outbound traffic. Only HTTP and eMule traffic is reported. On the bottom plot, HTTP traffic is small and not visible. 15

2.4 *outin-fract-unanswered-UDP* is an example of multiple cluster metric. . 23

2.5 *inout-avg-conn-per-IP-TCP* is an example of single cluster metric. . . . 24

2.6 Kad: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results. 36

2.7 KadU: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results. 36

2.8 Fraction of unanswered flows per destination port. 38

2.9 Host leaves the group about 11 hours after the beginning of the trace. Fraction of unanswered flows on the top and total number of unanswered flows on the bottom. 40

2.10 Connections made by the KadU client $h1$ towards $Server1$ (fake server) and $Server2$ (full server). 41

3.1 a) Normal Operation. b) Attack. 51

3.2 Index poisoning attack in Overnet 51

3.3 a) Number of unsuccessful Kad flows sent to ports that received more than 500 unsuccessful flows. b) Fraction of unsuccessful Kad flows to ports that received more than 500 unsuccessful flows. 56

3.4 Complete sequence of steps for normal buddy operations with probing-based validation mechanisms. 65

Figure Page

3.5 Source-throttling: Impact of m on search delay. Measured in Kad. . . . 75

3.6 Source-throttling: Sensitivity to Planetlab Site. For each site, each bar is the average over 5 runs of the 90th percentile of the delay of successful searches. The error bars show the standard deviation. Measured in Kad. 76

3.7 Source-throttling: Sensitivity to the number of membership entries returned in a reply message. For each site, each bar is the average over 5 runs of the 90th percentile of the delay of successful searches. The error bars show the standard deviation. Measured in Kad. 76

3.8 Destination-throttling: Sensitivity to Planetlab Site. For each site, each bar is the average over 5 runs of the percentage of contacted prefixes that are blocked. The error bars show the standard deviation. Note that for all but one site, when (D_{prefix},F_{prefix}) = (10,10), 0% of prefixes are blocked. Measured in Kad. 79

3.9 Source-throttling: Impact of m on the join time of nodes. Each bar is the average over 5 runs of the 90th percentile of the join time. The error bars show the standard deviation. Measured in ESM. 81

3.10 Traffic seen at the victim as a function of time, with 5 attackers, and 50% of the nodes behind NAT. Measured in Kad. 85

3.11 Average time a search takes to locate the index node, with different NAT percentages. Each bar is the average over 5 runs. The error bars show the standard deviation. Measured in Kad. 86

3.12 Fraction of nodes that see more than 90% of the source rate, with 10% of the nodes as attackers and different NAT percentages. Each bar is the average over 5 runs. The error bars show the standard deviation. Measured in ESM. 88

4.1 ISP setup and trace collection. 96

4.2 Fraction of inbound and outbound traffic seen at the monitoring point, generated by the P2P communities. Note that all this traffic stays within the Campus/ISP. 104

4.3 Fraction of clients in *Campus-Community* and *ISP-Community* with a community usage ratio over 80%(90%). 105

4.4 Download rates for various systems. 107

4.5 Download rates from the *ISP-ADSL* trace, distinguishing the access technology of the source. 108

4.6 Ratio of bytes sent and bytes received per client in *ISP-Community*. . . 109

Figure Page

4.7 RTT for connections initiated for various system. 110

4.8 Comparison of output from the gravity model and the real data for the *ISP-FTTH* trace. 113

4.9 Comparison of output from the gravity model and the real data for the *ISP-ADSL* trace. 113

4.10 Relative error of the gravity model for the *ISP-ADSL* and the *ISP-FTTH* traces. 114

4.11 CDF of volume of traffic related to P2P communities, seen by all links in the ISP topology, for the *ISP-Community* and *GenericOnly*, varying the access technology of peers. 118

4.12 Difference in traffic per link considering the *ISP-Community-CurrentTech* and *GenericOnly-CurrentTech* scenarios. There is one plot for each link type. 119

5.1 High level sequence of steps to retrieve content. 127

5.2 RTT to YouTube content servers from each of our vantage points. . . . 132

5.3 Radius of the CBG confidence region for the YouTube servers found in the datasets. 133

5.4 CDF of YouTube flow sizes. 134

5.5 Number of flows per session with different values of T for the US-Campus dataset. 135

5.6 Number of flows per session for all datasets using T=1 second 135

5.7 Fraction of the total YouTube video traffic served by a data center with an RTT less than a given value from the dataset collection point. . . . 137

5.8 Fraction of the total YouTube video traffic served by a data center with a distance less than a given value from the dataset collection point. . . . 137

5.9 Variation of the fraction of video flows directed to a non-preferred data center over time. One hour long time periods are considered. 138

5.10 Breakdown of sessions based on whether flows of the session are sent to preferred data center. 140

5.11 Fraction of the total YouTube video traffic served by the preferred data center (top graph) and total number of video flows (bottom graph) as a function of time for the EU2 dataset. 142

Figure Page

5.12 Fraction of all video flows, and video flows to non-preferred data centers for each internal subnet of the US-Campus dataset. 143

5.13 Number of requests for a video to the non-preferred data centers. . . . 144

5.14 Load related to the top 4 videos with the highest number of accesses to the non-preferred data centers for the EU1-ADSL dataset. 144

5.15 Average and maximum number of requests per server in the preferred data center of EU1-ADSL dataset. 145

5.16 Number of video sessions per hour seen by the server handling *video1* in the preferred data center of the EU1-ADSL dataset. A breakdown of sessions based on whether flows are directed to preferred data centers is also shown. 145

5.17 Variation over time of the RTT between a PlanetLab node and the content servers for requests of the same test video. 147

5.18 Reduction in RTT from PlanetLab nodes to the content servers when a test video is downloaded twice. The first access may incur a higher RTT due to unavailability of content in the preferred data center. 148

5.19 Start delays with redirection. 150

5.20 Fraction of flows with slow download rate 152

5.21 Fraction of flows with slow download rate, for preferred and non-preferred data centers 153

5.22 Average fraction of video download for slow flows and not-slow flows. . 153

ABSTRACT

Torres Guerra, Ruben Dario Ph.D., Purdue University, December 2011. Measurement-Driven Characterization of Emerging Trends in Internet Content Delivery . Major Professor: Sanjay G. Rao.

In the last decade, there have been radical changes in both the nature of the mechanisms used for Internet content distribution, and the type of content delivered. On the one hand, Peer-to-Peer (P2P) based content distribution has matured. On the other hand, there has been a tremendous growth in video traffic. The goal of this thesis is to characterize these emerging trends in content distribution and understand their implications for Internet Service Providers (ISP) and users. Such characterization is critical given the predominance of P2P and video traffic in the Internet today and can enable further evolution of content delivery systems in ways that benefit both providers and users.

In this thesis, we make the following contributions: (i) We develop novel methodologies to identify undesirable behavior of P2P systems, and expose the prevalence of such behavior; (ii) We characterize private P2P communities, and discuss the implications of our findings on recent research on localization of P2P traffic within an ISP; (iii) We shed light into the factors that govern the data-center selection for video delivery in geographically distributed settings by characterizing YouTube, the most popular video distribution network in the Internet.

A common thread underlying these contributions, and a distinguishing highlight of this thesis is the analysis of terabytes of traffic traces collected from the edge of multiple ISP and Campus networks located in different countries.

1. INTRODUCTION

Internet content distribution has experienced radical changes in the last decade. These changes range from the mechanisms used to deliver content to the type of content being delivered. In particular we see two broad trends:

• In the last few years, P2P systems have developed to the point that they are widely used for many different applications such file sharing, video streaming, gaming, voice-over-IP and others. Some of these applications have become popular enough that they have been commercialized and are being used by millions of clients. For example, Skype [1], a voice-over-IP system, has more than 100 million clients [2]. Similarly, BitTorrent [3] is a file sharing application with several million users.

• Video is becoming the dominant type of content distributed on the Internet today. This is probably because of the popularity of video on demand services such as YouTube [4], Hulu [5] and Netflix [6], the improvement of broadband technologies (e.g. Fiber-to-the-Home) and the development of new and easy ways to access this content through mobile devices and digital media receivers such as Apple TV [7].

A characterization of these emerging trends in content distribution is necessary to greatly help understanding their implications for Internet service providers (ISP) and users. On the one hand, while there have been some work characterizing video content distribution [8–12], we are just scratching the surface in this area. Moreover, video distribution systems are constantly changing, so new research is required. On the other hand, while P2P systems have received much attention, new trends as they mature have developed: (i) with the growth in complexity of these systems, it becomes critical to monitor P2P nodes and ensure they behave as expected; (ii) P2P traffic has increased costs of transit traffic on ISPs, but recent research proposals to solve this problem [13–21] still require more scrutiny.

In this thesis, we make an effort to characterize some of these new trends in both P2P technologies and video content distribution. First, we develop methodologies to detect and expose undesirable behavior of P2P clients on a real ISP network. Second, we study the implications of localization of P2P traffic to an ISP by characterizing private communities of P2P users, where all users are confined to within a single ISP. Finally, we shed light into the factors that govern the data center selection for video delivery in YouTube, the most popular video distribution network in the Internet. A better understanding of these types of systems could enable researchers to conduct what-if analysis, and explore how changes in video popularity distributions, or changes to the infrastructure can impact ISP traffic patterns, as well as user performance. A distinguishing aspect of this thesis is the use of a measurement-driven approach to analysis. We analyze terabytes of traffic traces from five different large-scale networks spread across three countries including nation-wide ISPs and University campuses. The traces are collected with a tool that identifies the application that generates TCP and UDP flows using a combination of Deep Packet Inspection (DPI) and statistical classifiers.

In the rest of the introduction, we give a summary of the various pieces of this thesis and present the expected contributions of this work. We also provide a roadmap of the thesis at the end of this section.

1.1 Detecting and Preventing Undesirable Behavior of P2P Clients

P2P systems are large-scale, with millions of participants and complex design interactions. Further, they are difficult to debug because of their distributed nature and they may have security vulnerabilities [22–26]. Hence P2P clients could exhibit patterns of undesirable behavior. However, this behavior is not easy to detect since there is neither a clear definition of normality nor a list of anomalous behavior to look for. In this thesis, we systematically study a trace from a real P2P deployment in a point-of-presence (PoP) within a large-scale nationwide ISP and expose the prevalence

of undesirable behavior in the system. In addition, we develop a methodology to detect these types of behavior in the network, which rely on a combination of data mining techniques and manual inspection using domain knowledge of the P2P system and the ISP network.

One type of undesirable behavior we found in the analyzed traces that could be highly detrimental to the Internet is P2P clients participating in a Distributed Denial of Service (DDoS) attack. Since these systems consist of millions of participants spread around the world, a DDoS attack involving them would be difficult to shut down. In this thesis, we survey the various techniques that have been proposed to cause DDoS attacks exploiting P2P systems. In addition, we present an in-depth analysis and evaluation of mechanisms that can be implemented to make P2P systems robust to these types of vulnerabilities.

1.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems

Recently, there have been research efforts on mechanisms to localize P2P traffic to within ISP boundaries [13, 14]. While a majority of this research has taken the benefits of localization on users and ISPs for granted, we believe that a more critical examination is essential. We study implications of P2P traffic localization in the context of private communities of P2P systems. In the communities we analyze, membership is restricted by requiring that users must be connected to the same network (e.g., same ISP). In our study, we combine analysis of traffic traces with the use of models to evaluate the impact of localization on the ISP internal network and on user performance.

In a related joint work [27], we perform a simulation study on the impact of P2P traffic localization on ISP profitability if localization policies are adopted by ISPs throughout the Internet. Our simulations are based on detailed models of inter-AS P2P traffic and inter-AS routing, localization models that can predict the extent to

which P2P traffic is reduced, and pricing models to predict the impact of changes in traffic on an ISPs profit. To evaluate our models we use a large-scale crawl of BitTorrent involving over millions users sharing millions of files.

1.3 Dissecting Video Server Selection Strategies in the YouTube CDN

Today, video accounts for close to 25% of the traffic in the Internet [28–30]. YouTube is one of the main reasons for this video traffic dominance. In this thesis, we characterize the video delivery mechanisms employed by YouTube to serve video around the world. To this end, we analyze week-long traffic traces, containing more than 2.5 million YouTube TCP flows, collected from two nation-wide ISPs and two large Campus networks in two different continents. A more general goal of this work is on gaining better understanding of the factors that may affect server selection policies, which can help researchers to better analyze video distribution systems. A unique aspect of our work is that we perform our analysis on groups of related YouTube flows. This enables us to infer key aspects of the system design that would be difficult to glean by considering individual flows in isolation.

Our results indicate that clients that belong to a network are served from a preferred YouTube data center. This is implemented mostly through DNS-level resolution of content server names to IP addresses located in the preferred data center. More interesting however, is that there are cases in which clients are redirected to data centers different than the preferred. This can be due to a variety of reasons such as load balancing, video flash crowd and availability of rare content.

1.4 Research Methodology

Our research methodology comprises three major blocks: (i) Measuring P2P and content delivery systems on large ISPs and campus networks; (ii) Mining insights from collected data and (iii) Designing P2P systems and traffic models.

• *Measuring P2P and video delivery systems on large ISPs and campus networks*: We believe the best way to understand the impact of P2P and video delivery systems on large IP networks is by measuring them. We have collected and analyzed traces from five networks, including two large European ISPs with millions of customers and two large Campus networks, one in the United States and one in Europe, with thousands of users.

• *Mining insights from collected data*: We have analyzed terabytes of data looking for interesting insights that can help explain the behavior of P2P systems today. For instance, we studied more than 50 different metrics in millions of TCP and UDP flows, in order to find anomalies of P2P clients. Similarly, in our study of P2P private communities as well as the YouTube CDN study, we characterized these systems by analyzing millions of TCP and UDP flows in five different and large networks.

• *Designing systems and models based on insights*: Finally, we leverage the insights from the data to develop methodologies, systems and models. For instance, we have proposed methodologies to detect undesirable behavior of P2P clients. In addition, we have implemented changes to real P2P applications for video broadcasting and file sharing to make them robust to DDoS attacks.

1.5 Contributions

In this section, we present the contribution of this thesis. We group them according to the major pieces of the thesis:

1.5.1 Detecting and Preventing Undesirable Behavior of P2P clients

• *Characterization of undesirable behavior of P2P clients:* Our work is one of the first to show that undesirable behavior exists and is prevalent in real networks. Our analysis shows several examples of undesirable behavior including evidence of DDoS attacks exploiting live P2P clients, significant amounts of unwanted traffic that may harm network performance, and instances where the performance of participating

peers may be subverted due to maliciously deployed servers. We systematically study traces collected from a PoP within a nationwide ISP with more than 5 million users. This is a very unique network, where 70% of the inbound traffic and 95% of the outbound traffic are due to P2P data. We found it is hard to distinguish undesirable behavior from normal behavior in an automatic fashion due to the heterogeneity of the P2P traffic and the difficulty in characterizing normal behavior of clients. Hence, we develop mechanisms to detect undesirable behavior, that combine data mining techniques and manual inspection through domain knowledge.

• *Design and evaluation of mechanisms to prevent P2P systems from being exploited to create DDoS attacks:* We study in detail a type of undesirable behavior which we found with our detection methodology, P2P clients participating on DDoS attacks. Further, we propose and extensively evaluate active-probing based membership validation mechanisms to make P2P systems robust to these types of attacks.

1.5.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems

We present the first measurement study of communities of P2P clients that are localized to a network. We study two such systems used for content sharing, one hosted in a large nationwide ISP and the other in a campus network. In addition, we show the performance benefits experienced by clients of these systems and present a study of the effect of these systems in the traffic volumes carried by links in the host network. We draw lessons from our characterization study that apply to recent research on localization of P2P traffic to within ISP boundaries. In particular, our results indicate that (i) in ISPs with heterogeneous access technologies, the performance benefits to users on localizing P2P traffic is largely dependent on the degree of altruism of peers behind high-bandwidth access technologies; and (ii) while localization can reduce the traffic on Internet peering links, it has the potential to cause

a significant increase in traffic on internal links of providers, potentially requiring upgrades of network links.

In a joint parallel effort [27], we study the Internet-wide impact of P2P traffic localization on ISP profitability. Our contributions include a methodology to perform what-if analysis on the adoption of P2P localization by all ISPs in the Internet or by a limited number of ISPs. Some of our key findings include: (i) residential ISPs can actually lose money when localization is employed; (ii) the reduction in costs due to localization will be limited for ISPs with small P2P populations; and (iii) some ISPs can better increase profitability through alternate strategies to localization by taking advantage of the business relationships they have with other ISPs.

1.5.3 Dissecting Video Server Selection Strategies in the YouTube CDN

Our analysis indicates that the YouTube infrastructure has been completely redesigned compared to the one previously analyzed in the literature. In the new design, most YouTube requests are directed to a preferred data center and the RTT between users and data centers plays a role in the video server selection process. More surprisingly, however, our analysis also indicates a significant number of instances where users are served from a data center that is not the preferred one. In one of our datasets, up to 55% of video requests were not served from the preferred data center, while in most datasets at least 10% of requests were not served from the preferred data center.

We identified a variety of causes underlying accesses to non-preferred data centers. In some cases DNS mechanisms resulted in users being directed to non-preferred data centers for reasons including load balancing to handle variations in system load due to the day/night patterns in YouTube usage. Interestingly, we found other cases where video was streamed to the user from a non-preferred data center, even though DNS directs the user to the preferred data center. The common causes underlying such

cases included (i) alleviation of load hot spots due to popular video content; and (ii) accesses to unpopular video content that may not be available in a given data center.

1.6 Thesis Roadmap

Chapter 2 describes our findings of undesirable behavior of P2P clients in a large ISP. Chapter 3 presents a case study of an undesirable behavior, a DDoS attack exploiting P2P clients, and mechanisms to prevent such attack. Chapter 4 presents our analysis of a localized community of P2P clients. Chapter 5 presents our characterization study of the YouTube CDN. Finally, Chapter 6 lists a summary of our contributions and the future work.

2. INFERRING UNDESIRABLE BEHAVIOR FROM P2P TRAFFIC ANALYSIS

Peer-to-peer (P2P) systems have rapidly emerged in popularity in the last few years, and they have matured to the point we have recently seen several commercial offerings, including file sharing, VoIP and multimedia applications. Recent studies [31] indicate that over 60% of network traffic is dominated by peer-to-peer systems, and their emergence has drastically affected traffic usage and capacity engineering.

With the growth of P2P systems, many of which involve millions of hosts, and complex interactions between participating peers, it becomes critical to monitor these systems, and to ensure they are behaving as intended. Indeed, several reports are emerging about potential vulnerabilities in these systems either due to implementation bugs, or design flaws [22–26]. The behavior may be undesirable either from the perspective of the performance of the system, or in terms of unwanted traffic (malicious or otherwise) generated by the systems.

Detecting undesirable behavior is of interest to network operators, P2P system developers, and actual P2P end-users. Network operators may wish to identify causes for large traffic consumption, or they may want to optimize P2P traffic delivery, e.g., limit traffic peering costs. Knowledge of undesirable behavior and its causes can aid P2P system developers in augmenting the design of the systems. Finally, end-users seek to ensure that their host is not being exploited for malicious purposes, and care about application performance.

While the ultimate objective is automated identification of undesirable behavior, there is limited understanding in the community today on the patterns of undesirable behavior that P2P systems may exhibit, and the prevalence and seriousness of such behavior in real networks. Our primary contribution in this paper is to create such understanding by systematically analyzing real traffic traces collected from a

Point-of-Presence (PoP) of a nation-wide ISP. In the ISP we consider, 70% (95%) of inbound (outbound) traffic is due to eMule [32], a popular file-sharing system, and the associated Kad network, one of the largest DHT-based deployments. We analyze a 25 hour trace, comprising about 2TB of data. Another interesting aspect of this dataset is the use of a modified Kad system - called KadU - within the ISP network that was optimized by a large community of ISP users to exploit the peculiarities of the ISP architecture.

One of the key challenges we faced in our study is that it is hard to distinguish undesirable behavior from normal usage in a completely automated fashion, given the intrinsic heterogeneity of P2P traffic, and given there are few assumptions that can be made about the underlying nature of undesirable behavior in P2P systems. Undesirable behavior can be predominant, given it can arise due to flaws in the design or implementation of the system. This complicates the use of automated techniques widely adopted in the detection of anomalies of general network traffic such as [33–35], which assume most data-points are normal, and which identify anomalous behavior by detecting sudden and significant deviations from normal values.

Consequently, our methodology employs a combination of data-mining techniques, and manual inspection through domain knowledge. The behavior of individual hosts is characterized with respect to a wide range of metrics over multiple time samples. The set of metrics chosen is broad, since there is limited a priori knowledge of the types of undesirable behavior that may be present. Standard clustering algorithms are utilized to identify homogeneous groups of samples. Finally, the clusters are manually inspected, correlated and interpreted using domain knowledge to identify undesirable patterns.

Our methodology reveals several interesting findings, both confirming already known types of undesirable behavior of P2P systems, as well as highlighting new undesirable patterns. Some of our most relevant findings include:

• We show evidence of real DDoS attacks being conducted on DNS servers by exploiting P2P systems.

• We show that stale membership information and presence of hosts behind Network Address Translators (NATs) can result in the failure of 15% of TCP connections and 18% of UDP flows incoming to the PoP. This may hurt peer performance, introduce unnecessary traffic, and may waste significant computation resources of state-full network devices, such as firewalls or NAT boxes.

• We show instances where maliciously deployed servers can subvert the performance of hosts participating in the P2P system.

While much of our analysis is conducted with Kad, and KadU given their predominant usage in the ISP network, we extended the analysis to consider other popular P2P systems, BitTorrent [3] and DC++ [36]. Given these systems are not widely used in the network, our analysis is conducted on a separate one-week long trace in which sufficient data samples are present. Our analysis exposes undesirable behavior in these systems as well. Overall, our results shed light on the prevalence and impact of undesirable behavior in P2P systems, and demonstrate the potential of a systematic traffic-analysis approach in uncovering such behavior.

2.1 Methodology Overview

The methodology we propose in this paper seeks to infer undesirable behavior of P2P systems, by identifying possibly atypical traffic. Our methodology may be viewed as consisting of the following steps, as depicted in Figure 2.1.

In our analysis, we assume that data is collected at the edge of a network, for instance at the edge of an enterprise network. We assume that flow-level records of all UDP and TCP data traversing the network edge is available. While well-known flow level loggers such as Cisco NetFlow [37] can be used to generate flow records, a key requirement for our study is that flow-level records are classified based on application, and flows corresponding to the P2P system of interest are clearly identifiable. Several techniques have been developed for classification of traffic as P2P (for instance [38–44]), which may be leveraged. In this paper, we use datasets

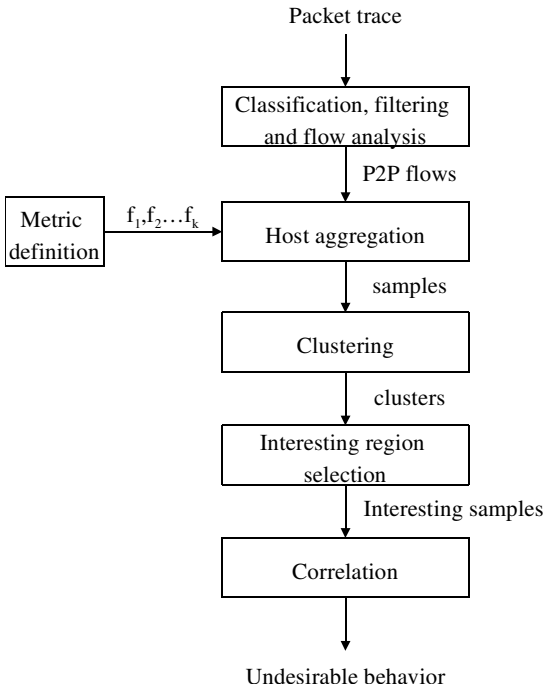


Fig. 2.1. Schematic overview of the proposed methodology.

where traffic is classified using Tstat [45], a passive sniffer with deep packet inspection (DPI) capabilities. Raw packets are sniffed from the link of interest, flows are passively rebuilt, and classification is performed in an online fashion based on the application layer payload.[1]

While we begin with per-flow measurement information, we aggregate this information to capture per-host behavior. We conduct our analysis at the host level since our goal is to characterize peer activity - for instance, we are interested in capturing peers that exhibit undesirable behavior such as searching aggressively, or generating large amounts of traffic. We capture host behavior using several metrics such as the number of active flows, the total number of received connections, and the average size of packets sent and received. For any given host h, and in a given time window $[i\Delta T, (i+1)\Delta T]$, and for each metric $f_m,\ m = \{1, 2, \ldots, k\}$, a sample of the metric $f_m(h, i)$ is obtained for that time window. We study host behavior in various time

[1]In our context, encrypted payload has not been a major issue, but in general one approach to deal with it is using behavioral classifiers.

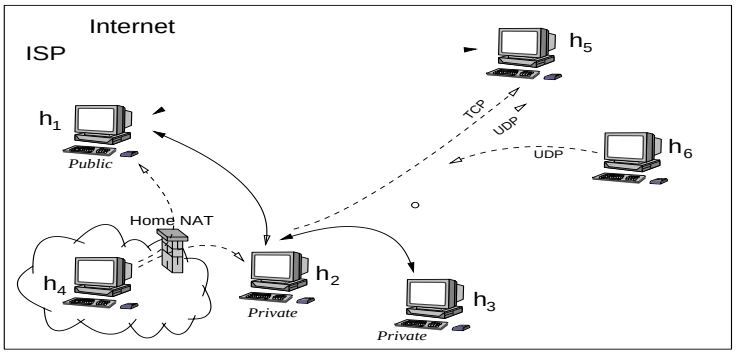


Fig. 2.2. Schematic representation of possible connections from hosts in the MiniPop versus other hosts.

windows, since the host might be demonstrating normal behavior overall, but might exhibit interesting behavior for certain periods of time.

The next step consists of detecting interesting, and potentially undesirable patterns of behavior that hosts may exhibit. To achieve this, samples corresponding to a given metric are fed to a clustering algorithm. In particular, we adopt a density-based clustering algorithm - clusters are regarded as regions in the data space in which the objects are dense, and which are separated by regions of low object density (noise). As output of this step, we get, for each metric, clusters of samples $\{f_m(h,i)\}$. Through manual inspection and domain knowledge, clusters are labeled as normal or possibly interesting. Interesting samples are then correlated across hosts to identify if they correspond to particular hosts, or are spread across multiple hosts. In addition the analysis may rely on correlating interesting behavior across multiple related metrics.

2.2 Datasets

Real traffic traces are collected from a main broadband telecommunication ISP in Europe, offering telecommunication services to more than 5 millions families. Thanks to its *fully IP* architecture, and the use of both Fiber to the Home (FTTH) and Digital Subscriber Line (xDSL) access, the ISP offers converged services over a single broadband connection. No PSTN circuit is offered to end-users, so that only IP

connectivity is adopted to offer data, VoIP and IPTV services over the same infrastructure [46]. The very peculiar mix of FTTH and high-quality ADSL access makes the ISP the leader in providing high speed access in its country, and the preferred ISP among high-end users.

2.2.1 Setup and Trace Collection

A Metropolitan Area Network (MAN) Ethernet-based architecture is adopted in the last mile. Residential and small business customers are connected to a Home Access Gateway (HAG), which offers Ethernet ports to connect PCs, the set-top box and traditional phone plugs. In case of FTTH access technology, HAGs are connected to Ethernet switches in the building basement, which are then interconnected to form Gigabit Ethernet rings. Rings are terminated at the so called MiniPoP routers, which offers connectivity to the ISP backbone. Customers are offered a 10Mbps half-duplex Ethernet link. In case of ADSL access, the HAGs are connected by the DSLAM to backbone routers. Customers are offered 1024kbps upstream and 6Mbps or 20Mbps downstream links.

Addressing and NATs: Both private and public addresses are offered to end users, as shown in Figure 2.2. A small number of hosts (for instance, host h_1), have public IP addresses and these hosts have unrestricted end-to-end IP connectivity with other Internet hosts. The vast majority of hosts (for instance hosts h_2 and h_3) are assigned private IP addresses. Whenever such hosts communicate with hosts in the external Internet (for instance, h_5 and h_6), the data communication involves traversal of an ISP-wide NAT. Note however that plain end-to-end IP connectivity is offered among hosts inside the ISP network, and communication between hosts inside the ISP (for instance, h_1, h_2, and h_3) does not involve NAT traversal. At the peering point, a Full-Cone NAT service [47] is implemented. This forbids any TCP connection initiated from the external Internet. However, it is possible that UDP flows initiated from the external Internet are permitted. In particular, once a host behind a full-cone NAT

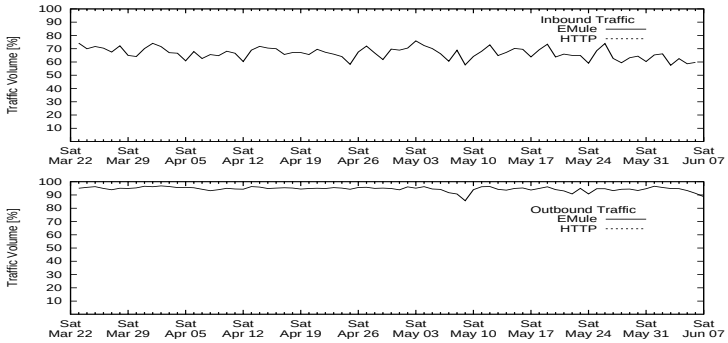


Fig. 2.3. Traffic volume shares in the MiniPoP. The top plot reports the inbound traffic, while the bottom plot reports the outbound traffic. Only HTTP and eMule traffic is reported. On the bottom plot, HTTP traffic is small and not visible.

(for instance, host h_3) sends a UDP packet to the external Internet, it can receive UDP packets on the port from any arbitrary external host, (for instance, h_6). Finally, in addition to the ISP-wide NAT, individual users (for instance, host h_4) may also employ home NAT boxes. Host h_4 cannot be contacted by any host (unless proper configuration at the home NAT is provided).

Trace Collection: Traces have been collected at a MiniPoP router during March and April 2008. A probe PC running Tstat was used to analyze in real time all the packets going to and coming from all the users in the MiniPoP, and produce a flow level log that has then been post-processed later[2]. In this paper we report results obtained focusing on a subset of the dataset, corresponding to about 25 hours, or about 2TB of information. About 2,200 different hosts were active in the MiniPoP, exchanging packets to about 782,000 different hosts in the Internet. Few hosts in the MiniPoP are using public IP addresses, which correspond in general to servers installed in small offices.

[2]A flow is identified by the traditional 5-tuple. In case of TCP, a flow starts when the SYN packet is observed. If the three-way-handshake is (not) completed, then the TCP flow is said to be *(un)successful*. In case of UDP, a flow starts when the first packet is observed. If (no) packet is observed in the reverse path, then the UDP flow is said to be *(un)answered*. Flows end after no packets have been observed for 10 minutes.

2.2.2 Description of P2P Systems

The most popular P2P system used among the users in the ISP is eMule [32], a widely deployed file sharing application. To demonstrate this, Figure 2.3 shows the byte-wise traffic volume percentage as measured during about three months. The top and bottom plots report results considering inbound and outbound traffic respectively (i.e., bytes destined to/sourced from a host in the MiniPoP). The two most popular protocols are HTTP and eMule, with other protocols accounting for no more than 10% of traffic. In particular, eMule accounts for about 60-70% of traffic on the inbound traffic, while it has a share of more than 95% on the outbound volume. HTTP traffic is predominant only in the inbound traffic, since hosts in the MiniPoP act as clients.

We focus our analysis on eMule traffic given its large predominance in the dataset. eMule supports both a centralized architecture, referred to as *eMule network*, and a DHT system, referred to as *Kad network*. In particular, eMule servers do not store any files, but they only maintain a database of files and users per file. The Kad network is a large-scale DHT-based system based on Kademlia [48]. A Kad client looks up for both content and peers sharing content using the DHT instead of relying on the eMule servers. Once a client has found a peer that is sharing the desired file, direct connections are used to download/upload the actual data using end-to-end TCP connections; communication with the eMule server goes preferentially over TCP, while Kad relies on UDP only.

The original eMule/Kad networks have mechanisms in place to identify clients behind NAT, and limit the performance of such clients when they try to download content. This impacts the performance of hosts with private IP addresses in the ISP, since a NAT is traversed when communicating with hosts in the Internet, e.g., eMule servers. Given that the large majority of ISP hosts have been given private IP addresses, the performance of eMule is severely limited. Therefore, a community of ISP users modified the original eMule client [49] to form a closed P2P network that we call *KadU network*. The network is closed in the sense that all KadU clients

belong to the ISP network only. The Kad protocol has been modified, so KadU messages can only be exchanged among peers running the modified eMule version and using IP addresses actually used by the ISP. This ensures that a KadU client cannot operate in the Internet. Similarly, the peer selection mechanism has been modified to preferentially connect to other KadU clients. The KadU peers perform search operation on the KadU network by default, rather than relying on server-based search as in the default eMule configuration. No changes have been made to the eMule part, so that both server and P2P protocols are the same as in the original eMule, and the modified eMule client and original eMule client can perfectly interoperate.

Besides avoiding the NAT issues, running the modified client has several advantages. Indeed, it is desirable to download content from other clients in the ISP because the large percentage of hosts connected by FTTH access guarantees much higher upload capacity than the typical one offered by ADSL providers. Furthermore, given that all the ISP peers are in the same European country, the content that is available in the P2P system matches the interest of the community, and it is easier to trade content in the closed network than in a worldwide network. For these reasons, clients in KadU typically see much better performance than the one typically achieved by clients in the Kad network.

2.2.3 Preliminary Traffic Analysis

In the considered 25 hours dataset (on a Wednesday), we identified 478 clients running KadU inside the MiniPoP, and exchanging traffic with about $229,000$ KadU clients outside the MiniPoP. For Kad, we identified 136 clients which were exchanging packets with about $300,000$ clients in the Internet. Table 2.1 presents details on the trace characteristics for the Kad, KadU and eMule systems. Most of the paper will focus on these systems. But, in Section 2.8, we extend the analysis to consider other systems.

Table 2.1
General Statistics of P2P Traffic

Direction	TCP		UDP	
	eMule		eMule/Kad/KadU	
	succ. connections	Bytes	flows	Bytes
MiniPoP to ISP	264k	512G	4.7M	820M
MiniPoP to Internet	377k	80G	412.7k	58M
ISP to MiniPoP	174k	341G	3.8M	735M
Internet to MiniPoP	0	0	208k	35M

Knowing the address space allocation for the ISP and for the MiniPoP, we are able to classify all hosts as being in the MiniPoP, in the ISP (but not in the MiniPoP) or in the Internet. We leverage this classification in Table 2.1. Each row provides statistics about traffic exchanged between hosts in two classes. The second and third columns give the number of successful TCP connections classified as eMule, and the amount of bytes they carried. The last two columns show similar numbers for UDP flows classified as eMule, Kad or KadU. For example, the MiniPoP to ISP row reports that (i) there was a total of $264k$ eMule TCP connections initiated from inside the MiniPoP to clients inside the ISP, which carried a total of $512GB$ of data; and (ii) around 4.7 million UDP flows (classified as eMule, Kad, or KadU) were initiated in the same direction, and about $820MB$ of data was exchanged.

From this table, we see that: (i) the bytes exchanged between hosts within the ISP is much larger than the bytes going to the Internet, for both TCP and UDP. This is due to the extensive usage of the KadU network, and its efficiency in localizing traffic communication to within the ISP; (ii) there is a non-negligible amount of TCP and UDP traffic exchanged with the Internet. This is because the use of Kad clients is still prevalent. In addition, even clients that use the KadU network may need to rely on eMule if the content cannot be located within the ISP; and (iii) there are no TCP connections initiated from the Internet to the MiniPoP, but there are UDP flows

though. This is due to the full-cone NAT at the edge of the network, as previously explained in Section 2.2.1.

2.3 Traffic Classification

In this section, we describe the mechanisms we use to identify P2P traffic. We also present our methodology to aggregate flow metrics into samples per host.

2.3.1 Selecting Flows of Interest

In order to correctly identify eMule and Kad/KadU traffic, we employed an approach described in [45] which implements deep packet inspection (DPI), and produces a flow level log as output. When a new flow is identified, the DPI classifier looks at the application layer payload to identify a set of well-known protocols. All eMule and Kad/KadU protocol messages are included, and manual tuning has been adopted to guarantee conservative classification. While the performance of the DPI is out of the scope of this paper, we manually verified that the false positive and false negative probability is practically negligible.

The output of the classification and flow analysis phase is a flow level log, in which each flow that has been observed and classified as eMule, Kad or KadU is listed, along with a list of measurements. In particular, in this paper we exploit the following per-flow information: (i) *flow id* defined as <src_ip, src_port, dst_ip, dst_port, protocol_type>; (ii) *first and last packet time*; (iii) *number of sent and received packets*; (iv) *number of sent and received bytes*. Note that the above information can be easily derived by any flow level logger, such as NetFlow [37], running directly at routers.

2.3.2 Aggregating Flows into Host Samples

As described in Section 2.1, we aggregate flow measurements into host metrics $f_m(h, i)$. A sample f_m is obtained for host h at every time slot i of size $\Delta T = 5$

minutes. The latter choice enables us to track changes in host behavior over the order of minutes, and it is unlikely for host behavior to significantly shift over this period.

Given that clients could be part of either the Kad or KadU network each with very different properties, we would like to study each system in isolation by separately aggregating Kad and KadU flows into samples. However, while the two networks differ in the UDP protocol used in the Kademlia DHT, both employ identical TCP-based protocol on the data path, e.g., to exchange content. As a consequence, the DPI classification can successfully distinguish UDP-based control messages, but the TCP-based data flows are classified identically as eMule.

To handle this, we adopt the following heuristic. Consider a time slot i and a host h. If UDP flows are present, then we classify the sample as either Kad or KadU based on the classification of UDP flows. All TCP-based metrics are then classified accordingly. In the dataset, there are $12,963$ KadU samples and $1,519$ Kad samples. It is possible that a time slot includes both Kad and KadU - however, there are only 35 such samples, so that we can simply discard them.

Finally, it is possible to have samples with neither Kad nor KadU flows, but exclusively eMule TCP connections. There are $1,200$ such samples, most of which are due to 4 hosts running the centralized eMule protocol only. We do not consider them further.

2.4 Metrics

In this section, we present the list of the metrics considered in this paper, which is summarized in Table 2.2. All the selected metrics are very simple and intuitive metrics. Some of them have been previously proposed for both traffic characterization and classification considering both P2P systems, and traditional client/server applications. Some are specifically defined considering the scenario we are facing, e.g.,

to highlight eventual Kad and KadU dissimilarities, or to pinpoint possible atypical behavior.

If not otherwise specified, each metric is evaluated separately for UDP and TCP flows, given that the considered systems make use of both protocols. When needed, *TCP* (*UDP*) will be appended to the metric name, as appropriate. For relevant metrics, we consider the location of the flow initiator as either being inside or outside the MiniPoP. For ease of notation, metrics involving flows initiated inside (outside) the MiniPoP will be prepended with the term *inout* (*outin*) followed by the metric name.

In Section 2.4.1 we consider the metrics general to all flows first, and then the metrics to which initiator location is specified are detailed in Section 2.4.2.

2.4.1 Metrics Independent of Flow Initiator

These metrics consider various measurements that do not depend on the location of the flow initiator. We group them in two categories, *Flows*, which include per-flow basic statistics and *Data transfer*, which includes data exchange related metrics. Given a host h in the MiniPoP and a time slot i, we have:

• **Flow** related metrics: (i) *avg-duration* is the average duration of flows started during time slot i; (ii) *live-conn* is the total number of flows that were active during time slot i. This includes flows that have started in the current time slot and flows that started in previous time slots and are still active in the current one; (iii) *fract-incoming-conn* is the ratio of flows initiated from the outside to the total number of flows.

• **Data Transfer** related metrics: (i) *bps-rcvd* and *bps-sent* are the average bits per second (referred to as *bps*) received and sent respectively; (ii) *avg-pkt-size* is the average size of packets sent and received; (iii) *ratio-bytes-sent-to-rcvd* is defined as $\frac{B\_sent - B\_rcvd}{B\_sent + B\_rcvd}$, where *B_sent* (*B_rcvd*) is the total amount of bytes sent (received). *ratio-*

bytes-sent-to-rcvd $= -1$ for hosts receiving data only, while *ratio-bytes-sent-to-rcvd* $= 1$ for hosts that send data only.

2.4.2 Metrics Dependent on Flow Initiator

In this case, four categories of metrics have been selected:

• **Flow** related metrics: *total-conn-attempts* is the total number of flows initiated (inout) or received (outin). This includes both successful and unsuccessful connections when considering TCP, and both answered and unanswered flows when considering UDP.

• **Destinations** related metrics: (i) *avg-conn-per-IP* is the ratio of all flows to the number of distinct destinations. A similar metric was used in [43] for P2P traffic classification, in which the authors showed that it is rare that P2P clients open concurrent connections to other peers; (ii) *total-peers* is the total number of distinct peers; (iii) *dest-ports* is the total number of distinct destination ports; (iv) *1024-dest-ports* is the total number of distinct reserved destination ports, i.e., ports from 0 to 1024. Since reserved ports should not be used by non standard application, we include this metric to highlight possible abuse.

• **Failures** related metrics: (i) *failure-ratio* is the ratio of unsuccessful TCP flows to total TCP flows [3] (ii) *fract-unanswered-appl* is the fraction of TCP flows where the TCP handshake is successfully completed, but the destination never sends any application data for the duration of the connection. (iii) *fract-unanswered* is the fraction of unanswered UDP flows to total UDP flows.

• **ISP** related metrics: *ISP-to-Internet-ratio* is the ratio of the number of peers within the ISP that are contacted to the total number of contacted peers.

[3]Note that unsuccessful TCP flows cannot be classified as eMule, since no payload can be inspected. Hence, we take a conservative approach and only consider as eMule related failures those that are directed to the default eMule port.

Table 2.2
List of Metrics

Metrics Independent of Flow Initiator	Flows	*avg-duration*
		live-conn
		fract-incoming-conn
	Data Transfer	*bps-rcvd*
		bps-sent
		avg-pkt-size
		ratio-bytes-sent-to-rcvd
Metrics Dependent on Flow Initiator [inout, outin]	Flows	*total-conn-attempts*
	Destinations	*avg-conn-per-IP*
		total-peers
		dest-ports
		1024-dest-ports
	Failures	*failure-ratio* [TCP only]
		fract-unanswered-appl [TCP only]
		fract-unanswered [UDP only]
	ISP	*ISP-to-Internet-ratio*

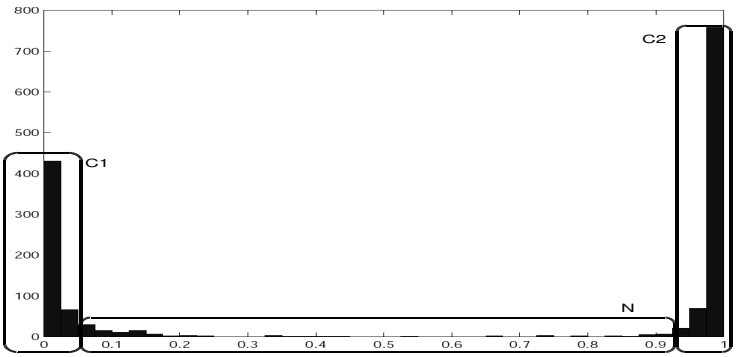


Fig. 2.4. *outin-fract-unanswered-UDP* is an example of multiple cluster metric.

2.5 Identifying Unwanted Behavior

Our goal is to identify undesirable behavior of P2P systems. The key challenge we faced in our study is that an exhaustive list of potential undesirable behavior

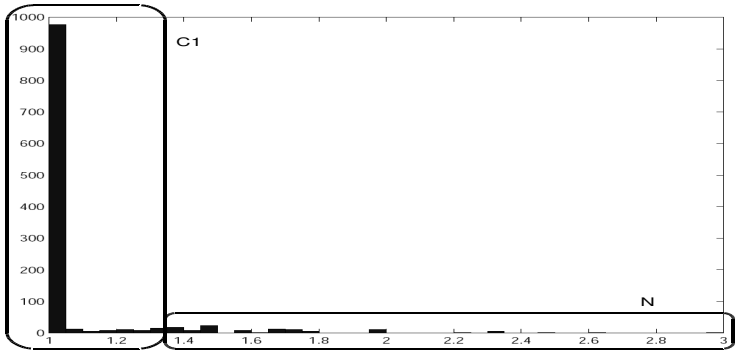


Fig. 2.5. *inout-avg-conn-per-IP-TCP* is an example of single cluster metric.

is not available to us a priori. Moreover, the intrinsic heterogeneity of P2P traffic makes it hard to clearly distinguish undesirable behavior from normal usage. Consequently, our methodology employs a combination of data-mining techniques, and manual inspection through domain knowledge.

As a first step, we employ clustering algorithms [50] to obtain a set of *coarse* clusters of the data. Without the need of any training data, clustering algorithms aim at partitioning the data set into subsets (or "clusters") so that samples in the same subset share common traits, i.e., they are close to each other according to a notion of distance. Clustering algorithms are often useful for outlier detection, where outliers may emerge as small clusters far apart from the others. As a second step, we extensively resort to domain knowledge and manual inspection to interpret the clustering results, zoom in on interesting patterns, and identify undesirable behavior.

2.5.1 Density-based Clustering

Among clustering algorithms, density based clustering uses the concept of dense region of objects. In such schemes, dense regions of objects are considered a cluster and low density regions are considered as noise. In particular, we selected DBScan [51], since it is well known and offers several advantages: it automatically determines the number of clusters (contrary for example to the k-means algorithm); it is robust to

noise, i.e., isolated samples; and finally it does not have any bias versus any cluster shape.

Intuitively, DBScan groups together points that fall in a dense region of the metric space. Given a point in the data set, density is estimated as the number of points within a specified radius of that point. There are three types of points: (i) **core point** is a point that has more than *Minpts* around it within a distance $d \leq \epsilon$; (ii) **border point** is a point that is within a distance ϵ of a core point but is not a core point; (iii) **noise point** is any point that is neither a core point nor a border point. With these definitions in mind, DBScan puts two core points in the same cluster if they are within a distance ϵ of each other. Also, a border point within distance ϵ of a core point is put in the same cluster as the core point. Finally, noise points are labeled as such.

For each metric m, we consider the set of all samples $F_m = \{f_m(h, i)\}$ collected during the desired observation period, for each host h and for all time slots i. In this paper, for a 5 minutes observation period and a 25 hour trace, $0 \leq i \leq 300$. We apply clustering algorithms to each metric individually, and define the distance between two samples as simply $d = |f_m(h_1, i_1) - f_m(h_2, i_2)|$. We choose to apply clustering on individual metrics rather than on multidimensional spaces for several reasons. First, each metric sample distribution is generally very skewed, which makes clustering difficult per se. When considering a multidimensional space obtained as the Cartesian product of skewed metrics, the result of clustering is hard to predict and control, e.g., to impose a coarse clustering. Further, distance in multidimensional space may be difficult to define, since each metric have very different support, e.g., $x \in [0, 1]$ and $y \in [0, \infty)$ make it hard to appreciate the spread on the x dimension. Note that this is typical of our scenario, e.g., considering metrics like *outin-fract-unanswered* and *avg-duration*. Although dimensional reduction and normalization techniques exist, the outcome from them may be difficult to control and interpret. Finally, possible undesirable behavior can be already identified when considering a single metric, while

the correlation between undesirable behavior across different metrics can be later checked exploiting the domain knowledge of the targeted scenario.

We illustrate the operation of the DBScan algorithm, and the impact of the parameters *Minpts* and ϵ with an example. Consider Figure 2.4 which shows the histogram of *outin-fract-unanswered* for UDP traffic considering the Kad dataset. More than 650 samples (around 36%) fall in the range $[0, 0.08]$, while more than $1,300$ (around 58%) samples fall in the range $[0.98, 1]$.

Table 2.3 reports the DBScan result when applied to the dataset in Figure 2.4, for different values of *Minpts* and ϵ parameters. Each cell shows the number of clusters produced by DBScan, and the fraction of the samples that are classified as noise. For instance, for ϵ=0.1, and *Minpts*=40% of the total samples, there is 1 cluster, with 40.8% of the samples classified as noise. For large values of *Minpts* (60%) we see that 0 clusters are produced for most ϵ values, and all 100% of the samples are classified as noise. This is because no point has a sufficiently large neighborhood or density to be classified as a core point. As we decrease *Minpts* however, the noise region decreases, and clusters emerge. For ϵ=0.1, and for *Minpts* 20% or lower, 2 clusters are always identified, which matches our intuition from the Figure. We observe that DBScan is relatively robust to the input parameter setting in our scenario, and that there are several parameter settings that can achieve a reasonable coarse clustering.

We employ a simple iterative search heuristic to identify a value of *Minpts* and ϵ that can achieve a reasonable coarse clustering. Our heuristic seeks to obtain a clustering result with noise region that is non empty but not too large, e.g., a small percentage of samples. The reason for requiring a small number of samples to be classified as noise is to avoid cases where many smaller clusters are merged into one larger cluster with no noise region (for instance, ϵ=0.5, *Minpts*=20% in Table 2.3), or to prevent clusters being formed with a small number of points. We start with $\epsilon = 0.1$, *Minpts*=50% and keep decreasing *Minpts*, until the resulting noise falls in the target region. Then, ϵ is decreased until the noise region is exceeds the target value. Large ϵ enlarge clusters adding noise points and eventually merging clusters,

Table 2.3
DBScan Sensitivity

		Minpts					
		1%	5%	10%	20%	40%	60%
ϵ	0.01	3(4%)	2 (15%)	2 (16%)	2 (19%)	1 (49%)	0 (100%)
	0.05	2 (1.1%)	2 (2.3%)	2 (3.7%)	2 (4%)	1 (41%)	0 (100%)
	0.1	2 (0.5%)	2 (0.9%)	2 (1%)	2 (1.2%)	1 (40.8%)	0 (100%)
	0.2	2 (0%)	2 (0.1%)	2 (0.1%)	2 (0.2%)	2 (0.3%)	0 (100%)
	0.5	1 (0%)	1 (0%)	1 (0%)	1 (0%)	1 (0%)	1 (33%)

while small ϵ results in possible splitting of clusters that might not be of interest. The results we present employ a target noise region of 6%, but we have found that DBScan is relatively robust to the choice in our scenario, and any value in the range 2-10% would provide very similar results.

2.5.2 Interesting Region Selection

After getting the output from DBScan, intuitively we could consider the samples in the noise region to be the interesting ones. However, undesirable behavior could be so prevalent to form a whole cluster, and hence, only considering the noise points may cause information loss. We therefore believe the interesting region should be selected based on domain knowledge from the network operator, P2P developer, or the end-user. In particular, applying DBScan to each metric, two possible cases are obtained: (i) metrics exhibiting a single cluster and a noise region; and (ii) metrics exhibiting multiple clusters and one or more noise regions.

In cases where the metric exhibits a single cluster, the interesting region typically coincides with the noise region. To illustrate this, consider Figure 2.5, which shows the histograms of values taken by the *inout-avg-conn-per-IP-TCP* metric considering the Kad dataset. As shown in the Figure, when DBScan is employed, a single cluster ($C1$) is produced which includes all samples in the range $[0, 1.4)$, and a noise region,

including samples in $[1.4, 3]$. The noise region is interesting since eMule clients are not expected to open more than one TCP connection with the same host. We have further investigated the samples in this region, and have found them to be due to hosts being attacked by *fake servers*, as we explain in Section 2.7.3. We also note that for such metrics, DBScan enables choosing the thresholds for the noise region appropriately - simpler heuristics like selecting the top or bottom 10% of samples as showing interesting behavior do not take the distribution of data into account and may not be as effective in general.

In cases where the metric exhibits multiple clusters, the choice of interesting region can only be supported by the knowledge of the considered application, metric, and scenario. For example, in Figure 2.4, DBScan identifies two clusters $C1, C2$ and a noise region, which confirms the visual intuition. In this case, we consider the interesting region to include cluster $C2$, since it represents samples in which most externally initiated UDP connections are unanswered. We analyze this in further detail in Section 2.7.2. More generally, the interesting region could include a combination of multiple clusters and noise regions.

2.5.3 Correlation Across Interesting Samples

Having identified the interesting samples for each metric, we employ several simple heuristics to identify correlations across the samples, which in turn can aid making inferences of undesirable behavior. We describe these below:

• *Hosts dominating interesting samples:* We consider the number of distinct participating hosts (or IP addresses) to which the interesting samples for a given metric correspond. If the entire interesting cluster for a metric can be attributed to a small number of participating hosts, it is an indication that the interesting behavior is a property of those hosts. If however the interesting cluster is spread among several hosts, it is an indication that the interesting behavior is more general and not due to a few hosts.

• *Correlations across metrics:* We consider whether interesting behavior seen across multiple metrics are correlated, and are due to the same underlying cause. We typically rely on domain knowledge to determine such correlations. For instance, in Section 2.7.2, we used domain knowledge to reason that a large number of interesting samples seen in four of the metrics we considered were directly related. Likewise, in Section 2.7.4, we isolate hosts that generate a large number of samples in the interesting region across multiple metrics, and use these observations to reason about the potential behavior of the hosts.

2.6 Results

In this section, we present high level characteristics of the Kad and KadU networks. We then discuss results with DBScan and the selection of interesting regions for various metrics.

2.6.1 High Level Characteristics of Systems

In this section we provide high level background on the Kad and KadU networks, highlighting key differences between them.

• *In contrast to Kad, KadU traffic typically stays within the ISP*: Kad clients mostly contact peers in the Internet while KadU clients mostly contact peers within the ISP. The *inout-ISP-to-Internet-ratio* metric was 1 for almost all KadU samples when UDP traffic was considered. Interestingly, KadU clients did contact more peers in the Internet when TCP traffic was considered. This was not entirely expected and will be further investigated in Section 2.7.4.

• *In contrast to Kad, KadU clients use default UDP/TCP ports:* When the *dest-ports* metric is considered, the median value of KadU samples is 1, while it is 33 for Kad. This is because KadU clients run in a friendly environment in which no throttling is imposed on P2P traffic by the ISP. Hence there is no need to try masquerading P2P traffic by using random ports. On the contrary, Kad clients run in the Internet,

where ISPs may block P2P traffic, and there is a greater tendency for users to adopt random ports (and possibly protocol obfuscation).

• *In contrast to KadU, Kad clients see almost no incoming TCP traffic due to a NAT at the edge of the ISP*: The metric *fract-incoming-conn-TCP* is equal to 0 for almost all Kad samples, while it has a bell distribution for KadU samples. The reason for this is that there is a NAT at the edge of the ISP, as discussed in Section 2.2.2, which forbids incoming TCP connections from the Internet. Interestingly, Kad clients can still receive UDP flows initiated in the Internet. This is because the NAT at the edge of the ISP is a Full Cone NAT.

• *KadU clients exchange much more data, with a prominent seed-like behavior*: When the *bps-rcvd* and *bps-sent* metrics are considered, the 90%*ile* for KadU samples is 164kbps and 674kbps respectively. In contrast, the 90%*ile* for Kad samples is only 36kbps and 54kbps. The much higher performance in KadU is due to the effectiveness of the optimizations in the KadU client, as well as the large installation of high-speed FTTH users in the ISP. Further, we noticed that KadU clients present a predominant seed-like behavior (for example, the 90%*ile* of the *bps-sent* metric is 4 times the 90%*ile* of the *bps-rcvd* metric). We believe this may be attributed to the high-speed upload bandwidth of the FTTH users in the ISP.

2.6.2 Interesting Region Selection

In this section, we present the results of applying DBScan to our dataset and the interesting regions we identified based on manual inspection. For single cluster metrics, we simply selected the noise region as interesting, as discussed in Section 2.5.2. Hence, we focus on metrics that involved multiple clusters.

The sensitivity of the interesting regions was tested in our dataset by splitting the 25 hour trace into two halves and then running DBScan over each portion, as well as running DBScan over the entire trace. One half corresponded to day-time activity and the other half to night-time activity. For single cluster metrics, the

Table 2.4
Metrics with Multiple Clusters - Kad

Name	C/N	Range	percent	Explanation
avg-pkt-size-TCP [*Bytes*]	**C1**	[55 250]	16.28%	Primarily control
	C2	[726 955]	17.05%	
	C3	[956 1,348]	61.66%	
	N	[296 723]	5.01%	
outin-fract-unanswered-UDP	C1	[0 0.08]	36.14%	Left group or home NAT
	C2	[0.93 1]	58.04%	
	N	[0.08 0.92]	5.82%	
ratio-bytes-sent-to-rcvd-UDP	**C1**	[-1 -0.63]	17.98%	Left group or home NAT
	C2	[-0.62 0.45]	78.06%	
	N	[0.5 1]	3.96%	
inout-1024-dest-ports-UDP	C1	[0 0]	73.72%	DDoS attack
	C2	[1 1]	18.12%	
	N	[2 6]	8.16%	
ratio-bytes-sent-to-rcvd-TCP	**C1**	[-1 -0.62]	13.49%	Selfish hosts
	C2	[-0.4 0.62]	55.06%	
	C3	[0.62 1]	27.66%	
	N	[-0.62 -0.41]	3.79%	

results of clustering were similar, with only marginal changes to clusters' width and noise regions. The multiple clusters metrics, on the other hand, had minor changes in clusters for some metrics, but overall, the final trend of the interesting regions was preserved. In the rest of the section, we focus on clusters obtained using the entire trace.

Kad

In this section we present results for Kad, which are reported in Table 2.4. The first column shows the metric name and transport protocol. The second column identifies a region as a cluster or noise, in which we highlight the interesting one

Table 2.5
Metrics with Multiple Clusters - KadU

Name	C/N	Range	percent	Explanation
inout-ISP-to-	**C1**	[0 0.62]	51.77%	Traffic
Internet-ratio-TCP	C2	[0.62 1]	48.23%	within ISP
outin-fract-	C1	[0 0.22]	33.82%	Left group
unanswered-UDP	**C2**	[0.8 1]	60.21%	or home
	N	[0.22 0.8]	5.97%	NAT
ratio-bytes-	**C1**	[-1 -0.71]	49.42%	Left group
sent-to-rcvd-UDP	C2	[-0.25 0.37]	44.69%	or home
	N	[0.37 1]	5.89%	NAT
fract-incoming-	C1	[0 0.32]	18.85%	Left group
conn-UDP	**C2**	[0.57 1]	75.58%	or home
	N	[0.32 0.56]	5.57%	NAT
outin-failure-	C1	[0 0]	62.33%	Left group
ratio-TCP	**C2**	[1 1]	33.06%	or home
	N	[0.01 0.97]	4.61%	NAT
ratio-bytes-	**C1**	[-1 -0.36]	5.61%	Selfish
sent-to-rcvd-TCP	C2	[-0.36 1]	91.8%	hosts
	N	[-0.62 -0.36]	2.59%	

in bold. The third column shows the actual range of sample values in each cluster, while the fourth column reports the percentage of the samples that are in the cluster. Finally, the fifth column shows the explanation why the selected region is interesting.

We summarize key observations as follows:

• *Samples with predominantly control messages:* The first row of Table 2.4 shows the clusters found by DBScan for the *avg-pkt-size-TCP* metric. There are three clusters for this metric. Cluster $C1$ contains 16.28% of the samples, and it refers to samples whose flows exhibited "small" average packet size. $C3$ corresponds on the contrary to "large" average packet size, while $C2$ corresponds to a cluster with "mid-sized" packets. These clusters correspond to hosts exchanging mostly control messages,

mostly data messages and a mix of control and data messages respectively. Among those, cluster $C1$ is interesting since it corresponds to samples where only control messages were exchanged. This could be for benign reasons, for instance, a host that does not download or upload content. But it could also indicate undesirable behavior, for instance a host being part of a P2P botnet. One potential indication of malicious activity is a host that is persistently sending only control messages in all its samples. We did not find evidence of this in our trace, leading us to believe there was no malicious activity.

• *Samples of peers that do not reply to incoming requests*: When the *outin-fract-unanswered-UDP* metric is considered, it is striking that there is a cluster ($C2$) with samples in the range 0.93 to 1 and which includes 58.04% of the samples. This cluster corresponds to samples where almost every UDP flow initiated from the outside is unanswered, indicating potentially anomalous behavior. Likewise, considering metric *ratio-bytes-sent-to-rcvd-UDP*, cluster $C1$ corresponds to samples where UDP packets are mostly received, indicating again that the peer inside the MiniPoP is not responding to external queries. These two clusters are related, and we analyze further in Section 2.7.2.

• *Communication with reserved ports*: We consider metric *inout-1024-dest-ports-UDP*, which intuition suggests should be close to 0, since P2P applications are not expected to run using a reserved port. But both cluster $C2$ and the noise region N refers to values of this metric larger than 0, accounting for 26.29% of the samples. In Section 2.7.1 we investigate this metric further and present evidence of a DDoS attack on DNS servers.

• *Selfish versus seed behavior*: Considering the metric *ratio-bytes-sent-to-rcvd-TCP*, three clusters are shown. $C1$ represents samples for hosts with selfish behavior (mostly receiving data), $C2$ represents samples for hosts that are both receiving and sending and $C3$ shows samples for hosts with seed behavior (mostly sending data). Considering P2P file sharing application, a user is expected to contribute fairly to the community, so cluster $C1$ represents possibly undesirable behavior.

KadU

In this section we focus on metrics where DBScan found multiple clusters for KadU metrics, which are reported in Table 2.5. We summarize key observations as follows:

• *Degree of communication within ISP*: Here we focus on metric *inout-ISP-to-Internet-ratio-TCP*, for which DBScan found two clusters. Cluster $C2$ corresponds to samples for which peers within the ISP are predominantly contacted. Cluster $C1$ represents those samples for which mostly peers in the Internet are contacted. The presence of cluster $C1$ is not expected since KadU is optimized for communication with peers inside the ISP. We further analyze $C1$ in Section 2.7.4.

• *Samples of peers that do not reply to incoming requests*: Like in Kad, DBScan found cluster $C2$ for metric *outin-fract-unanswered* and cluster $C1$ for metric *ratio-bytes-sent-to-rcvd* for UDP, which characterize peers that do not reply to incoming requests. In addition to these metrics, two more related metrics were found to have multiple clusters in KadU which we believe is related to the same issue. First, for the metric *fract-incoming-conn-UDP*, cluster $C2$ contains all samples for which hosts mainly receive UDP flows. We note the cluster had a prominent spike around 1, which indicates that for a large number of samples, flows are only being received. Second, for the metric *outin-failure-ratio-TCP*, cluster $C2$ corresponds to samples in which all incoming TCP connections failed. A detailed analysis is presented in Section 2.7.2.

• *Selfish versus seed behavior*: The metric *ratio-bytes-sent-to-rcvd-TCP* has a very different distribution considering KadU, showing that the large majority of peers have a seed-like behavior, which are clustered in $C2$. Also, there is a cluster of samples that suggests a subset of peers act as selfish clients, not willing to share content. We therefore select again this latter cluster as interesting.

2.6.3 Host Distribution in Interesting Region

Having identified the interesting regions, we next consider the number of distinct participating hosts (or IP addresses) to which the samples correspond. If the entire interesting region for a metric can be attributed to a small number of participating hosts, it is an indication that those hosts are particularly abnormal. If however the interesting cluster is spread among several hosts, it is an indication that the interesting behavior is more general.

Figure 2.6 shows, for each Kad metric, a point reporting the fraction of hosts that generate 90% of the samples versus the fraction of samples in the interesting region. For example, the metric *inout-1024-dest-ports* for UDP has 26% of its samples in the interesting range. 90% of these interesting samples have been generated by 24% of the hosts running Kad. We have circled those metrics for which we present key findings later. In addition, a similar plot is shown for KadU in Figure 2.7.

We focus on metrics in the right side of Figures 2.6 and 2.7, which correspond to those with a large fraction of interesting samples spread across many hosts. These metrics are the most interesting and we present and discuss our findings on them in Section 2.7. For most metrics in the bottom left of the figures, corresponding to those with interesting samples generated by a few hosts, we found the causes were usually benign and did not point to undesirable activity. However, a few cases deserve to be mentioned and we discuss them further in Section 2.7.

2.7 Key Findings

In this section, we present examples of undesirable behavior exposed by our methodology.

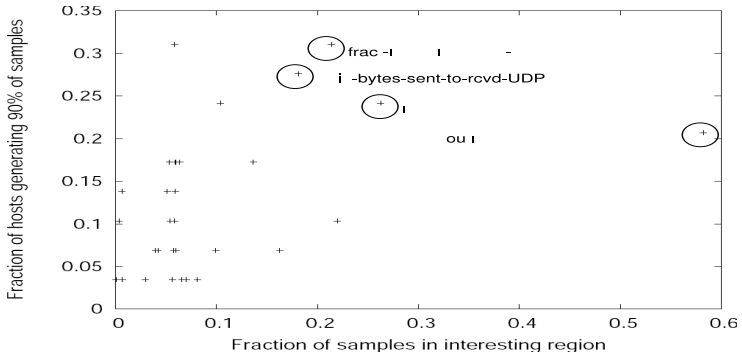


Fig. 2.6. Kad: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results.

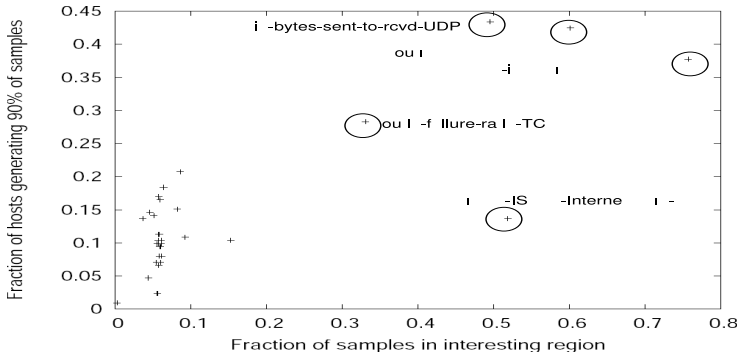


Fig. 2.7. KadU: fraction of samples in the interesting region versus fraction of clients generating them, for various metrics. Circled are metrics with most relevant results.

2.7.1 DDoS Attacks Exploiting P2P Systems

In this section, we describe our findings when studying the metric *inout-1024-dest-ports-UDP*, which was specifically added to observe undesirable traffic directed to reserved ports. Referring to Figure 2.6, Kad clients contacted peers to restricted ports for 26.25% of the samples, which is suspicious. We therefore isolated the samples in the interesting regions and looked at the destination port of those samples. It turns out that port 53 was the most common destination port, receiving $1,711$ out of $3,087$

flows destined to port 1024 or below. Note that no other port in the reserved range received more than 175 flows in total.

We further investigated and verified that flows destined to UDP port 53 were valid Kad flows, and not actual DNS flows misclassified by the DPI. Moreover, the destination IP address of the flows corresponded to actual DNS servers not managed by the ISP, but serving domains in countries far away from the location of the MiniPoP. Finally, we noticed that most of the suspicious flows were unanswered. To better highlight this, Figure 2.8 shows the fraction of unanswered flows as a function of the destination port number. Notice the spike at port 53, which indicates that this port has the highest ratio of unanswered flows of more than 90%. Other spikes refer to typical Kad ports found in the dataset.

As a final observation, we noticed from Figure 2.6 that 25% of Kad peers were generating the interesting samples for *inout-1024-dest-ports-UDP*. On further study, we found that across all these peers, while less than 2% of Kad flows initiated are destined to reserved ports, more than 30% of these flows target port 53. This indicates that the problem is not specific to a small subset of Kad peers, but is more predominant.

We believe these results show evidence of DDoS attacks on well known DNS servers exploiting the Kad network. In such an attack, a malicious client in the Kad network, spreads contact information (IP address and port) about the victim (an actual DNS server) as if it were part of the Kad network. Later, innocent clients send regular Kad messages to the DNS server. Finally, we note that there has been some awareness of such attacks in eMule technical forums [52, 53], and in fact, the top most destination in our trace was mentioned in [53] as being under attack.

2.7.2 Unnecessary P2P Traffic

Consider the 3 metrics on the top right corner of Figure 2.7. These correspond to *ratio-bytes-sent-to-rcvd-UDP*, *outin-fract-unanswered-UDP* and *fract-incoming-conn-*

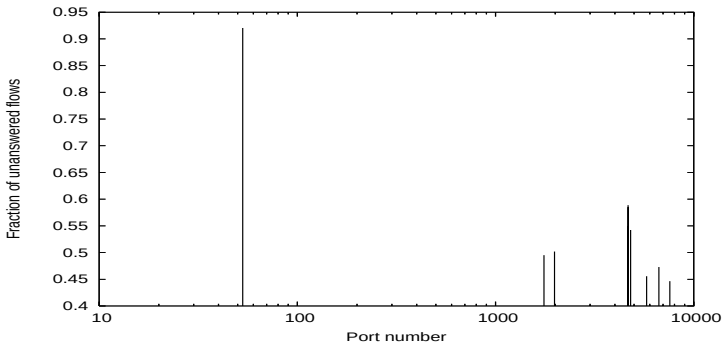


Fig. 2.8. Fraction of unanswered flows per destination port.

UDP. For each metric, 40% to 60% of the samples are in the interesting region, and about 60% of the KadU hosts are involved. The same three metrics are also highlighted in Figure 2.6 when considering Kad.

This clearly indicates some unexpected behavior, and points to a potentially significant problem. Investigating further, we observed that all metrics hint to a large number of UDP flows incoming to the MiniPoP that are never answered. In particular, 28% of UDP flows coming to the MiniPoP are unanswered, and 65% of this is due to Kad and KadU clients.

In addition, with the KadU dataset a high fraction of TCP failures is observed. Investigating further, 116,000 TCP connections coming to the MiniPoP failed, which accounts for 30% of all TCP incoming connection attempts. Roughly 50% were due to KadU. Recall that for Kad peers, no incoming TCP connection is possible due to the ISP NAT.

Having a large number of failed TCP connections or UDP flows is undesirable not only from the perspective of the introduced traffic, but also from the state that may need to be maintained by various devices in the network (such as NATs and firewalls).

We believe there are two key reasons for unanswered flows. First, some P2P participants are behind home NATs. Other peers may learn about these participants through P2P membership management mechanisms, and may (unsuccessfully) attempt to communicate with them. Second, when a host leaves a P2P system, other

peers may continue to attempt contacting it due to stale information in the P2P network.

Figure 2.9 shows an example of a host that left the P2P network, but which continues to receive packets for more than 14 hours after its departure. The top plot shows the time series for *outin-fract-unanswered-UDP*. Note the sharp transition from 0 to 1 which corresponds to node departure. The bottom plot depicts the total number of unanswered incoming UDP flows. Over 60 flows per minute are received during the next 2.5 hours, after which about 1 flow per minute is still observed for several hours until the end of the trace.

We have devised simple heuristics to identify flows that are unanswered due to the departure of a host. This is based on the observation that a host that leaves the P2P network will not initiate any new UDP or TCP flows; in contrast, hosts behind NATs are likely to initiate flows to other peers. We found that host departure is responsible for 41% and 48% of the unanswered UDP flows for Kad and KadU respectively, and the rest is due to hosts behind home NATs. For failing TCP connections, 75% were sent to hosts that appear to have left the P2P network. These results indicate that both factors (node departure and home NATs) play an important role in explaining the results.

Overall, these results indicate that better mechanisms must be designed to handle stale P2P membership, and hosts behind NATs for a P2P system to exhibit more friendly behavior to network operators. In particular, it is important for membership management algorithms to avoid propagating hosts behind NATs, and to ensure stale information is eliminated in a timely fashion.

2.7.3 Malicious P2P Servers

In this section we describe our findings when studying the metric *avg-conn-per-IP-TCP*. The interesting region for this metric in both Kad and KadU corresponds to samples where a peer contacts the same destination host more than once within a

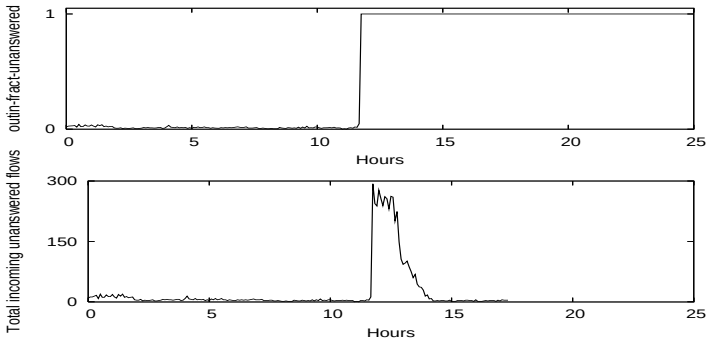


Fig. 2.9. Host leaves the group about 11 hours after the beginning of the trace. Fraction of unanswered flows on the top and total number of unanswered flows on the bottom.

sample time window. We found that 94% of the interesting samples for KadU dataset were generated by only two hosts. In the following, we focus our analysis on one of the hosts which we call $h1$, with the results being similar for the other host.

We found that $h1$ generated a large number of flows to two servers, namely $Server1$ and $Server2$. Figure 2.10 shows the number of connections $h1$ initiated to these servers during the whole trace. The X axis shows the connection start time, and the Y axis shows the connection ID. Positive IDs show connections opened to $Server1$, while negative IDs show connections opened to $Server2$. The average connection duration is 15 and 8 seconds respectively. For periods when the host was active, the inter-connection time to both servers is relatively small, i.e., 51 and 63 seconds for $Server1$ and $Server2$ respectively.

To further understand this behavior, we searched for information on the IP address of both servers and found that $Server1$ was reported as a *fake server* and $Server2$ was reported as a *full server* [54]. A fake server pretends to be a legitimate eMule server to fool clients with the goal of spying on them and to inject false information to disrupt the P2P system. These servers might be planted by parties such as the RIAA (Recording Industry Association of America) [55]. Fake servers may also impact the performance of victim peers since such peers cannot exploit the eMule network to search and exchange content. A full server is a legitimate server that has reached

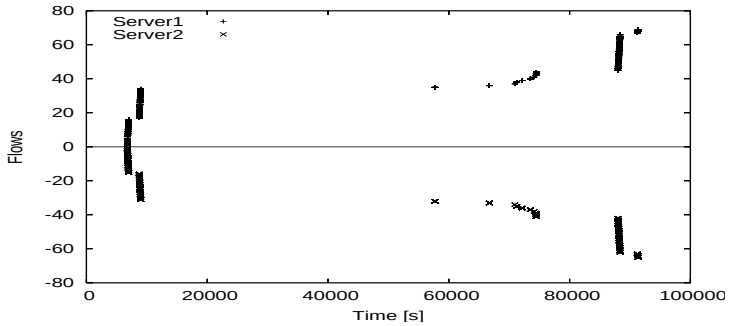


Fig. 2.10. Connections made by the KadU client $h1$ towards $Server1$ (fake server) and $Server2$ (full server).

the maximum number of clients it can serve, so that further requests are denied. We believe the list of servers that host $h1$ has is limited, and possibly contains $Server1$ and $Server2$ only. This would result in $h1$ persistently initiating connections to both servers.

Considering the Kad dataset, the methodology pointed out an analogous problem. 92% of the interesting samples in the *avg-conn-per-IP-TCP* metric were generated by a single host. Once again, we found the host had a large number of connections to a particular server. Interestingly, we could not confirm from available manually maintained lists whether this host was a fake or full server, and we believe this is a hitherto unknown fake server. In general, we believe a traffic analysis approach such as ours can help in automatically identifying or inferring servers/peers with suspicious behavior, rather than relying entirely on manually maintained lists.

2.7.4 Other Interesting Findings

In this section we present some other examples of the findings highlighted by our methodology:

• *Inter ISP traffic - KadU*: As mentioned in Section 2.6.1, the metric *inout-ISP-to-Internet-ratio-TCP* for KadU shows a cluster in the range 0 to 0.62, with the majority of samples in the range 0 to 0.03. This represent clients where a large fraction of the

connections was directed to peers in the Internet. In fact, 20% of the P2P traffic incoming to the MiniPoP is sent from the Internet. While some of the behavior is caused by clients that are searching for content not present in the KaU network, we believe there are several clients not using the KadU network to search. This is an undesirable behavior considering that the KadU developers optimized KadU to maintain P2P traffic within the ISP.

• *Abnormal behavior with "buddy" maintenance mechanisms - KadU*: The metrics *outin-avg-conn-per-IP-TCP* and *outin-total-conn-attempts-UDP* highlighted an atypical region for which a host was receiving a lot of TCP and UDP flows in the KadU dataset. By investigating the anomalous samples for these metrics, we have found a single host which was responsible for 57% and 33% of interesting samples respectively. We looked further and found that a single external peer opened 825 TCP connections and $1,678$ UDP connections to this host in a 25 hours period. Looking at the message type exchanged among these two peers, we discovered that messages were related to the eMule "buddy" mechanism. To allow a client C behind a (home) NAT to upload content, C finds a "public" peer (or buddy) who forwards requests from other clients to it. Then, C can directly initiate a connection to the client requesting the content. Normally, clients behind a NAT use a single TCP connection to the buddy. The large number of connections initiated by this particular client is therefore atypical, and points to incompatibilities between the Kad/KadU protocol and the (home) NAT/Firewall, which repeatedly closes the connections.

• *Isolating very active peers - Kad and KadU*: Our methodology pointed out potentially interesting peers which account for a large number of interesting samples in several metrics. We isolated the hosts responsible for more than 10% of the interesting samples for at least 5 metrics, finding 3 KadU peers and 6 Kad peers. For example, a client was generating many interesting samples for the metrics *live-conn-TCP*, *inout-total-conn-attempts-UDP* and *bps-rcvd-TCP*, which show the host was aggressively searching and downloading content. Similar results were observed for other clients. While we did not find evidence of malicious activity, we believe our

methodology was able to isolate very aggressive behavior, which is important from the ISP point of view, and also for the end users, e.g., to avoid leacher behavior.

2.8 Generalizing to other Systems

While much of our analysis is conducted with Kad, and KadU given their predominant usage in the ISP network, we are extending the analysis to consider other popular P2P systems. In this section, we present preliminary results reporting our findings with the BitTorrent [3] and DC++ [36] systems. Given these systems are not as widely used in the network, our analysis is conducted on a separate one-week long trace so sufficient data samples may be obtained.

• *Idle TCP connections in BitTorrent*: Our methodology showed that clients have a high *fract-unanswered-appl*, i.e., a large fraction of successful TCP connections to which the contacted peer never replied. We found that more than 40% of the samples are in the range 0.6 to 1. These connections are typically short lived, with more than 90% of them lasting less than 30 seconds. We believe this occurs when a client contacts a peer that is no longer sharing the file being searched. This is another example of how stale information leads to wasted network resources.

• *Unnecessary P2P traffic in BitTorrent*: Similar to Kad and KadU, we found that stale information and NAT presence could account for a large fraction of unanswered UDP flows in BitTorrent. 32% of all samples and 10% of all UDP flows were sent to hosts that left the P2P network. In addition, for BitTorrent we noticed that from the 4.2 million UDP flows initiated in the PoP, more than half are unanswered which can also be due to stale membership and NATs.

• *DDoS attack exploiting DC++*: We noticed that the interesting region for the metric *inout-1024-dest-ports-TCP* ranges from 2 to 9 ports contacted in a time slot. Further investigation showed that many of these connections were targeted to port 80 and did not receive a response from the destination. Manual inspection showed that the contacted IPs were real web servers and not DC++ clients. We hypothesize these

flows are part of a DDoS attack exploiting DC++. Attacks of this nature have been previously reported [56]. In addition, we found that 95% of the DC++ connections stay within the ISP. Of the connections that leave the ISP, 21% are destined to ports below 1024 and potentially contribute to DDoS attacks, as described above.

2.9 Related Work

Many recent works have focused on P2P traffic classification. In general, two main approaches have emerged: packet inspection techniques [45, 57] and behavioral classification techniques [39–44]. Our work aims at analyzing the subset of traffic which has been already classified as P2P to identify any undesirable behavior these systems might have.

Anomaly detection of network traffic in general (for example, [33–35, 58]), has been widely studied. Many of these works have developed automated techniques for detecting anomalies. The techniques typically leverage the fact that most data-points are normal, and flag anomalies based on sudden and significant deviations from baseline values. Our work differs in several ways. First, our focus is on obtaining better understanding on the types of undesirable behavior that P2P systems in the wild exhibit. Undesirable behavior can be predominant, and anomaly detection techniques based solely on deviations from baseline behavior are not sufficient in our context. This led us to rely on domain knowledge as part of our analysis. Second, our notion of undesirable behavior is broad, and includes not only malicious activities, but also many other patterns of undesirable behavior peculiar to P2P systems, for e.g., wasted resources caused by NATs and stale information in the system (Section 2.7.2). Third, we have considered a much wider range of traffic features than typical anomaly detection work, given limited a priori knowledge of the types of undesirable behavior that P2P systems exhibit. That said, it would be interesting to develop more automated analysis techniques for the identification of undesirable behavior in P2P systems in the future.

Our work both corroborates known patterns of undesirable behavior in P2P systems, and provides more insights into them. In particular, our findings on DDoS confirm recent works where researchers showed the feasibility of exploiting P2P systems to launch DDoS attacks on the Internet [23–26]. While these works proposed attack heuristics and showed the feasibility of attacks, our work is one of the first to show evidence of real attacks taking place in the wild. Our findings on fake servers similarly support [55]. Our results (Section 2.7.3) have not only shown peers impacted by well-known fake servers [59, 60], but also shown the potential to automatically detect hitherto unknown fake servers.

2.10 Conclusions and Discussion

As a primary contribution of this paper, we have shown that P2P systems in the wild exhibit many types of undesirable behavior, and we have provided insights into the prevalence, characteristics and impact of such behavior. We have also shown the potential of a systematic approach involving P2P traffic analysis in uncovering such behavior. Our results include instances where the performance of the P2P system itself may be impacted (e.g. due to maliciously deployed servers), as well as examples where P2P system behavior can be detrimental to the network (e.g. DDoS attacks exploiting P2P systems, or unwanted traffic due to hosts behind NATs and stale group membership). While there has been some prior awareness of these issues in the community, to our knowledge, our is the first work that systematically studies P2P traffic patterns with a view to identifying the undesirable behavior they exhibit.

Our analysis suggests that undesirable behavior may be exhibited by a range of P2P systems. Further, most examples of undesirable behavior that we found point to intrinsic design limitations in the underlying systems themselves, which leads us to believe that our findings are likely to hold if the systems are analyzed in other networks as well. That said, generalizing the findings across multiple networks, and a wider range of P2P systems is an important aspect of our ongoing work.

In this paper, we have adopted a semi-automated methodology that combines data-mining with extensive use of domain knowledge in interpreting the results. This has been necessitated given that there is limited understanding in the community today on the characteristics of undesirable behavior that P2P systems may exhibit, and since the intrinsic heterogeneity of P2P traffic makes it hard to clearly distinguish undesirable behavior from normal usage. Undesirable behavior can be predominant, complicating the use of automated techniques which identify anomalous behavior by detecting significant deviations from normal values. An interesting avenue for future research is exploring more automated analysis techniques, for instance based on identification of significant shifts in P2P system behavior across networks and across time, and by employing rules general across an entire class of P2P systems.

3. PREVENTING DDOS ATTACKS ON INTERNET SERVERS EXPLOITING P2P SYSTEMS

3.1 Introduction

Peer-to-peer(P2P) systems have matured to the point we have recently seen several commercial offerings [61–63]. Given the increasing prevalence of the technology, it becomes critical to consider how such systems can be deployed in a safe, secure and robust manner.

Several works [64–68] have studied how malicious nodes in a P2P system may disrupt the normal functioning, and performance of the system itself. In this paper, however, we focus on attacks where malicious nodes in a P2P system may impact the **external** Internet environment, by causing large-scale distributed denial of service (DDoS) attacks on web servers and other sites not even part of the overlay system. In particular, an attacker could subvert membership management mechanisms, and force a large fraction of nodes in the system to believe in the existence of, and communicate with a potentially arbitrary node in the Internet. The attacks are hard to detect and track-down as the packets being exchanged between the attacker and innocent nodes are not distinguishable from normal protocol packets.

While the community has been aware of the possibility of exploiting P2P systems to launch DDoS attacks for several years (for example [23]), a number of researchers have highlighted the criticality of the problem in recent years. The feasibility of exploiting the intrinsic characteristics of P2P systems for indirection attacks was first systematically shown in [25]. Since then, several works including our own [24, 26, 69–71] have demonstrated the generality of the problem, by showing that a variety of extensively deployed systems may be exploited to launch DDoS attacks. The systems include unstructured file-sharing systems such as Gnutella [24], and BitTorrent [26,

69], DHT-based file-sharing systems such as Overnet [25], and Kad [70, 71], and a video broadcasting system based on End System Multicast (ESM) [70]. Attacks can be significant - for example, [70] showed an attack where over 700 Mbps of UDP traffic was generated at the victim by exploiting Kad using 200 attacker machines. Creating the attack heuristics was as simple as modifying two source files and less than 200 lines.

While traditional ways to launch DDoS attacks such as DNS reflector attacks [72], or botnet-based DDoS attacks [73] are more widespread today, there is evidence to suggest that exploiting P2P systems to launch DDoS attacks in the wild is on the rise [53,56]. For instance, Prolexic Technologies has reported that they have observed what they term a DC++ attack [56] that involved over 300K IP addresses. Discussions in the eMule forum [52, 53] have indicated DDoS attacks on DNS servers exploiting the DHT-based Kad system. We present evidence of this in Section 3.2.3 through traffic measurements collected at the edge of a MiniPoP of an ISP. We believe it is imperative to systematically study the problem given the large-scale deployments of P2P systems, the emerging reports of attacks in the wild, and the relative lack of attention to the area.

In this paper, we seek to obtain a deeper understanding into the threats by studying the intrinsic design limitations of existing P2P systems which leave them vulnerable to such DDoS attacks. As a first contribution of this paper, we categorize all known DDoS attacks presented to date. In our classification, we focus on the underlying cause for achieving *amplification*. By amplification, we refer to the ratio of the number of messages (or bytes) received by the victim to the total number of messages (or bytes) sent by all malicious nodes. We focus on amplification since this is the key factor that determines whether an attack is attractive to a malicious node. We then articulate key design principles that P2P designers must follow to avoid these sources of amplification. The principles highlight the need to validate membership information before they are further used or propagated, and the need to protect against multiple references to the victim. While these principles are almost

obvious in retrospect, the failure to follow the guidelines in a wide range of deployed systems, and the resulting repercussions are striking.

As a second contribution of this paper, we systematically explore the effectiveness of an active probing approach to validating membership information in mitigating the threats. We focus on this approach since it does not rely on centralized authorities for membership verification, and since it is applicable to both structured (DHT-based) and unstructured approaches. The key issues with an active probing approach are ensuring that the probes used for validation themselves do not become a source of DDoS, and dealing with benign validation failures that may occur due to packet loss, churn in group membership, and the presence of hosts behind Network Address Translators (NATs). We present simple mechanisms to address these issues, and show that with the mechanisms, the maximum amplification achievable by attackers can be bounded. We have incorporated these mechanisms in two contrasting applications - a DHT-based file-sharing system (Kad [32]), and a video broadcasting system (ESM [74]) with stringent performance requirements. Through extensive experimental evaluation, we show that the schemes may be suitably parameterized to ensure that DDoS attacks are effectively limited, while not sacrificing application performance.

3.2 DDoS Attacks by Exploiting P2P Systems

Recently researchers have shown how a variety of P2P systems can be exploited to launch DDoS attacks on any Internet host such as a web server [24, 26, 69–71, 75]. Each of these works presents attack heuristics on a specific system, and to date there have been several different attacks reported on five widely deployed P2P systems. In this section, we begin by presenting an overview of these systems in Section 3.2.1, and then summarize the attacks exploiting them in Section 3.2.2.

3.2.1 Systems Background

Kad and **Overnet** are large-scale DHT-based file sharing systems with each having more than one million concurrent users. Kad is supported by the popular eMule [32] client and its clones, while Overnet is supported by eDonkey [76]. These two systems are similar because they both implement the Kademlia [48] protocol, and differ primarily in implementation issues. In both systems, each participating node maintains a routing table with a subset of peers. For any given file, there are multiple "*index nodes*", each of which maintains a list of members who own that file. Index nodes are regular participants, who have an ID close to a file ID. Every node periodically publishes to the index nodes what files it owns. When a node wants to download a file, it first employs an iterative query look-up mechanism to locate an index node, and it obtains a list of members having the file from the index node.

BitTorrent is a very popular tracker based unstructured P2P system for file sharing. In BitTorrent, if a node wants to download a file (say a movie), it must first download the torrent file for that movie, which is published through out-of-band channels such as websites. The torrent file contains a list of trackers. A tracker is a central server which maintains the membership information of a swarm. Each node contacts one or more trackers to obtain a list of peers in the swarm, and starts exchanging data with the peers. Each node also contacts the trackers periodically afterwards to discover more peers.

Gnutella is another popular unstructured P2P file sharing system which has a two-tier hierarchy. In Gnutella, when a node launches a query for a file, other nodes may reply with a query hit message that includes the IP and port of a peer that has the file. In addition, when the node requests the peer for the file, the format of the file request message is HTTP based.

ESM is one of the first operationally deployed P2P video streaming systems. It constructs a multicast tree on top of an unstructured overlay for data delivery, and employs a gossip-based membership management protocol. Each node periodically

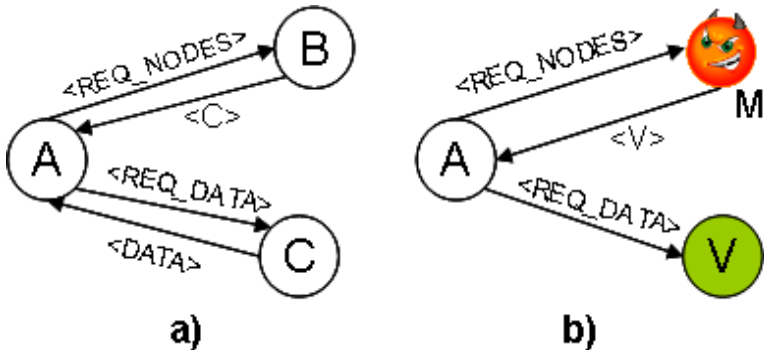


Fig. 3.1. a) Normal Operation. b) Attack.

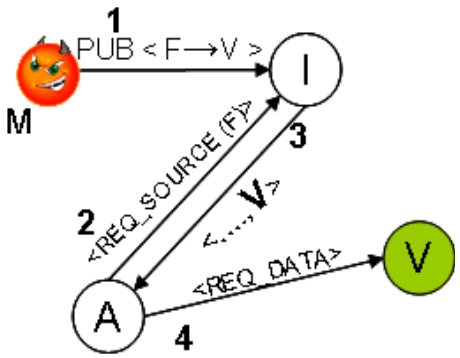


Fig. 3.2. Index poisoning attack in Overnet

picks another node at random, and sends it a subset of the peers it knows. A node adds to its routing table any peer that it has not already known, and may use these peers for various protocol operations such as parent selection.

3.2.2 Attacks

In any scalable P2P system, a node A may learn about another node C from a peer node B, as shown in Figure 3.1.a). For example, this is required when a node locates index nodes, or obtains a list of sources for a file. However, this operation can be exploited by a malicious node M to redirect A to the victim V, as shown in Figure 3.1.b). V can potentially be any Internet host such as a web server. In this

section we summarize various ways in which this vulnerability has been exploited to cause large scale DDoS attacks.

Attacks exploiting Kad and Overnet: While [70] and [25] have presented attacks on Kad and Overnet respectively, we believe that most of these attacks are applicable to both systems. Thus we summarize them together here.

Index poisoning [25]: This attack is depicted in Figure 3.2. Here, a malicious node M publishes to an index node I that the victim holds some file. Later any innocent participant A looking for a set of sources for the file will contact I and be redirected to the victim V. The redirected participants will try to establish TCP connections to the victim in order to download the file. This could not only result in TCP SYNs, but also result in successful TCP connections if for instance an actual web or mail server were running on the victim. The bar for such an attack could be raised by not requiring nodes to insert their IP and port information in application layer messages to begin with, however we note that index poisoning attacks could still occur if malicious nodes could conduct packet level spoofing.

NAT-buddy exploit: This attack may be viewed as a variant of the *Index poisoning* attack. It exploits a NAT traversal mechanism that is commonly used in today's P2P systems, including both Kad and Overnet. In such a mechanism, a node behind a NAT could select a public node as its "buddy". When the NAT node publishes a file, information about the buddy is included in its publish message to the index nodes. A malicious node exploits this to launch a DDoS attack, by advertising the victim as its buddy. When innocent participants obtain a set of sources from the index node, they contact the buddy (victim) as per normal protocol operations resulting in a DDoS attack on the victim. This attack is briefly discussed in [25]. The attack could potentially be prevented by modifying the underlying protocols so that NAT nodes are required to publish content through their buddies, however the protocol modifications must ensure the overheads at the buddy are not increased significantly, and must include mechanisms to ensure malicious nodes cannot deliberately increase

the overheads on the buddy, for instance by providing information about non-existent files.

Search hijack [70]: This attack exploits the parallel lookup mechanism of Kad and Overnet, where multiple nodes may be included in a reply to a query. In particular, when a malicious node receives a search query from an innocent participant, it includes in the reply multiple (fake) logical identifiers, all sharing the IP address of the victim. This results in the innocent participant querying the victim multiple times. Note that in order to enable distinct users behind the same NAT to participate in the system, Kad, Overnet, and many other P2P systems allow a participating node to communicate with multiple logical identifiers even though they share the same IP address.

Routing table poisoning [25]: This attack is specific to Overnet. It exploits the announcement messages, which enable nodes to announce themselves to others. In particular, a malicious node may put the victim's IP address in an announcement message and send it to an innocent participant, by exploiting a vulnerability specific to Overnet. This results in the innocent participant adding the victim to its routing table, and using it for normal protocol operations. Like with index poisoning attacks, the bar for such an attack could be raised by not requiring nodes to insert their IP and port information in application layer messages, however the attacks could still occur if malicious nodes could conduct packet level spoofing.

Attacks exploiting BitTorrent: [69] presents an attack where malicious nodes in a BitTorrent swarm can report to the tracker that the victim is a participating peer, through packet spoofing. This would cause the tracker to redirect innocent participants to the victim, who in turn repeatedly initiate TCP connections to the victim. [75] presents a variant of this attack where the tracker itself is malicious and falsely tells innocent participants in the swarm that the victim is a participating peer. [26] presents an attack where the attacker publishes fake torrent files to web sites of known torrent search engines. Each of the fake torrent files includes the victim's IP several times, each with a different port. Innocent participants who download the

torrent files believe that there are different trackers running on the same IP, and repeatedly try to connect to each of the trackers.

Attacks exploiting Gnutella: [24] presents an attack where malicious nodes include the victim's IP address in their replies to file query messages sent by innocent participants. The victim in this attack is a web server which hosts some files. Further, the file names in the reply messages are constructed in such a way that the file requests sent by innocent participants to the victim will look exactly like genuine HTTP requests from normal web clients. This causes the victim to upload an entire file to the redirected nodes.

Attacks exploiting ESM: This attack is presented in [70], where a malicious node M exploits the push-based nature of the gossip protocol in ESM. In particular, a malicious node generates false information about the victim being part of the group, and aggressively pushes the information as part of its gossip messages to innocent participants. In addition, the malicious node includes the victim's IP several times in a gossip message, each with a different logical ID, similar to the *Search hijack* attack in Kad.

3.2.3 Potential Evidence for Real Attacks in the Wild

We have analyzed a one day trace collected at the edge of a MiniPoP of an ISP, where we found potential evidence of abnormal traffic to DNS servers from peers running the Kad system. Our trace consists of flow-level logs [1]. Our analysis considers all Kad UDP traffic between hosts inside the network and hosts in the Internet. Figure 3.3.a) shows the number of unanswered flows (i.e., flows for which only outbound traffic was seen), as a function of the destination port number. Impulses were plotted only for ports that received more than 500 unanswered flows. The spike at port 4672 is expected since this is the default UDP Kad port. However, it is interesting that

[1] A flow is identified by the 5-tuple comprising source and destination addresses and ports, and protocol. A UDP flow starts when the first packet is observed, and is considered to end if no packet is seen in any direction for 200 seconds. If a packet is never observed in the reverse path, then the UDP flow is said to be unanswered.

there is a spike at port 53 (DNS port). Figure 3.3.b) shows the fraction of unanswered flows out of the total outgoing flows, as a function of the port number. The graph shows that port 53 has the highest ratio of unanswered flows with 92%.

We considered whether these results could be due to actual Kad clients running on port 53, which are aiming to hide their presence in firewalled networks. We believe there are two reasons this is unlikely to be the case. First, a majority of flows (92 %) to port 53 are unanswered, while for other ports, the percentage of unanswered flows was at most 60%. Second, manual investigation (e.g., by doing an nslookup on the target IP addresses), indicated that most destinations were DNS servers (e.g., in China and Thailand)

We believe these results are abnormal, and potentially point to a DDoS attack exploiting the Kad system. Our results also corroborate discussions in the eMule forum [53], and works by other researchers [52], which further strengthens our belief that these results may be due to a DDoS attack. Finally, we note that others have observed DDoS attacks in the wild exploiting other P2P systems [56].

3.3 Eliminating Attack Amplification

While specific solutions may potentially be designed for each of the attacks listed in Section 3.2, the fact that a multitude of attacks have been reported against a range of systems leads us to explore more general principles and techniques to protect against the entire class of attacks. In this section, we dissect the attacks in Section 3.2.2, and identify a few underlying patterns which lead to large attack amplification. Based on the insights, we articulate a few generic design principles for P2P developers.

3.3.1 Classifying Attacks by Amplification Causes

Based on the source of amplification, the attacks described in Section 3.2.2 can be classified as follows (also in Table 3.1):

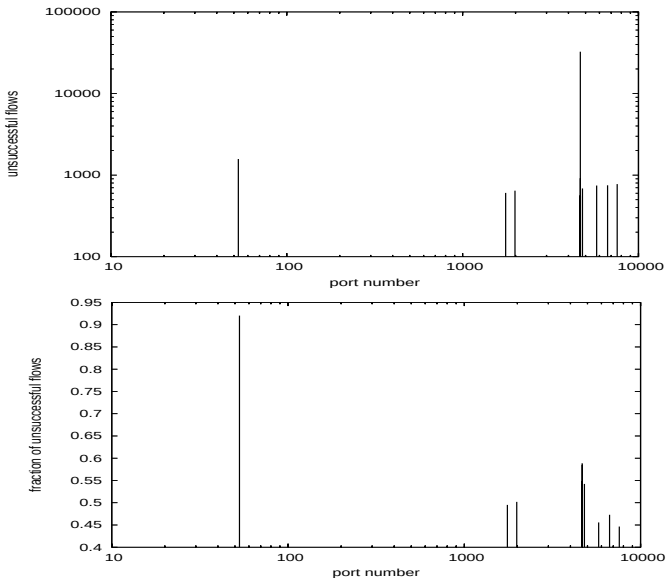


Fig. 3.3. a) Number of unsuccessful Kad flows sent to ports that received more than 500 unsuccessful flows. b) Fraction of unsuccessful Kad flows to ports that received more than 500 unsuccessful flows.

Table 3.1
Classification of DDoS Attacks by their Major Source of Amplification

System	*Attack*	Repeated packets to victim	Multifake	Delegation	Triggering large reply from victim
Kad & Overnet	Index poisoning [25]			√	
	NAT-buddy exploit [25]			√	
	Search hijack [70]		√		
Overnet	RT poisoning [25]	√			
BitTorrent	Fake report to tracker [69]			√	
	Malicious tracker [75]	√			
	Fake torrent file [26]		√	√	
Gnutella	Gnutella attack [24]				√
ESM	ESM attack [70]	√	√	√	

Repeated packets to victim: This is the simplest source of amplification, where an innocent participant may keep using the victim it learned from a malicious node

for various protocol operations, despite the fact that it has never been able to communicate with the victim. Surprisingly, many of the systems have this vulnerability.

Multifake: In this case, a malicious node may convey the same victim multiple times to an innocent participant, by disguising the victim as multiple different members of the system. This is a major source of amplification in the case of the *Search hijack* attack on Kad and Overnet, the attack on ESM, and the attack on BitTorrent involving fake torrent files [26]. This heuristic can be generalized to mount an attack on a *network*, by having the malicious node including fake membership information about several different IP addresses, all of them belonging to the same network.

Delegation: For attacks in this class, amplification is achieved because the innocent participants spread fake membership information. This is a major source of amplification in the *Index poisoning* and *Nat-buddy exploit* attacks in Kad and Overnet where the index nodes propagate the victim, in the attacks on BitTorrent, where the trackers and torrent websites propagate the victim, and in the attacks on ESM, where innocent participants gossip about the victim to each other.

Triggering large reply from victim: This class includes attacks such as those on Gnutella [24], where an entire file is sent in response to a query, leading to high amplification.

3.3.2 Principles for Robust Design

We next present several key design principles which if followed could eliminate all the amplification causes.

• *Validate before use*: Each node must validate membership information it learns, before adding the information to its routing table and/or using it for protocol operations. This eliminates the *Repeated packets to victim* vulnerability since it ensures that no protocol packets will be sent to a node until it has been validated. Further, it also eliminates the *Triggering large reply from victim* vulnerability because an in-

nocent node would not send file requests to the victim server since the server cannot be successfully validated.

• *Validate before propagation*: Any membership information must be first validated before being propagated to any other nodes. This ensures that innocent participants do not propagate fake membership information. A potential source of attack amplification such as *Delegation* is avoided as attackers are now constrained to infecting the innocent participants directly. It is worth pointing out that this principle must be followed for all operations involving exchanging membership information. For instance, in Kad, the principle is followed when nodes learn other members through search process, but not followed when index nodes receive publish messages from other nodes, thus Kad is still vulnerable to the *Index poisoning attack*.

• *Preventing multiple references to the victim*: This principle guards against amplification due to *Multifake*, where an attacker conveys the same victim multiple times to an innocent participant by disguising the victim as multiple distinct members of the system. A particularly interesting case is where the attacker is able to convey the victim multiple times in a single membership message, as in the *Search hijack* attack, because this has interesting implications for our defense mechanisms as we will see in Section 3.4.

3.4 Enhancing DDoS Resilience

As discussed in Section 3.3, the key principles involved in limiting DDoS amplification is validating membership information. Several approaches may be adopted to this end, and we discuss this further in Section 3.4.1. In this paper, we explore in depth the potential of an active probing approach to validating membership information. We discuss the considerations that motivated us to focus on such an approach, and details of the approach in the rest of the section.

3.4.1 Approaches for Validating Membership Information

We discuss possible approaches that may be adopted for validating membership information:

• *Use of centralized authorities:* Centralized authorities can simplify validation of membership information by providing signed certificates that indicate that a member belongs to a group. The certificates may be distributed through the membership management mechanisms, and enable a participant to verify the membership information corresponds to a genuine member. However, the existence of central authorities cannot be assumed in many P2P deployments, and we only consider mechanisms that do not rely on their existence.

• *DHT-specific approaches:* It may be feasible to leverage the properties of DHTs, and design solutions tailored to them. For instance, one approach is to assign each node an ID dependent on its IP address, for instance the hash of its IP address [77]. An attacker that attempts to provide fake membership information in response to a search query must then ensure that (i) the victim ID obeys the necessary relationship to the victim IP; and (ii) the victim ID is close to the target ID of the search. These dual constraints on the victim ID may be difficult to simultaneously satisfy, complicating the attack. Issues that need to be addressed with such an approach include the need to accommodate multiple participants behind the same NAT which share a common IP address, and the fact the victim ID is only loosely constrained by the target ID. While the approach has promise, we do not explore it further since our focus in this paper is on mechanisms that apply to both structured and unstructured approaches.

• *Corroboration from multiple sources:* Another approach to validating membership information is based on prior works on Byzantine-tolerant diffusion algorithms [78–81]. In this approach, a node will accept and communicate with a newly-learned peer only if it learns about the peer from multiple other nodes (say k). Such schemes are susceptible to attacks where the attacker has control over k or more nodes, because then he can make these malicious nodes lie about the same fake membership informa-

tion, and defeat the corroboration. Such attacks may be particularly easy to conduct in conjunction with a Sybil attack. Another concern with the approach is that from a performance perspective, the larger the k value, the longer it may take to receive responses from all k nodes, which can slow down convergence and performance [82].

3.4.2 Validating Peers through Active Probing

In this paper, we explore the potential of an active probing approach for membership validation. We focus on such an approach because (i) it does not rely on centralized authorities; and (ii) the technique applies to both unstructured approaches as well as structured DHT-based approaches. Thus, the technique has potential to be widely applicable to a range of existing P2P deployments.

In the approach, when a node receives a *membership message*, it probes any member included that it did not know before, and does not send further messages to it or propagate it unless it receives a response. We use the term *membership message* very broadly to refer to any protocol message that contains information about other participants in the group. This includes, for example, search replies and file publish messages in Kad, Overnet and Gnutella and gossip messages in ESM. It also includes messages where a node may directly announce itself to other nodes.

It is possible that a probe sent to the victim could trigger a spurious response from some other application running on the port under attack. To prevent this, the probe request and response should contain: (i) a predefined byte pattern unique to the application and distinct patterns for the request and the response and (ii) a sequence number in the request which will be incremented in the response.

While probing-based validation has potential, the key issues are ensuring the probes themselves do not become a source of DDoS, and dealing with benign validation failures that may occur due to packet loss, membership churn, and the presence of hosts behind Network Address Translators (NATs). We discuss heuristics to handle these issues in the rest of the section.

3.4.3 Preventing DDoS Exploiting Validation Probes

To prevent validation packets from being a source of DDoS attacks, two broad approaches may be adopted. A first approach involves coordination across members in the system, so as to ensure only a subset of members conduct the validation. Such coordination mechanisms are themselves subject to attack when malicious nodes are involved, given that members may not be truthful in sharing information about validation failures. While reputation mechanisms such as [83] may be used to address these concerns, these mechanisms are often themselves vulnerable to attacks such as whitewashing (see [84] for a survey of schemes and attacks). Further, many schemes (for example [83]) rely on centralized authorities or pretrusted nodes, and we are interested in solutions not depending on their existence. Thus we focus on mechanisms that only rely on locally observable events. More specifically we employ two schemes as described below:

Source-Throttling: This scheme seeks to limit the attack traffic that a single membership message from a malicious node can induce. This prevents attacks like the *Search hijack* in Section 3.2.2, which enabled an attacker to achieve significant attack amplification. In the scheme, when a node receives a membership message, rather than try to validate all the members included in the message at the same time, validations are performed to at most m members initially, where m is a parameter. A new validation to an additional member is conducted only when a previous validation is successful. This mechanism ensures that a single message from an attacker can trigger at most m validation messages to the victim under attack. Combined with the rest of the validation framework, this limits the total attack amplification achievable by the attacker to m, as we discuss in Section 3.5.1.

While small values of m are desirable to keep attack amplification small, the main concern is the potential impact on performance. In particular, validation failures may occur for benign reasons resulting in unnecessary throttling. Consequently genuine information in a received membership message may be ignored. Further, the validation

process may incur some delay, possibly resulting in higher latencies or convergence times for the application. In Section 3.7, we evaluate the feasibility of employing small m values for real applications.

Destination-Throttling: The *source-throttling* scheme by itself could prove highly effective in thwarting the attacker in many circumstances since it bounds the amplification the attacker can achieve. We augment this scheme with a simple mechanism we term *destination-throttling*, that can limit the number of packets each innocent participant sends to the victim. We note that even with *destination-throttling* the victim can receive $O(N)$ attack packets, where N is the total number of participants in the system. However, since the amplification is bounded, it would require the combined set of all attacker machines to send $O(N)$ messages to innocent nodes to redirect them to the victim.

An obvious solution to limiting packets that each innocent node sends to the victim is to blacklist sources that cause repeated validation failures, and ignore further membership messages from them. This heuristic by itself does not suffice however, since there are potentially many sources, and further the malicious node may mount a Sybil attack. Instead, with *destination-throttling,* repeated validation failures to a destination are used as an indication that it is under attack, and future validations are not sent in such a case.

A destination could refer to an <IP,port>, an IP, or an entire network. The network to which an IP belongs may be determined by a longest prefix match on a database of prefixes and netmask information extracted from BGP routing table snapshots [85, 86]. Such a database could be obtained by a client in an out-of-band fashion at the start of the session, and the information is unlikely to change over the duration of a typical session. Alternately, though less accurately, an IP could be assumed to belong to a /24 network. In the rest of the paper, we use the terms "prefix" and "network" interchangeably.

Every client maintains the total number of validation packets, and the number of failed validations to each <IP,port>, IP, and prefix. Each validation failure is asso-

ciated with a time-stamp indicating when the failure occurred, and only failures that occurred in a recent time window T are considered. A validation packet is suppressed if either the <IP, port>, IP address, or prefix is suspected under attack. An <IP, port>is suspected under attack if more than F_{ipport} failures have been observed to it. A prefix (IP) is suspected under attack if it has seen at least F_{prefix} (F_{ip}) failures that involve D_{prefix} (D_{ip}) distinct IPs (<IP, port> pairs). The set of parameters must be chosen so that the likelihood of destinations being falsely suspected under attack due to benign validation failures is small. We discuss this further in Section 3.7.

With the scheme as above the total validation failures to a prefix before it is suspected under attack could be as low as F_{prefix}, and as high as $F_{ipport} * D_{ip} * D_{prefix}$. To better contain the magnitude of a potential attack, once F_{prefix} failures have been seen to the prefix, validations are permitted only if there has been no prior failure to the IP. With this modification, at most $F_{prefix} + D_{prefix}$ validation failures are allowed to the prefix. A similar heuristic is applied to failures to individual IPs.

The *destination-throttling* scheme could potentially be exploited by malicious nodes to create attacks on the performance of the P2P system itself. We analyze the potential for such attacks in Section 3.5.2. A variant of the *destination-throttling* scheme that could be used to raise the bar against such attacks is to consider a prefix under attack if the percentage of failed validations to a prefix exceeds a threshold. While we believe such a variant could be easily integrated with our solution, we focus on a solution based on the total validation failures to each prefix to ensure the total number of packets sent by each innocent node to a victim prefix can be bounded under DDoS attacks.

3.4.4 Avoiding Validation Failures with NATs

Many P2P systems employ *NAT-Agnostic* membership management operations. In these systems, when member B propagates information about member X to member A, there is no indication as to whether X is behind a NAT or not. This can result

in benign failures with probing-based validation. In particular, if X were behind a NAT, a validation packet sent by A to X will not successfully reach X unless X had previously contacted A.

While alleviating communication issues with NATs is an ongoing area of work [87, 88], these techniques are not always effective with symmetric NATs [47], which is the most restrictive type of NATs, and accounts for close to 30% of NATs [87]. With symmetric NATs, different connections initiated by the same internal node generate different external IP and port mappings. Incoming packets are only allowed from the public nodes to which packets have been sent. Two nodes both behind symmetric NATs cannot communicate with each other.

To handle NATs (including symmetric NATs) and firewalls, we require that membership management operations are *NAT-Aware*. In particular, membership information about nodes behind NAT are propagated with a flag indicating they are behind NAT. When a node A learns about X, it probes X only if it is not behind a NAT, thereby avoiding benign validation failures. We discuss potential attacks on the scheme where a malicious member may falsify information regarding whether another member is behind a NAT in Section 3.5.

3.4.5 Illustrating the Scheme

Probing-based validation mechanisms may be easily integrated with various P2P systems, to protect the systems from the exploits described in Section 3.2.2. As a concrete example, Figure 3.4 illustrates the steps to prevent attacks exploiting buddy mechanisms in Kad and Overnet with our scheme: (1) Node N is behind a NAT, and sends an advertisement to node I indicating that a public node B is its buddy; (2) I validates B, and accepts information only on successful validation; (3) C obtains information that N has the file and B is the buddy; (4) C validates B, and then sends it a request; (5) this is relayed to N which then sends C the file. Note that steps 2

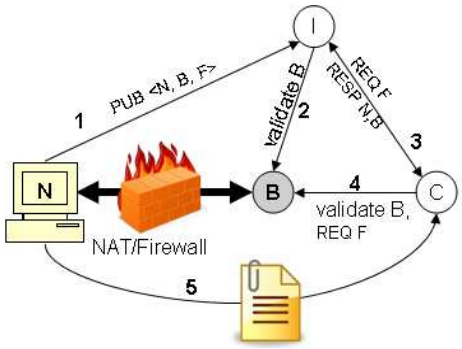


Fig. 3.4. Complete sequence of steps for normal buddy operations with probing-based validation mechanisms.

and 4 are the validation messages that are triggered with our framework. The other exploits described in Section 3.2.2 are similarly handled, and we omit the details.

3.5 Analysis

In this section, we analyze the effectiveness of probing-based validation mechanisms in thwarting DDoS attacks. We also analyze the vulnerability of the mechanisms to new attacks that may impact application performance, and suggest refinements to minimize the impact.

3.5.1 DDoS Attacks

We first consider DDoS attacks on hosts not participating in the P2P system, which is the primary focus of our paper, and what we view as a more critical threat. We then present possible refinements for handling attacks on participating nodes.

DDoS Attacks on Hosts not in the P2P System

We discuss the key measures of interest:

• *Message amplification*: this is the ratio of the number of messages received by the victim to the total number of messages sent by all malicious nodes. We observe that with source-throttling, the *message amplification* is at most m, independent of the number of malicious nodes. To see this, consider that a DDoS attack on a victim not in the P2P system consists entirely of validation messages. Each validation message must be triggered by a membership message directly received from a malicious node. This is because innocent members only propagate membership information they can validate, and hence do not propagate the victim. Thus the *message amplification* achievable is bounded by the maximum number of validations that may be sent to the victim due to a single membership message from an attacker node. This is bounded by m, by the throttling heuristics.

• *Bandwidth amplification*: this is the ratio of attack traffic received by the victim to the total traffic sent by all attacker nodes. More, precisely, the *bandwidth amplification* can be expressed as: $m * \frac{validation\_message\_size}{membership\_message\_size}$. Given that a validation message is small in general and smaller or at most comparable to the size of the membership message, the *bandwidth amplification* is also bounded by m.

• *Attack magnitudes*: The destination-based throttling scheme ensures that the total number of packets sent by each participating node to a victim is bounded. In particular, at most $F_{prefix} + D_{prefix}$ packets are sent to any victim prefix over a period of time T, where T is the time for which a failed validation is considered. Further, since this scheme is entirely based on the destination to which validation failures are observed, and does not depend on the source which triggered the validation, the bound holds irrespective of the number of attackers or under Sybil attacks. We note that the victim can still receive $O(N)$ attack packets, where N is the total number of participants in the system. However, we believe this is not a significant concern because the amplification is bounded by m, and the attacker must send $O(N)$ messages to innocent nodes to redirect them to the victim. While it may be possible to design mechanisms that can ensure not all innocent participants send attack packets to the victim, such mechanisms are likely to involve coordination across the nodes. Such

coordination mechanisms are not only complex, but also are themselves subject to possible attack when malicious nodes are involved. We made a deliberate decision to avoid such mechanisms given the feasibility of bounding attack amplification without them.

Man-in-the-Middle-Attacks: The above analysis assumes an attacker model where malicious nodes can only join the P2P system as regular participants. A more sophisticated attacker could potentially conduct a man-in-the-middle attack, perhaps by compromising routers. In particular, an attacker could intercept and respond to validation packets sent by innocent participants to the victim, tricking these participants into thinking the validation is successful. This is a potential concern because the tricked participants, which we term *delegates*, could propagate information about the victim, and consequently the amplification bounds above do not hold.

We note that to be effective, that attacker must strategically intercept validation packets from a moderate number of delegates, and this may not be trivial. If validations are intercepted from too many delegates (for instance, the man-in-the-middle is located close to the victim), the attacker is likely to see all validations to the victim; if the validations are intercepted from too few delegates, the total traffic induced at the victim due to the delegates is small.

The message amplification under man-in-the-middle attacks may be expressed as $\frac{m*M_{redirn}+M_{del}}{M_{redirn}+M_{MIM}}$. Here M_{redirn} is the number of redirection messages sent by the malicious nodes, and M_{MIM} is the number of validations to the victim that are intercepted by the attacker to conduct the man-in-the-middle attack. M_{del} is the number of validation messages received at the victim induced by membership information propagated by the delegates. Let us assume that the maximum number of innocent participants to which each delegate could spread information about the victim is K. This bound could be achieved by simply having a delegate conduct periodic revalidation, and limiting the rate at which it spreads information about innocent participants, or by having the delegate conduct a revalidation each time it spreads membership information to to K other participants. Then, M_{del} is bounded by $K * M_{MIM}$, and the overall

message amplification is $\frac{m*M_{redirn}+K*M_{MIM}}{M_{redirn}+M_{MIM}}$. It is easily verified that this quantity is between m and K. Finally, the amplification is likely to be even smaller because the analysis does not consider that the man-in-the-middle will not only see traffic due to validations from the delegates, but also normal protocol traffic packets from the delegates.

DDoS Attacks on Participating Nodes

We next discuss the case when a participating node is the victim of the attack. A straightforward way to extend the probing-based validation mechanisms is to require the victim to explicitly deny validation requests if it receives them at an excessive rate. On receipt of such a denial, an innocent participant considers the validation to have failed. While this can help limit the attack, this is not enough because innocent participants that were previously able to successfully validate the victim, (which we again refer to as *delegates*), could still continue to propagate the membership messages.

The validation traffic seen at the victim due to membership information spread by the delegates depends on (i) the number of delegates; and (ii) the rate at which each delegate spreads the victim to other innocent participants. The second factor is under the control of the delegates and is easily controlled. However, it becomes important to ensure the total number of delegates does not grow in unbounded fashion.

A possible heuristic to limit the number of delegates is to have each node A restrict the number of other nodes that may successfully validate it over any window of time, and deny all other requests. In addition, as described above, delegates are required to periodically revalidate A, and if a revalidation fails, they are required to stop sending further packets to A, and stop propagating A to others.

An attacker could exploit this heuristic by sending a large number of validation messages to the victim, thereby causing the victim to deny validations from innocent peers. We term such an attack a *Disconnection Attack*, and note that the attack

targets the performance of the P2P system itself. It is unclear how attractive such an attack is since it would require the malicious node to send $I * U$ messages each revalidation period, where I is the delegate limit per node, and U is the number of nodes to be disconnected. While it is perhaps not hard to disconnect a single node, disconnecting a significant fraction of nodes in the system involves a large number of messages given Kad and Gnutella have over a million users. For instance, if $I = 10000$, $U = 1000$ (representing a disconnection of 0.1% of all participants), and assuming a revalidation is conducted every minute, a malicious node would need to send about 160,000 messages per second. On the other hand, this revalidation traffic would not be a significant burden for innocent nodes - considering that typically each node has a few hundred neighbors, the revalidation traffic required of a typical node would be about 2 messages per second. Finally, we note that if the attacker only controlled a small number of IP addresses or prefixes, such disconnection attacks could be further prevented by blacklisting peers that repeatedly send validation messages.

In structured DHT-based overlays, a concern with limiting delegates is that a node may deny a validation request from a peer with an ID close to its own, thereby impacting the DHT structure. While this may not be a concern ordinarily if the delegate limit is chosen conservatively, it may be possible for an attacker to create such a situation by redirecting a lot of innocent nodes (whose IDs are far from the victim's ID) to the victim. To defend against this type of attack, the scheme may be modified for structured overlays to have nodes probabilistically accept a peer as a delegate, based on the ID distance between the two. The closer the peer, the higher the probability. The scheme is effective given that a node will not have too many peers with an ID close to it, and an attacker has no control over the ID of an innocent node. We defer a more detailed investigation of these issues to future work.

3.5.2 Attacks on Destination-throttling

In this section, we discuss possible attacks on the destination-throttling scheme. We note these attacks don't apply to the source-throttling scheme, which was the primary mechanism in limiting attack amplification. We identify two variants of the attack:

• *Attacks on destination-throttling:* A malicious node M may flood another node A with several fake membership entries corresponding to a victim prefix. The resulting validation failures could force A to throttle the prefix, resulting in A being disconnected from valid participants in that prefix.

A similar attack could be performed to cause A to be disconnected from a single node rather than a prefix. A malicious node M may do so by flooding A with incorrect port information about the victim node. However, this attack is less attractive for an attacker than the previous one, since it only disconnects A from a single node. While our analysis below focuses on attacks to prefixes, we believe that similar arguments will hold for attacks to a single node.

• *Attacks on NAT-Aware mechanisms:* When a malicious node M propagates membership information to node A, it may falsely indicate that a participating node N has a public IP address, even though N is behind a NAT. This could induce validation failures from A to N, potentially resulting in A throttling the prefix to which N belongs.

We observe that a malicious node must send at least $d = \frac{F_{prefix}+D_{prefix}}{m}$ messages to disconnect a node from participants in one prefix. This is true since at least $F_{prefix} + D_{prefix}$ validation failures are required to a prefix before future validations to it are suppressed, and at most m validation failures may be induced by a single membership message due to the source throttling mechanisms. Based on our parameterization results in Section 3.7, we expect d to be of the order of 10 messages.

While such attacks are theoretically feasible, we believe they are not attractive for an attacker in practice, since a large number of messages must be sent to cause

a noticeable degradation in application performance. We note the messages involved to disconnect a node depends on the number of prefixes spanned by participating nodes, which is of the order of tens of thousands, and at least an order of magnitude larger than the attacks in Section 3.5.1. Further, in a system like Kad, to limit the ability of n participants to access a file, an attacker must disconnect the participants from all prefixes with nodes that have the file. Further, popular files are likely to be distributed across many prefixes. Disconnecting n participants from p prefixes requires at least $d * n * p$ messages. This can be high, considering that the system can have millions of clients distributed over tens of thousands of prefixes. In addition, note that any such disconnection is temporary, since validation failures are timed out after time T, and sustaining the disconnection requires continued messages from the attacker.

It is possible to limit the extent of disconnection attacks if the malicious nodes are localized to a small number of prefixes. In particular, once a prefix Z is suspected of being under a DDoS attack, source IP prefixes that have triggered a validation failure to Z are black-listed. Further validations to Z are sent only if they are triggered by source IP prefixes that have not been blacklisted. We note however that the fact malicious nodes are localized to a small number of prefixes may not be known apriori. Thus, it is desirable to dynamically tune the scheme to ensure the extent of disconnection attacks is limited if the number of prefixes spanned by malicious nodes is small, however ensure the scheme is not susceptible to DDoS attacks if the number of attacker prefixes is large. To handle this, once prefix Z is suspected of being under a DDoS attack, a validation triggered by a source prefix that is not black-listed is sent only if the source prefix belongs to a randomly selected fraction f of all possible prefixes. f is initialized to 1, and dynamically reduced, for example by a factor of 2 on each validation failure after the number of black-listed source prefixes exceeds a threshold. Intuitively, we expect the scheme to stabilize at f values which are inversely proportional to P, the number prefixes spanned by attackers. Further, the number of additional validation messages to the victim is at most $\log P$. We can

bound this quantity even more, by not permitting any further validations once f goes below a threshold.

Finally, the *destination-throttling* scheme as presented in the paper considers a prefix under attack if the number of validation failures to a prefix exceeds a threshold. A potential approach to raise the bar against disconnection attacks is to consider a prefix under attack if the percentage of validation failures to the prefix exceeds a threshold. We have focused on the former variant in this paper to bound the total number of packets sent by each innocent node to a victim prefix under DDoS attacks. However, the latter variant could be easily integrated if the need to prevent disconnection attacks is viewed more critical.

3.6 Evaluation Methodology

The primary question driving our evaluations is how effective the validation framework is in preventing DDoS attacks without sacrificing application performance. To understand this, we have integrated our framework into a file sharing application (Kad) and a video broadcasting application (ESM), two mature and contrasting P2P applications, with very different membership management designs and performance requirements. In the rest of the section, we present our evaluation goals, metrics and our experimental methodology.

3.6.1 Evaluation Goals

We have the following goals:

• *Performance under normal conditions:* We study the impact that the validation framework has on application performance under normal conditions when there are no attacks. This enables us to determine the performance of the throttling heuristics under benign validation failures, arising due to packet loss, churn, and NATs.

• *Impact of throttling parameters:* As our analysis in Section 3.5 indicates, the attack amplification, and attack magnitudes achievable with the validation framework are

directly dependent on the choice of the m parameter for the source-throttling scheme and the various F and D parameters for the destination-throttling scheme. On the one hand, it is desirable to keep these parameters small to limit DDoS attacks. On the other hand, the smaller the values, the greater the potential impact on performance. Thus our evaluations explore how various parameters of the throttling mechanisms impact application performance, and seek to identify operating ranges where the impact is small.

• *Impact on contrasting applications:* The performance with the schemes is dependent on the application itself. Hence, we explore the issues in the context of two very different applications, DHT-based structured file-sharing Kad and unstructured video broadcasting ESM.

• *Performance under attacks:* Finally, we evaluate the benefits of the validation framework both in limiting DDoS attacks, and in preventing degradation in application performance in the presence of malicious nodes.

3.6.2 Performance Metrics

In our evaluations with file distribution applications (Kad), we consider the fraction of successful searches, and the time a successful search takes to locate an index node. For video broadcast (ESM), we consider the fraction of the streaming video rate received by participating nodes and the join time of nodes to the multicast tree. In addition, in evaluating the *destination-throttling* scheme, we measure the number of prefixes (as well as IPs, and <IP,port>s) blocked by the scheme, in the absence of attackers.

3.6.3 Methodology

Our experimental methodology employs experiments both on the live Kad network, and on Planetlab.

• *Live Kad experiments:* The performance of both throttling schemes is sensitive to the extent to which benign validation failures are seen in realistic application deployment settings. This in turn depends on realistic churn rates, packet loss rates, and the fraction of participating NAT hosts. In addition, the performance of the destination-throttling scheme is sensitive to the number of participating nodes that share an IP prefix, and the number of participating nodes that share an IP address. To evaluate the performance of the schemes under such realistic conditions, several of our experiments are conducted on the live Kad network. We do this by implementing our validation framework in a Kad client, and having it join the live Kad network. We compare the performance of an unmodified Kad client on the live network, with multiple instances of the modified Kad client running different parameter settings.

• *Planetlab experiments:* One limitation of evaluations on live Kad is that the throttling schemes are implemented only on the clients we have control over. To evaluate the throttling schemes in settings where all participating clients implement the validation framework, we evaluate Kad on Planetlab. In addition, Planetlab evaluations enable us to compare the performance of the schemes in the absence of attacks, and under attack scenarios.

Since there are no long-running live ESM broadcasts, experiments on live ESM deployments are not feasible, and our ESM experiments are conducted on Planetlab. To ensure realism, our experiments with ESM leverage traces from real broadcast events [74] to model the group dynamics patterns, and bandwidth-resource constraints of nodes. Since most nodes on Planetlab have public IPs, we emulate NAT connectivity restrictions by implementing packet filtering to ensure that two Planetlab incarnations that are behind a NAT cannot communicate with each other. Note that our emulations model *Symmetric NATs*, the properties of which are elaborated in Section 3.4.4.

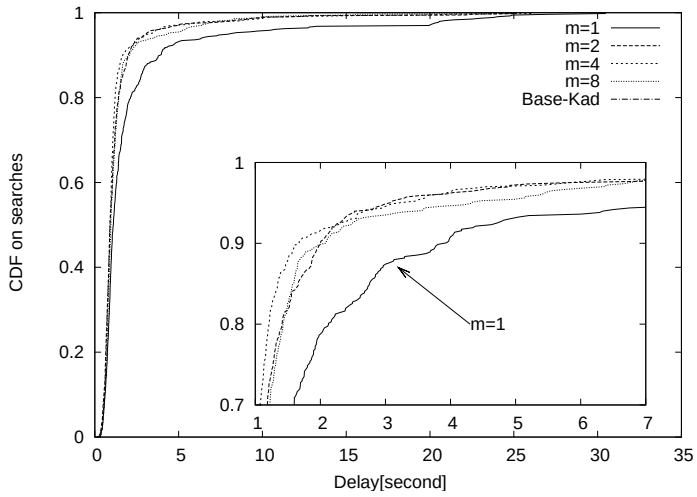


Fig. 3.5. Source-throttling: Impact of m on search delay. Measured in Kad.

3.7 Parameterizing the Validation Framework

In this section, we present results evaluating the impact of the validation framework and its various parameters on the performance of Kad and ESM. Our goal is to identify the possible sweet-spot - parameters that are small enough to minimize the DDoS attacks, yet large enough to tolerate most benign validation failures. We implemented the throttling schemes in real Kad and ESM clients. For comparison, in our experiments we also ran unmodified Kad and ESM clients, which we refer to as *Base-Kad* and *Base-ESM*. The experiments with Kad in this section were conducted in the live Kad network, while the experiments with ESM were conducted on Planetlab. All the experiments were conducted in the absence of attackers.

3.7.1 File Sharing Application

We parameterize the source-throttling scheme in Section 3.7.1 and the destination-throttling scheme in Section 3.7.1.

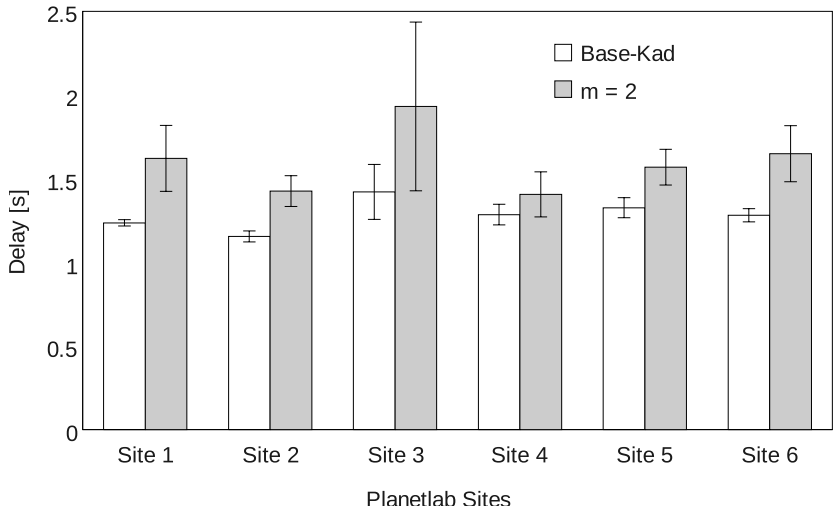


Fig. 3.6. Source-throttling: Sensitivity to Planetlab Site. For each site, each bar is the average over 5 runs of the 90th percentile of the delay of successful searches. The error bars show the standard deviation. Measured in Kad.

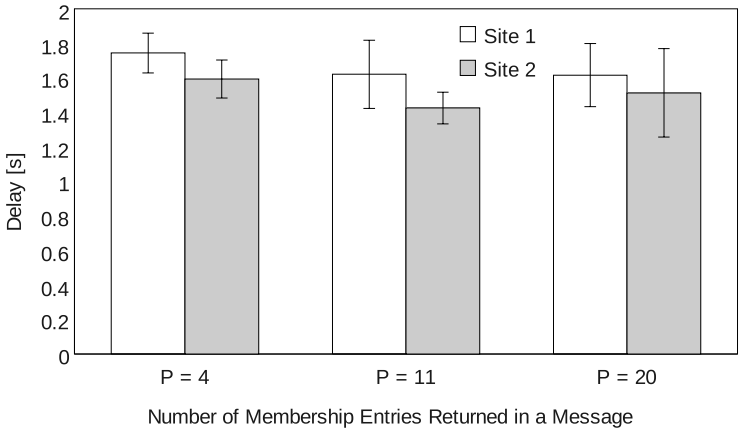


Fig. 3.7. Source-throttling: Sensitivity to the number of membership entries returned in a reply message. For each site, each bar is the average over 5 runs of the 90th percentile of the delay of successful searches. The error bars show the standard deviation. Measured in Kad.

Source-throttling: Impact of m

As described in Section 3.2.1, when a Kad node conducts a search for a keyword or a file, it issues queries, and receives replies containing membership entries. This

enables the node to locate index nodes for the keyword or the file. The source-throttling scheme impacts the search performance because entries returned in a reply message may not be fully utilized, and the validation process may incur some extra delay, as we have explained in Section 3.4.3. A factor that may affect the results is the number of membership entries returned in a reply message. In Kad, this number is a client-specified parameter which can be set by the node, and is included in the query messages sent to peers so they will know how many entries to include in the replies. In the mainstream implementations such as eMule and aMule client software, the default values for this number are 2, 4 and 11, depending on the type of the search. While we set it at 11 for our experiments, we also study the sensitivity of our results to this parameter.

We first measure the impact of the m parameter on application performance. We let four versions of our modified Kad client (i.e. with source-throttling) run in parallel, with each version running a different value of m (i.e. one version with $m = 1$, one version with $m = 2$ and so on). In addition, we let an unmodified client (i.e. Base-Kad) run at the same time. A random set of 1000 keywords from the English dictionary were picked, and each client conducted one search for each of these keywords in sequence in a one hour period. For all the keywords for which the Base-Kad client returned at least one index node, we measured the time each client took to locate the first index node.

In all, the Base-Kad client returned at least one index node for 567 searches. The fraction of these searches for which the source-throttling scheme also returned one or more index node was 94.5% for $m = 1$, 98.8% for $m = 2$, 99.5% for $m = 4$, and 99.6% for $m = 8$. Thus, while the throttling scheme did impact some of the searches the effect was minor for $m = 2$ and higher values.

Figure 3.5 plots the CDF of the search delay for successful searches (i.e. the time taken for the first index node to be returned). The line on the top represents the Base-Kad scheme, and the line on the bottom represents the source-throttling scheme with m set at 1. The remaining lines are the source-throttling scheme with m set to

2, 4, and 8. These lines are very close to each other and practically indistinguishable. Overall, the results indicate that while extremely aggressive levels of throttling ($m = 1$) can result in noticeable degradation in application performance, the degradation is minimal for even slightly higher values of m, such as $m = 2$ or $m = 4$.

Sensitivity: The previous graph showed results for one experiment from one site. We now consider six distinct Planetlab sites and conduct five runs from each site. For all sites, we make two hosts join the Kad network, one host running Base-Kad and the other running source-throttling with $m = 2$. Figure 3.6 shows the search delay for each Planetlab site. We show two bars per site, one for Base-Kad and the other for source-throttling with $m = 2$. For each run, the $90th$ percentile of the delay across successful searches is taken. Each bar is the average of the $90th$ percentile of the search delay over five runs. The error bars show the standard deviation. We observe that the results across sites are consistent with the results in Figure 3.5.

Additionally, we study the sensitivity of the results to the number of membership entries that are returned in a reply (P) in Figure 3.7. Here we only consider source-throttling with $m = 2$. Each group of bars correspond to a different value of P, with each bar corresponding to a different site. For each run, the $90th$ percentile of the delay across successful searches is taken. Each bar is the average of the $90th$ percentile of search delay over five runs. The error bars show the standard deviation. We notice that the performance of $m = 2$ is not sensitive to the change in the P setting. This further confirms the feasibility of using small m values in realistic settings. We repeated these experiments on additional Planetlab sites and results were similar.

Destination-throttling: Impact of D and F

We next study the impact of the destination-throttling scheme on Kad performance. In doing so, the key metric is the extent to which destinations (prefixes, IPs, and <IP, Port>s) are unnecessarily blocked due to benign validation failures.

Table 3.2
Destination-throttling: Percentage of contacted prefixes blocked with the particular search rate and for various combinations of (D_{prefix}, F_{prefix}). Measured in Kad.

# of searches	(D_{prefix}, F_{prefix})				
	(5,5)	(5,10)	(10,10)	(10,15)	(15,15)
100	0.07%	0%	0%	0%	0%
300	0.55%	0%	0%	0%	0%
1000	4.53%	0.44%	0.29%	0.07%	0.05%

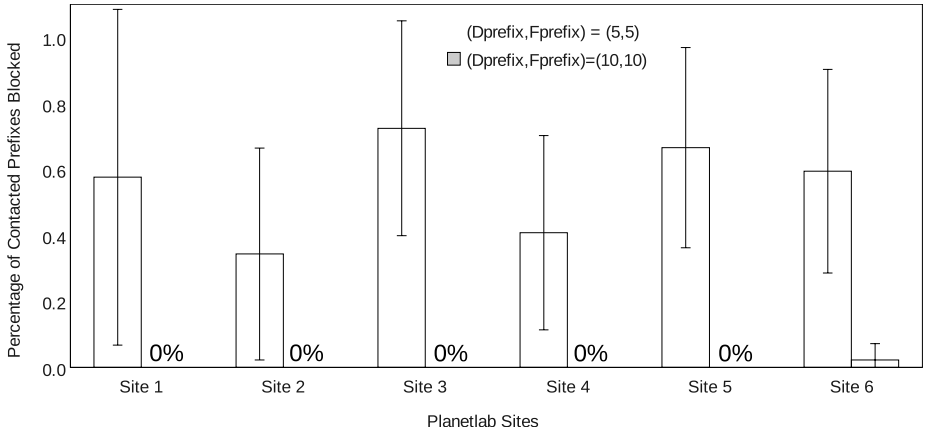


Fig. 3.8. Destination-throttling: Sensitivity to Planetlab Site. For each site, each bar is the average over 5 runs of the percentage of contacted prefixes that are blocked. The error bars show the standard deviation. Note that for all but one site, when (D_{prefix},F_{prefix}) = (10,10), 0% of prefixes are blocked. Measured in Kad.

Note that in the destination-throttling scheme, only the validation failures in a recent window of time are considered in deciding whether a destination is to be blocked or not. Hence our evaluation focuses on understanding validation failures seen by typical clients over such a time window. We believe that a reasonable window length should be tens of minutes, and use one hour in our evaluation[2].

[2]If the number of peers in the system is N, each peer could incur P validation failures before blocking a prefix, each validation packet is B bytes, and the validation failures are considered for a time T, then the expected validation traffic to a victim under a DDoS attack is $N*P*B/T$. For a population

A key factor that impacts the results is the aggressiveness with which clients conduct searches, as this impacts the number of validations conducted and failed validations seen. Thus we have done a sensitivity study to the rate at which our clients conduct searches. We choose the rate to be 100, 300 and 1000 per hour. According to a study of client query patterns [89], even 100 per hour is higher than the search rate of a typical client. 300 per hour is comparable to the most aggressive clients, while 1000 per hour is significantly beyond even most aggressive clients and stresses our scheme. Adopting a similar methodology as in Section 3.7.1, we instrument multiple versions of a client to join the live Kad network in parallel, each conducting an appropriate number of keyword searches over a one hour period.

Table 3.2 shows the percentage of prefixes blocked for a client conducting a particular number of searches in one hour. Results are shown for various combinations of D_{prefix} and F_{prefix}. A prefix is blocked if it has seen more than F_{prefix} failures involving D_{prefix} distinct IP addresses. To determine the prefix to which a client belongs, we use the Route Views dataset that helps map IP addresses to prefixes based on BGP data [85]. If any prefix is coarser than a /16 however, we simply consider the /16 as the prefix. The results show that barring extremely small values of the D_{prefix} and F_{prefix} parameters, the number of blocked prefixes is small. In particular, for 300 searches per hour, no prefixes are blocked if D_{prefix} is 5, and F_{prefix} is 10 or larger. Even with a search rate as high as 1000 per hour, the percentage of falsely blocked prefixes is less than 0.3% for a D_{prefix} of 10 or larger, and a F_{prefix} of 10 or larger. Overall, choosing F_{prefix} values in the 10-15 range, and D_{prefix} in the 5-10 range is effective.

We have conducted a similar evaluation to study the impact of the F_{ip}, D_{ip}, and F_{ipport} parameters. Our results show that with D_{ip} value of 3 or larger, and F_{ip} and F_{ipport} values of 5 or larger the scheme works well. With these settings, no destinations are blocked at the IP level even when the client conducts 1000 searches

of 1 million, with $P = 20$, $B = 100$ bytes, and $T = 1$ hour, this is about 4.4Mbps, which we believe is reasonable.

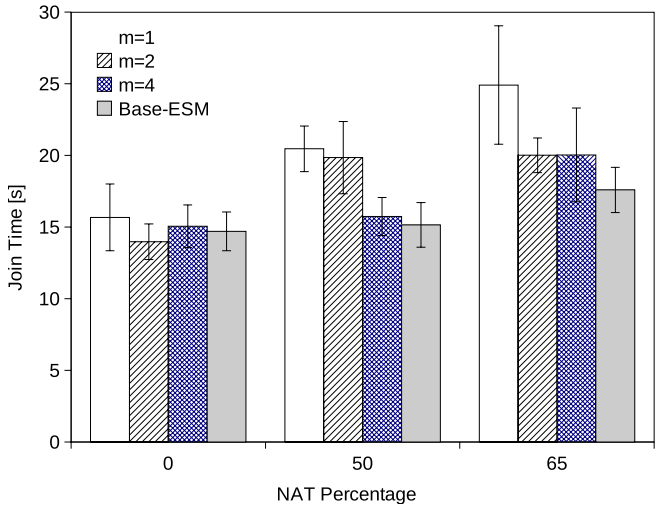


Fig. 3.9. Source-throttling: Impact of m on the join time of nodes. Each bar is the average over 5 runs of the 90th percentile of the join time. The error bars show the standard deviation. Measured in ESM.

per hour. Further, as the client search rate varies from $100 - 1000$ per hour, only $0.2\% - 0.4\%$ of the <IP, Port>s contacted are blocked. Interestingly, most of the blocked destinations corresponded to port 53, which is used for DNS service. We believe this corresponds to a real attempt of DDoS attacks exploiting the wild Kad system, as indicated by [52], and discussions on the eMule forum [53].

Sensitivity: We conducted sensitivity experiments of the results in Table 3.2 for 300 searches per hour and two (D_{prefix}, F_{prefix}) combinations of parameters, on six Planetlab sites. Figure 3.8 presents our results. There are two bars per site, each for a different combination of (D_{prefix}, F_{prefix}). Each bar shows the average over five runs of the percentage of contacted prefixes that were blocked. Error bars show the standard deviation. We can see that the results are consistent across sites with the results in Table 3.2. In particular, notice that for (D_{prefix}, F_{prefix}) = (10,10), five out of the six sites did not block any prefixes for the experiments conducted.

3.7.2 Video Broadcasting Application

Our ESM parameterization experiments were conducted on Planetlab. We leverage a trace from a real deployment [74] to emulate group dynamics and resource constraints in the system. We only show results for the source-throttling scheme. We were unable to parameterize the destination-throttling scheme since it requires realistic distributions of participants sharing a prefix or an IP address and this information was not available in the trace. Given this, we use the results from the Kad experiments to select the F and D parameters for ESM.

The Planetlab experiments emulate a 20 minute segment of the trace, with 148 joins, 173 leaves and 377 nodes in total. The streaming video rate employed is 450 Kbps, which represents typical media streaming rates in real settings [74]. In addition, we vary the fraction of NAT nodes in the system and turn off the NAT aware-heuristics, to increase the likelihood of benign failures and stress our scheme.

We observed that the average streaming rate received by nodes throughout the experiment is not affected by small values of m. More than 94% of the nodes received more than 90% of the streaming rate for $m = 1$, $m = 2$ and $m = 4$. To explain these results, consider that a key factor that may affect the performance of ESM is the number of entries the nodes have in their routing table. Having many entries ensures that nodes have enough potential parents to contact when they join the group or when a parent leaves. The source-throttling scheme can impact ESM by reducing the rate at which nodes learn about others, since membership information received may not be fully utilized. But this does not affect nodes in the long run, since over time they are still able to build a large routing table. Hence, the value of m does not affect the average streaming rate received by nodes.

To explore the potential impact of the source-throttling scheme on the performance of nodes in the initial phase, we consider the time it takes for a node to join the multicast tree. Figure 3.9 shows the 90 percentile of the join time for various values of m and different NAT percentages. There is one bar for each value of m. Each

bar is the average over 5 runs. Error bars show the standard deviation. The results show that while there is some increase in join time for small values of m, the impact is limited. For instance, the join time for $m = 2$ and $m = 4$ is only 2 seconds higher than Base-ESM, for settings with 65% NATs. Note that with NAT-aware heuristics in place, this difference would be even smaller.

3.7.3 Discussion

Our results indicate that the performance degradation with Kad and ESM is minimal, even with small values of the throttling parameters. Since the amplification and magnitudes of potential DDoS attacks are directly determined by these parameters, these results indicate the promise of the validation framework in effectively controlling DDoS attacks without impacting performance of these applications.

While the validation framework itself may be integrated easily in many P2P systems, the appropriate parameter choices are potentially dependent on the particular application. We now discuss the factors that might impact the parameter choice, and why we expect the parameters can potentially be kept small in general.

The primary concern with the source-throttling scheme is that when a membership message is received, benign validation failures incurred to some of the membership entries could prevent other potentially useful entries in that message from being validated (and hence utilized). To get more insight into this, consider that each membership message includes K entries, and assume the probability a probe will fail for benign reasons is p. With the source-throttling scheme, it may be easily verified that the expected number of membership entries that are probed is $\frac{m}{p}$, and the number of potentially useful entries probed is $\frac{m}{p} - m$. Since the expected number of potentially useful entries included in the entire membership message is $(1 - p)K$, the fraction of potentially useful entries that our source throttling scheme actually utilizes is $\frac{m}{pK}$. If the probability p of benign validation failures is kept low, for example by including NAT-aware heuristics, and by ensuring the system does a good job of minimizing

stale membership information, then, small m values will have only minor performance impact. Even if the application cannot eliminate benign validation failures, it may have other mechanisms that can help limit the performance impact. For example, when the Kad system receives a membership message in response to a search request, it preferentially probes entries that are closer to the search target. This ensures that the most potentially useful entries are utilized first, even if some of the entries are not utilized.

In our work, we have assumed the parameters are set uniformly across all clients and in static fashion. One could envision the need for the parameters to be set dependent on the client characteristics (e.g. aggressiveness of search patterns), or with differing levels of thresholds for different destination networks, based on their number of participants or bandwidth capabilities. In our evaluations, we have found that setting parameters conservatively based on worst-case scenarios (e.g. based on extremely aggressive search patterns) is sufficient to keep parameters low. That said, there may be potential benefits in other applications and deployment scenarios to tuning the parameters to individual clients or prefixes. Self-tuning mechanisms to dynamically determine appropriate parameters for any application, and deployment scenario are an interesting direction of future work.

3.8 Evaluation under Attack

In this section, we present results evaluating the effectiveness of our validation framework in minimizing DDoS attacks while achieving good performance even under attack. We implemented the validation framework on a Kad client and an ESM client. We refer to our modified clients as *Resilient-Kad* and *Resilient-ESM*. Again, we refer to the unmodified Kad and ESM clients as *Base-Kad* and *Base-ESM*. We set various throttling parameters according to the results obtained in Section 3.7. In particular, we set $m = 2$, $D_{prefix} = 10$, $F_{prefix} = 10$, $D_{ip} = 3$, $F_{ip} = 5$ and $F_{ipport} = 5$ for both

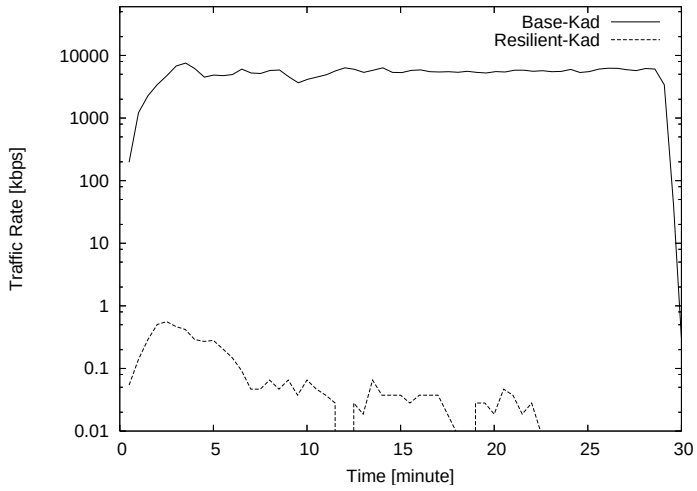


Fig. 3.10. Traffic seen at the victim as a function of time, with 5 attackers, and 50% of the nodes behind NAT. Measured in Kad.

Resilient-Kad and *Resilient-ESM*. All the experiments in this section were conducted on Planetlab.

3.8.1 File Sharing Application

In our Kad experiments, the inter-arrival patterns of nodes, and the stay time duration follow a Weibull distribution, based on [90]. A mean stay time of 10 minutes is assumed. Each experiment lasts 30 minutes, and involves 680 clients in total, with peak group size around 270. Each client conducts a search for a random logical identifier every 60 seconds, which is intended to simulate a search for a keyword or a file. There are 5 attackers that stay through the entire experiment and there is a single victim. Each attacker conducts the *Search hijack* attack from Section 3.2.2, and employs the *Attraction* [70] and *Multifake* (Section 3.3.1) heuristics. In particular, when an attacker initially joins the network, it proactively pushes information about itself to about 100 innocent nodes, forcing them to add the attacker to their routing tables. This causes many nodes to send search queries to the attacker and be redirected to the victim. In addition, in every search response, the attacker includes the

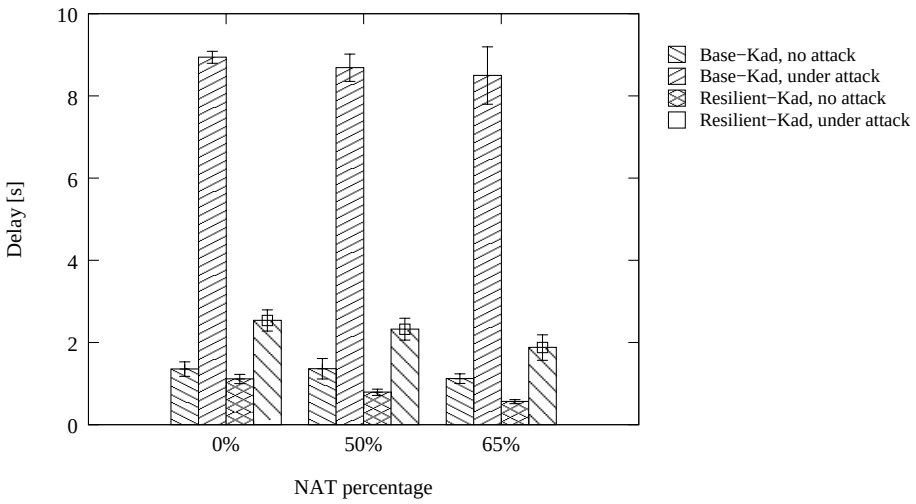


Fig. 3.11. Average time a search takes to locate the index node, with different NAT percentages. Each bar is the average over 5 runs. The error bars show the standard deviation. Measured in Kad.

victim's contact information about 100 times. This causes even larger attack magnitudes since innocent nodes send several messages to the victim for every response from the attacker (see [70] for more details).

Figure 3.10 shows the attack traffic generated at the victim as a function of time, from one experiment. With the *Base-Kad*, the traffic was as high as 10 Mbps throughout the run. Further, the traffic at each attacker was only about 250Kbps, which while higher than what a normal user sees, is 40 times lower than the traffic seen by the victim. However, with *Resilient-Kad*, the attack magnitude was effectively reduced by a factor of 100,000 from 10 Mbps to 0.1 Kbps.

Figure 3.11 compares the performance of the *Base-Kad* and the *Resilient-Kad* schemes. There are three sets of bars, each corresponding to a setting with a particular percentage of nodes behind NAT. Each set has four bars corresponding to the two schemes, with and without the presence of attackers. The graph shows the search delay, averaged over all searches conducted by all nodes throughout a run, then averaged over five runs. Error bars show the standard deviation. Here search delay is measured as the time taken to locate the node which is not behind NAT and which

has an ID that is the closest to the target ID. Our goal is to emulate location of index nodes in the real Kad network, and we note that only public nodes can be index nodes. We make the following observations.

First, in the presence of attackers, the search delay with the Base-Kad degraded significantly (over eight seconds in all NAT percentage settings). This was because nodes kept getting redirected by the attackers to the victim when they conducted searches. However, with Resilient-Kad, in the presence of the attackers the degradation was small and the search delay was still under three seconds for all NAT percentage settings. This was because fake information provided about the victim was quickly throttled and not used as part of the searches.

Second, in the absence of attackers, Resilient-Kad performed slightly better than Base-Kad. There are mainly two reasons for this. First, Base-Kad does not involve NAT-aware membership management. Thus, its routing table could include nodes behind NAT, and many search messages could fail since nodes behind NAT are contacted, leading to longer search times. In contrast, Resilient-Kad contains NAT-aware membership management leading to a routing table with fewer useless entries. Second, the destination-throttling scheme implemented in Resilient-Kad has the side effect that it can help purge stale membership information. In particular, consider a scenario where node A has left the group. Node B learns stale membership information about node A from some other node. In Base-Kad this membership information is accepted, adding to useless entries in B's table for some time. Further, the stale information may also be propagated by B to others. In contrast, with Resilient-Kad this membership information is not accepted due to the validation process, consequently helping avoid useless entries.

Finally, the performance of all schemes get better when the NAT percentage increases. This is because, as the number of public nodes decreases, there are fewer hops to reach the index node of a given target ID. Resilient-Kad nodes see better improvements since they only maintain entries of public nodes while Base-Kad nodes maintain useless entries of peers behind NAT.

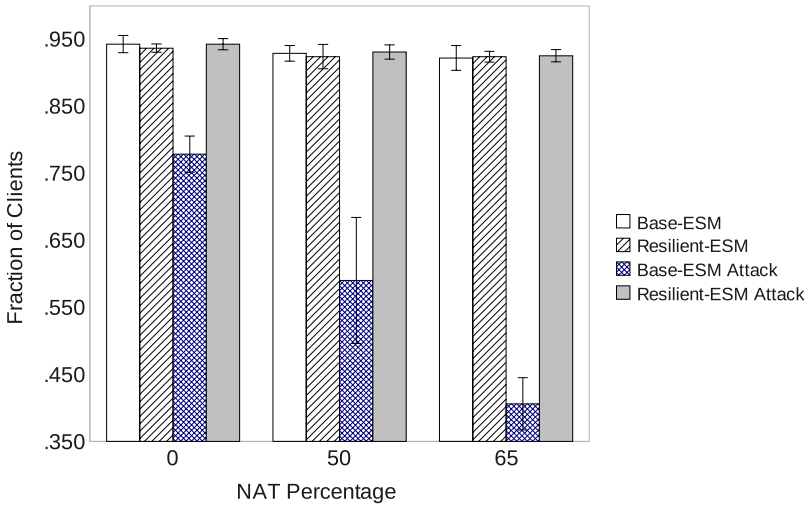


Fig. 3.12. Fraction of nodes that see more than 90% of the source rate, with 10% of the nodes as attackers and different NAT percentages. Each bar is the average over 5 runs. The error bars show the standard deviation. Measured in ESM.

3.8.2 Video Broadcasting Application

We consider the effectiveness of *Resilient-ESM*. We assume 10% of the nodes are malicious and perform an attack as described in Section 3.2.2 and in [70]. Our results indicate that the *Resilient-ESM* scheme is effective in containing attacks, reducing the attack magnitude from 10 Mbps to 0.1 Kbps.

Next, we consider application performance with *Base-ESM* and *Resilient-ESM*. Figure 3.12 shows the fraction of nodes that see more than 90% of the source rate, for both schemes, with and without attackers. Each bar is the average over five runs. Error bars show the standard deviation. We observe that *Base-ESM* shows significant degradation in performance in the presence of attackers. For instance, less than 70% of the nodes received more than 90% of the source rate, in settings with 65% NATs. This is because under attack, much of the membership information in the routing table of nodes is fake. This reduces the number of potential parents to contact when nodes join the group or when a parent leaves. Furthermore, we notice that the performance degradation on *Base-ESM* becomes more significant as the fraction of nodes behind

NAT increases. This is because in these regimes there are fewer potential parents in the system, so the impact of having fake entries in the routing table of nodes is even higher. In contrast, with *Resilient-ESM*, invalid membership information was not used, leading to fast convergence and high performance. In particular, 95% of the nodes received more than 90% of the source rate, for all NAT percentages, with and without attackers.

3.9 Interactions with P2P Developers

We have initiated discussions with P2P developers alerting them to the potential for DDoS attacks exploiting their systems, and encouraging them to make changes to their system to address the vulnerabilities. When we contacted the developers of eMule, they indicated they were already aware of our workshop paper [70], which had shown the feasibility of exploiting Kad (part of the eMule software) to cause DDoS attacks, and had implemented a number of changes to address the vulnerabilities. We identified some limitations of the changes, and the developers indicated they would implement additional mechanisms to address these in a future release. The combined set of changes limit the total number of entries in search response messages, and limit each response to have only one IP associated with an ID, and have at most 10 IPs for each /24 prefix. Similar restrictions are placed on the routing table entries. These changes are primarily intended to defend against attack heuristics such as *Multifake*, in order to bound attack amplification. These fixes are easily implementable and help solve some of the immediate problems. However, the fixes still suffer from a few limitations: (i) the amplification on /24 prefixes could be as high as 10; (ii) attacks on prefixes coarser than /24 are not prevented and the amplification of attacks on such coarser prefixes is not bounded; and (iii) each innocent participant could send an unbounded number of packets to the victim. We are in ongoing discussions with the developers to get our throttling mechanisms integrated, which could address these limitations. Our overall interactions show that P2P developers recognize the potential

for DDoS attacks exploiting their systems, and the value of designing systematic solutions to counter the threat.

3.10 Related Work

In Section 3.2.2, we have described most of previous research, which focus on exploiting individual P2P systems for DDoS attacks. In contrast, we have classified known DDoS attacks by amplification source, and identified principles to prevent amplification. In addition, ours is the first work aimed at enhancing the resilience of P2P systems to DDoS attacks.

The idea of employing probing-based mechanisms to validate membership information is briefly discussed in [25]. [24] discusses a solution specific to Gnutella which requires a node to complete a "handshake" with a peer prior to any file request message being sent to the peer, which could be viewed as a form of validation. In Kad, nodes employ a form of probing-based validation mechanism when nodes learn about members in search response messages (but nodes do not validate members learnt through publish messages). However, none of these consider the possibility that validation packets could be exploited to cause DDoS attacks. In fact, this was exploited in [70] to achieve large attack amplification. Further, we have also considered issues such as benign validation failures, DDoS attacks on entire network prefixes, and disconnection attacks. In addition, we have presented analysis and comprehensive performance evaluations of our framework. Finally, we have shown that our mechanisms are general and can defend against attacks on a diverse range of systems. [91] has used probing-based mechanisms to validate membership information to improve DHT lookup performance. In contrast, our focus is on DDoS detection and the interoperability of validation mechanisms with sources of benign failures such as NATs.

In our earlier paper [82], we showed the limitations of three other techniques in enhancing the resilience of P2P systems. A first technique was to limit DDoS attacks

by using pull-based membership management rather than push. However, while this reduces the susceptibility to attacks in some contexts, pull-based protocols are still vulnerable as our attacks on Kad show [70]. A second technique was to corroborate membership information from multiple sources. However, this technique is highly susceptible to Sybil attacks, and can incur significant performance degradation. A third technique is to limit the number of IDs associated with the same IP address, and the number of IPs sharing the same prefix that a node accepts. The limits are applied even if only genuine participants are involved, greatly limiting communication between participants. In contrast, this paper takes a different approach that requires nodes to be validated, and bounds the number of validation failures to each IP address or prefix, which in turn results in much fewer false positives. The focus on validations, throttling schemes, and analysis of their performance and security distinguishes our current work.

Researchers (e.g. [92]) have looked at exploiting unstructured file systems to launch DDoS attacks on P2P systems by introducing unnecessary queries, and having them flooded by the system. In contrast to these attacks, our focus is on DDoS attacks on external servers, caused by introducing fake membership management information.

Several works [64–68] focus on how malicious nodes in a peer-to-peer system may disrupt the normal functioning, and performance of the overlay itself. Many of these works, and most notably [66,67], focus on attacks on structured DHT-based overlays, and rely on trusted authorities to assign node IDs to principals in a certified manner. Our work differs in several ways. First, we focus on DDoS attacks on the external Internet environment, i.e. on nodes not participating in the overlay. Second, we focus on mechanisms that may be easily integrated into both structured DHT-based and unstructured non DHT-based systems. Third, we do not rely on a centralized authority to certify membership information. We believe these considerations are important to meet the threats for many existing extensively deployed systems. However, it may be interesting to investigate whether stronger security guarantees can be provided by exploiting DHT properties and using centralized authorities.

Several works have looked at the design of Byzantine resilient gossip protocols in the traditional distributed systems community in order to validate information (for example, [78, 81]). While there is much to learn from these efforts, we believe the scalability, heterogeneity and performance requirements with peer-to-peer networks and applications pose unique challenges and it is necessary to investigate the issues in the context of actual systems. Our focus in this paper is on exploiting P2P systems to launch DDoS attacks. In contrast, other works have explored attacks caused by DNS and web-server reflectors, and misuse of web-browsers and botnets [72, 93, 94]. A recent work [95] builds a DDoS attack model in the application layer, and proposes a defense mechanism against Layer-7 attacks by combining detection and currency technologies.

3.11 Conclusions

In this paper, we have made two contributions:

• First, we have shown that the feasibility of exploiting P2P systems to launch high-amplification DDoS attacks on web and Internet servers stems from a violation of three key principles essential for robust P2P design. These principles are: (i) membership information must be validated before use; (ii) innocent participants must only propagate validated information; and (iii) the system must protect against multiple references to the victim. While these principles are almost obvious in retrospect, the failure to follow the guidelines in a wide range of deployed systems, and the resulting repercussions are striking.

• Second, we have shown the effectiveness of an active probing approach to validating membership information in thwarting such DDoS attacks. We have focused on such an approach given that it does not rely on centralized authorities for membership verification, and is applicable to both structured and unstructured P2P systems. Despite the simplicity of the approach, it can keep attack amplification low (to a factor of 2), while having a modest impact on performance. For a video broadcast

application with stringent performance requirements, and for $m = 2$, the average source rate seen by nodes is practically unaffected, and when the 90%*ile* of client join time is considered, the increase is less than 12%. With Kad and for $m = 2$, the search time increases by less than 0.3 seconds on average.

While we have taken a key step towards enhancing the resilience of peer-to-peer systems to DDoS attacks, we are extending our work in several directions. From a security perspective, we are investigating mechanisms that can bound amplification when DDoS attacks are conducted on nodes actually participating in the system. From a performance stand-point, we are investigating self-tuning mechanisms to dynamically determine appropriate parameter choices for any application and deployment scenario.

4. UNCOVERING CLOSED COMMUNITY BASED P2P SYSTEMS

4.1 Introduction

The last decade has seen a rapid growth in popularity of peer-to-peer (P2P) systems, spanning diverse applications such as content distribution (e.g., BitTorrent, eMule, Gnutella), video streaming (e.g., PPLive,Coolstreaming), and audio conferencing (e.g., Skype). A vast majority of these systems are *Internet-scale,* and open to *any user* on the Internet. Indeed, the open nature of these systems is viewed as a key strength of P2P systems in enabling inexpensive and rapid deployment of services over the Internet.

In this paper, we raise the attention of the research community to the prevalence of *closed communities* of P2P users, and present an extensive characterization of such communities. Membership in such communities is restricted by imposing requirements on users that join the system. We focus on an important class of closed communities, where the primary criterion for admitting users is that they must be connected to the same network (e.g., same ISP). While several research efforts have extensively characterized the performance and traffic characteristics of open and Internet-scale P2P systems (henceforth referred to as *generic P2P systems*), e.g., [96–99], the study of closed and network-specific P2P systems (henceforth referred to as *P2P communities*) has received limited attention.

In this work, we characterize two communities, that we had the chance to monitor. The communities corresponded to two very contrasting networks. The first community has been created and used by customers in a large nation-wide ISP in Europe. The ISP offers customers Internet access, using both ADSL (1Mbps Uplink and 20Mbps downlink) and FTTH (10 Mbps uplink and downlink) technology.

The community observed in this network is based on the standard eMule P2P application [32], which has been modified by users to avoid problems caused by the assignment of private IP addresses to hosts inside the ISP network. We refer to this community as *ISP-Community* in this paper. The second community has been found inside a large university campus with hosts having high speed Ethernet connectivity. In this network, users modified the standard DirectConnect (DC) P2P application [100], so that only peers that run on hosts inside the campus can actually join the community. We refer to this community as *Campus-Community* in this paper.

Our main contributions are as follow:

• We show that P2P communities are extremely popular (e.g., generating more than 60% of total traffic for the ISP) and large-scale (e.g., comprising hundreds of thousands of users in the ISP network). The usage of the communities far exceeds usage of other more generic P2P systems - for e.g., in the campus network over 90% of the peers download over 90% of all P2P data using *Campus-Community*.

• We compare the performance of users of the P2P communities with users of more generic P2P systems. Our results show the performance benefits are largely determined by the access technologies of the users, and the degree of seed-like behavior shown by users behind high-speed access technologies. For instance, users of *Campus-Community* enjoy several orders of magnitude better performance than users of generic P2P systems in the campus network thanks to the high bandwidth provided by the campus LAN. In contrast, in the ISP network, the throughput of *ISP-Community* connections with senders behind ADSL links shows no particular improvement compared to generic P2P systems. However, the users of *ISP-Community* do see an improvement which may be attributed to a small fraction of senders behind FTTH links.

• We develop techniques to enable network providers understand how the growth of P2P communities impacts network capacity planning, and how projected changes in access technologies of users may affect these results. Our techniques center around

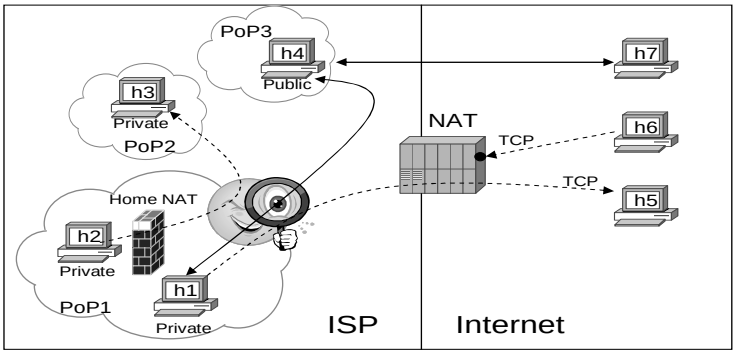


Fig. 4.1. ISP setup and trace collection.

a model we develop for the inter-PoP traffic of P2P communities. In contrast to prior work on traffic matrix estimation (for e.g., [101, 102]) which is agnostic to individual applications, our focus is on developing an application-specific traffic matrix model. Through simulations conducted using the model, we show that (i) while *ISP-Community* does reduce traffic on peering points as expected, more surprisingly, it results in a substantial increase in the traffic carried on internal network links (e.g., more than 60% of backbone links carry more traffic when *ISP-Community* is present); and (ii) this trend is exacerbated as more users move to high-bandwidth access technologies.

P2P communities must be distinguished from recent research proposals that have proposed mechanisms to ensure traffic of P2P systems is localized to ISP boundaries [13, 14]. Unlike these works, closed P2P communities have grown organically among users, and are already extensively deployed. Localization of traffic is not an explicit goal that spurred the growth of these communities, yet may occur as a prominent side-effect of the communities being closed to clients belonging to particular networks. That said, our results have important implications for research on localization of P2P traffic within ISP boundaries and indicate that benefits of localization should not be taken for granted. We discuss this in greater detail in Section 4.8.

4.2 Peer-to-Peer Communities

In this paper, we present an analysis of P2P communities in two different networks, (i) a nation-wide ISP in Europe; and (ii) a large-scale campus network in North America. We present more information about the networks and the associated P2P communities in this section.

4.2.1 P2P Community in an ISP Network

We describe a P2P community found in a nation-wide ISP in Europe. The ISP offers customers Internet access, using both ADSL (up to 1Mbps Uplink and 20Mbps downlink capacity) and FTTH (10 Mbps uplink and downlink capacity) technology. Hosts in each city are aggregated into Points-of-Presence (PoPs), which are then connected via the ISP backbone. Typically, hosts in the ISP are given a private IP address. As shown in Figure 4.1, plain connectivity is guaranteed to hosts inside the ISP network despite the use of private IP addresses. Whenever hosts with private addresses communicate with hosts in the external Internet, the data communication involves traversal of an ISP-wide NAT.

P2P systems typically have mechanisms in place to detect peers behind NAT and to limit their performance. This motivated a community of ISP users to modify eMule, a well known file sharing system, so that peers in the ISP could communicate with each other even though they have private addresses. The custom version of the eMule client was developed by ISP customers starting from 2003. The modification simply hardwires information about private IP addresses used within the ISP, and permits clients to send data to these addresses. We note that the default eMule system is associated with a DHT-based overlay known as *Kad*. The customized version of eMule (which we refer to as *ISP-Community*) builds a separate DHT overlay local to the peers in the ISP. This is achieved by modifying the message format of the original Internet-wide Kad overlay to ensure that the *ISP-Community* messages can only be processed by peers running the modified version.

Besides avoiding the NAT issue, *ISP-Community* offers other advantages. First, it is desirable to download content from other users connected to the same ISP since hosts within the ISP are interconnected through higher capacity backbone links. Second, a large percentage of hosts in the ISP are connected by FTTH, and their upload capacity is significantly higher than upload capacities of hosts connected to other ADSL providers. Third, given that all the peers in the community are in the same European country, the content that is available matches the interest and language preferences of users in the community. Finally, we note that *ISP-Community* clients could still use the global eMule system if content is not located within the local network. But this event is rare, as we will show in Section 4.4.2.

4.2.2 P2P Community in a Campus Network

The second network we analyze is a large university campus in North America with tens of thousands of end hosts in its network, interconnected by a high capacity backbone LAN. Users in the campus network are offered Fast Ethernet connections (100Mbps). In contrast to the ISP network, hosts in the campus receive public IP addresses, guaranteeing plain connectivity.

Motivated by the high bandwidth provided to clients in the campus network, students deployed a modified version of DC [100]. DC is a well known application for content sharing and chat, and we refer to the modified version as *Campus-Community*. In the traditional DC system, peers connect to a central *hub* for most of the system operations. However, in *Campus-Community*, there is no central hub, but a set of *hub clients* to avoid a single a point of failure. A *hub client* runs together with the DC application at each peer. When a peer is searching for content, a gnutella-like flooding algorithm is performed, in which peers forward the query to all their neighbors, until all peers receive the query. All peers that are sharing the content will reply back.

To enforce a closed membership, *Campus-Community* peers have been modified to only accept and initiate connections to other peers in the IP address range of the

campus. As a side-effect, the *Campus-Community* traffic is therefore highly localized to the campus network. In addition, we found that *Campus-Community* is the most popular P2P application in the campus. We identified over a thousand *Campus-Community* peers which contribute to a large fraction of the campus traffic.

4.3 Evaluation Goals and Methodology

In this section we present our goals and methodology.

4.3.1 Goals

The aim of this paper is to characterize and compare the closed network-specific P2P community systems against generic, open and Internet-wide P2P systems. To accomplish this, in this paper we seek answers for the following questions:

- How extensive is the use of P2P communities?
- How does the performance seen by users with P2P communities compare to performance seen with generic P2P systems?
- How does the Internet access technology of clients impact the performance of users of the P2P community?
- What are the implications of the growth of P2P communities in terms of traffic on network links for network providers?

We answer these questions with a combination of measurements and simulations. The network measurements help us characterize application performance, user behavior, and provide realistic traffic information to guide our simulations. The simulations enable us to study the implications of the communities on network providers. Based on actual traffic data derived from our measurements, we devise a methodology to infer the volume of P2P traffic each link of the network has to carry. We defer fur-

ther details on the simulation to Section 4.7 and focus on the methodology for the measurement study for the rest of the section.

4.3.2 Trace Collection Tool

Traces are collected with Tstat [45], a passive sniffer with advanced traffic classification capabilities. Starting from packet level traces collected in operational networks, Tstat groups packets into flows which are classified by application using a combination of Deep Packet Inspection and statistical classifiers, specifically targeting both plain and obfuscated P2P traffic. Tstat has been found to perform well in [103].

For each flow, Tstat collects various per-flow statistics such as bytes exchanged in each direction, flow duration and Round Trip Time (RTT) of packets. We refer the reader to [45, 104, 105] for more details.

4.3.3 Datasets

Our analysis is conducted on the following datasets:

ISP Network: Traces have been collected from two PoPs in a nation-wide ISP in Europe. A high-end PC running Tstat was used to analyze in real time all the packets going to and coming from all the hosts in the monitored PoPs, and produced a flow level log that has then been post-processed. The two PoPs are different in the type of Internet access technology of their hosted customers. In the first PoP, which we call *ISP-FTTH*, all customers are connected through FTTH links while in the second PoP, which we call *ISP-ADSL*, all customers are connected through ADSL links. For the *ISP-FTTH* PoP analysis, we focus on a one week trace collected during December 2008, with about 2,200 active customers in the PoP contacting over 4 million hosts. For the *ISP-ADSL* PoP analysis, we focus on a one day trace collected during April 2009, with about 20,000 active customers in the PoP contacting over 2 million hosts.

For these datasets, we label clients according to their access technology to the network. This information has been encoded by the ISP on the IP address of clients

and is easily obtainable from the traces. In addition, we associate clients with the ISP PoP were they reside. This information has been provided by the ISP operators.
Campus Network: The trace has been collected at the edge of some campus dormitories, using a methodology similar to the ISP setting. We report results from a 13 hours trace of a weekday in April 2009, during which there were about $2,000$ distinct active hosts in the monitored dormitories. These hosts contact more than a million other hosts.

4.3.4 Comparing P2P Communities with Generic P2P Systems

Our measurement studies compare the performance of closed P2P communities with generic Internet-scale P2P systems observed in the same network, and at the same time. We compare the *Campus-Community* and *ISP-Community* systems to two other regular, well known and open P2P applications: (i) the traditional eMule [32] application, which we refer to as the *ISP-Generic*; and (ii) the BitTorrent [106] application, which we refer to as *Campus-Generic*. Both *ISP-Generic* and *Campus-Generic* are the second most popular P2P file sharing systems after the P2P community applications in the respective traces. We note that our comparisons in the campus setting are based on two different underlying systems (the *Campus-Community* system is based on DC while the *Campus-Generic* is based on BitTorrent). While ideally, the comparisons are best performed using the same underlying system, this is infeasible since the campus network does not have a closed P2P community and a generic variant both based on the same underlying system. Thus, our comparisons in the campus network case could be impacted by other differences in the underlying systems involved, besides the open/closed nature of the systems. However, we believe the impact of these differences is relatively minor for most of the results, and the comparisons do provide important insights in the context of our study.

4.4 Characterizing Community Usage

In this section, we begin by characterizing the extent to which P2P communities are used in the monitored networks. Then, we consider the degree to which users rely on the P2P communities to access content.

4.4.1 Prevalence of Communities

Tables 4.1, 4.2 and 4.3 summarize general statistics for the ISP and campus traces. For all tables, the first column shows for the various systems, the number of peers identified inside the monitored PoPs in the ISP and the dormitories in the campus. The second column shows the total number of external peers that are contacted by the monitored hosts. The third column gives the total data exchanged by peers, while the fourth column provides the average data exchanged per peer.

First, notice in Table 4.1 that the *ISP-Community* population in the monitored PoP is almost twice as large as the *ISP-Generic* population, with 858 *ISP-Community* peers versus 470 *ISP-Generic* peers. In addition, *ISP-Community* peers generate over 3 times more connections and exchange over 13 times more data than *ISP-Generic* peers, and each peer exchanges over 7 times more traffic when using *ISP-Community*. Table 4.2 shows similar results for the *ISP-ADSL* trace. However, notice that the difference in the total data exchanged between *ISP-Community* and *ISP-Generic* is not as large as in the *ISP-FTTH* trace. This is because ADSL peers cannot upload as much data as FTTH peers due to their limited upstream capacity. As we will see later, the higher upload capacity of FTTH peers plays a key role and justifies the success of *ISP-Community*. Finally, note that the total number of distinct *ISP-Community* peers found when combining the two traces, amounts to around $600,000$ peers.

Table 4.3 shows similar results: the *Campus-Community* population in the PoP is 1.4 times larger than the *Campus-Generic* population. Besides, *Campus-Community*

Table 4.1
Traffic Summary for the *ISP-FTTH* Trace

Systems	Peers monitored	Peers Contacted	Total Data Exchanged	Per Peer Data Exchanged
ISP-Community	858	497.4K	9141GB	10.65GB
ISP-Generic	470	317.7K	683GB	1.45GB

Table 4.2
Traffic Summary for the *ISP-ADSL* Trace

Systems	Peers monitored	Peers Contacted	Total Data Exchanged	Per Peer Data Exchanged
ISP-Community	7074	325.1K	6669GB	0.94GB
ISP-Generic	2351	829.4K	1021GB	0.43GB

Table 4.3
Traffic Summary for the Campus Trace

Systems	Peers Monitored	Peers Contacted	Total Data Exchanged	Per Peer Data Exchanged
Campus-Community	270	1.82K	970GB	4.61GB
Campus-Generic	196	1.006M	14.65GB	0.28GB

peers exchange 60 times more data than *Campus-Generic* peers in total, and each peer exchanges 16 times more data when using the *Campus-Community*.

Figure 4.2 shows the fraction of inbound and outbound traffic observed at the monitoring point which may be attributed to the P2P communities. There are three groups of bars, corresponding to the campus, and ISP scenarios. The fraction due to the communities is computed for each hour of the trace and the 50th and 90th percentile of these values is shown. The results show that throughout the duration of the trace, close to 90% of the outbound traffic may be attributed to the P2P communities for the Campus and *ISP-FTTH* settings. In the case of *ISP-ADSL*, the

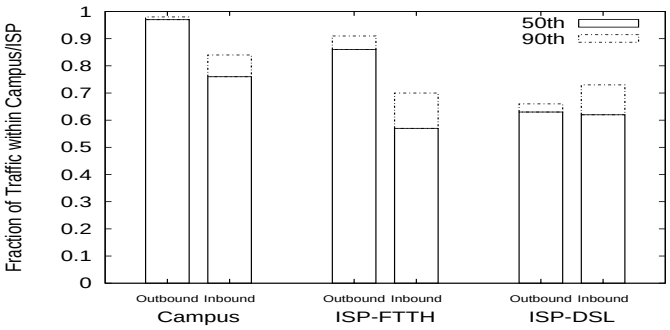


Fig. 4.2. Fraction of inbound and outbound traffic seen at the monitoring point, generated by the P2P communities. Note that all this traffic stays within the Campus/ISP.

P2P community outbound traffic is reduced to around 60% of the total outbound traffic. This is because of the limited upload capacity of ADSL customers. On the other hand, the fraction of inbound traffic, while slightly lower due to the higher fraction of HTTP traffic, is still over 60% in the three settings. The results clearly illustrate the overwhelming popularity of the P2P communities.

4.4.2 User Reliance on Communities

We were interested in measuring the extent to which users rely on the community to obtain the content they require. To evaluate this, we measure the community usage ratio $U(p, \tau)$, of peer p during time interval τ by considering the ratio of bytes p downloads from other peers using the community to the total bytes downloaded using any P2P system (including the community). We selected τ to be 1 hour long. Intuitively, this metric quantifies the extent to which a peer has to back up to an Internet-wide generic P2P system to retrieve content which cannot be found in the P2P community.

Figure 4.3 shows the mean $U(p, \tau)$ per client. The plot reports two bars for each of *Campus-Community* and *ISP-Community*. Each bar shows the fraction of clients with mean $U(p, \tau)$ greater than 80% (and 90%) for the community they are part

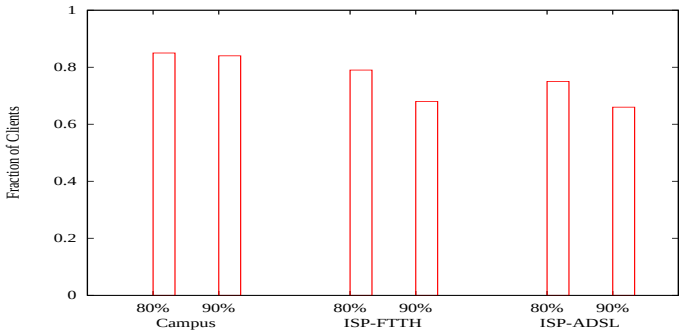


Fig. 4.3. Fraction of clients in *Campus-Community* and *ISP-Community* with a community usage ratio over 80%(90%).

of. In general, $U(p, \tau)$ is high. In particular, for *ISP-Community*, 70% of the peers download more than 90% of the data from the community. For *Campus-Community*, the fraction of clients is even larger, e.g., more than 90% of the peers download more than 90% of data from the P2P community. This suggests that the P2P community is self-sustaining and peers usually locate content within the community.

4.5 User and System Performance

In this section, we study the performance seen by users with the P2P communities, and compare this to the performance of the generic P2P systems. The most direct metric to evaluate user performance is the file transfer completion time. However our dataset does not allow us to gather this information since our probes do not perform parsing and interpretation of application header. Instead, we evaluate the download and upload rates and the delay per connection and per host, which are metrics clearly related to the overall user performance.

4.5.1 Throughput

In this section, we investigate to what extent users achieve better throughput if they use the P2P community. We then consider various factors that could affect the performance of users, such as their access technology.

Figure 4.4 presents results for users' performance considering the per host download rate $D(h)$, which is the average download rate across all TCP connections initiated or received by a host[1]. Again, the median and 90th percentile of the distribution among hosts is reported. We observe that for the campus setting, the median improvement in download rate of the P2P community over the generic system is a factor of 740. However, the improvement is only a factor of 6 and 4 for the *ISP-FTTH* and *ISP-ADSL* settings respectively.

Impact of access technology: We further study the ISP setting to understand the impact that heterogeneity in access technology of users has on the download performance. To provide appropriate context, Table 4.4 shows the breakdown of ISP users by access technology. For each trace, the first column shows the total number of external contacted peers. The second and third columns detail the total number of such peers connected by FTTH or ADSL links respectively. The fourth column shows peers for which we do not have access technology information. Neglecting the latter group, we notice that for both traces, there is a ratio of 1:5 between high speed FTTH links and slower and more congested ADSL links, which reflects the provider technology penetration.

Figure 4.5 shows the Cumulative Distribution Function (CDF) of the per-connection download bitrate for different types of sources contacted by *ISP-Community* clients in the *ISP-ADSL* trace. The curves labeled *ADSL Sources* and *FTTH Sources* refer to sources located inside the ISP connected by a particular access technology. The third curve labeled as *Internet Sources* corresponds to *ISP-Generic* clients, and is shown for comparison purposes. This curve refers to connections served by sources outside the

[1]To avoid connections carrying only control messages, we consider connections with more than 50KB downloaded

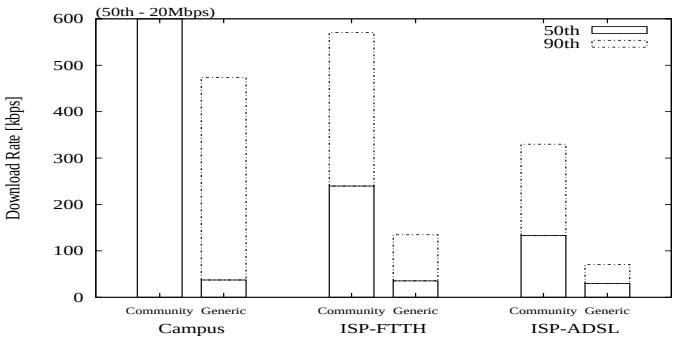


Fig. 4.4. Download rates for various systems.

Table 4.4
Access Technology of *ISP-Community* Peers

Trace	Total	FTTH	ADSL	Unknown
ISP-FTTH	497.4K	80.8K	356.4K	59K
ISP-ADSL	325.1K	52.8K	270.8K	1013

ISP for which no information about the peer access technology is available. The key observation from this graph is that the performance benefits of using *ISP-Community* is due to the high capacity FTTH sources which clearly enable much higher download rates. The 50th (90th) percentile of the download bitrate from FTTH sources is 120Kbps (591Kbps) compared to 23Kbps (64Kbps) for Internet sources, an improvement of five to nine times. Interestingly, only the top 20 percentile of connections involving ADSL sources perform better than connections involving Internet sources, and even here the difference in performance is minor. The same observations hold for connections from the *ISP-FTTH* trace. In particular, when FTTH clients download from FTTH sources, the median throughput is 10 times higher than when they download from both ADSL and Internet sources.

Degree of seed-like behavior in communities: We next investigate the extent to which users in the P2P community contribute more data than they receive and how this depends on the access technology of the user. We consider the ratio of bytes

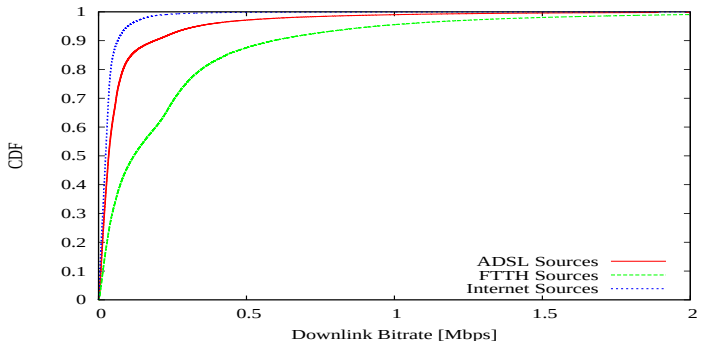


Fig. 4.5. Download rates from the *ISP-ADSL* trace, distinguishing the access technology of the source.

sent and bytes received per client in the whole dataset, which we call $R(p)$, where p is a peer. Notice that a value less than one represents clients that are mostly receiving (also called leechers) and a value larger than one represents clients that are mostly sending (or seeds); a value close to 1 represent clients that both send and receive data in similar amounts.

Figure 4.6 shows the CDF of $R(p)$ in *ISP-Community* for the *ISP-FTTH* and the *ISP-ADSL* traces. We clearly see that *ISP-Community* users behind an FTTH link have a more seed-like behavior. For instance, the median $R(p)$ for FTTH clients is 2.05 which shows that half of the clients send twice as much as they receive. In contrast, the median $R(p)$ for ADSL clients is 0.56 which implies a leecher-like behavior.

Implications: These results combined indicate that much of the performance improvement seen by *ISP-Community* clients over *ISP-Generic* clients stems from the fact that a small portion of the users in the ISP are connected by a high-speed FTTH technology, and these users contribute much more data than they receive. In addition, these results have broader significance for research on localizing P2P traffic within ISP boundaries [13,14]. While most research in this area has taken for granted that localizing traffic benefits users and ISPs, our results indicate that in ISPs with *heterogeneous* access technologies, the performance benefits to users on localizing P2P traffic is largely dependent on the degree of seed-like behavior of peers behind

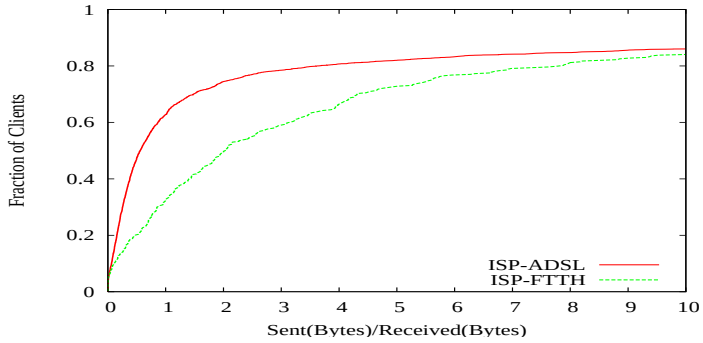


Fig. 4.6. Ratio of bytes sent and bytes received per client in *ISP-Community*.

high-bandwidth access technologies. We elaborate further on these implications in Section 4.8.

4.5.2 Delay

An intuitive way to evaluate the effectiveness of P2P localization is by measuring the delay of localized connections. One would expect that connections in the community system will exhibit a lower RTT than connections in the generic system. Figure 4.7 shows the per connection average RTT for the three vantage points. We observe that while the difference in RTT between *Campus-Community* and *Campus-Generic* is prominent, it is less noticeable between *ISP-Community* and *ISP-Generic*. To understand this better, we consider the impact of ISP clients behind ADSL lines on the RTT of connections. We observe that the distribution of RTT for *ISP-Community* connections to peers behind ADSL access links is comparable to *ISP-Generic* flows, while a significant improvement is noticeable for sources behind FTTH access links (which have a ten times higher upload capacity). This implies that congestion in the upstream direction of ADSL clients is causing queueing delays which subsequently increases RTTs.

A second reason for the lack of RTT improvements between *ISP-Community* and *ISP-Generic* is that many clients in *ISP-Generic* tend to access content within the

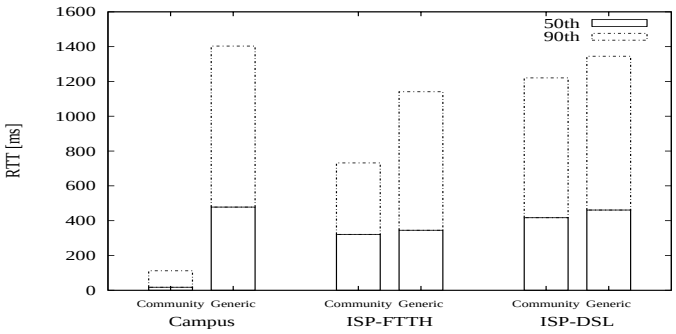


Fig. 4.7. RTT for connections initiated for various system.

same small European country. For instance, in our *ISP-ADSL* dataset, we found that 52% of connections leaving the ISP were destined to the same country and 81% to Europe. More generally, this is because users in the same geographic region tend to be interested in the same content, due to common language and culture [107]. We also note that *Campus-Community* users are mostly English speaking users and thus the content tends to be more spread throughout the world.

Implications: One could expect that localization of P2P traffic would reduce RTT and increase throughput of connections. But, our results show that these benefits may be limited by the access bandwidth of users inside the ISP and user demographics. Our observations agree with findings by other researchers [107].

4.6 Traffic matrix for P2P communities

In Section 4.5, we have focused on the extent to which user performance is improved when P2P communities are used. We next consider the impact that P2P communities have on network providers. We focus our study on the ISP network given it is a larger and more interesting setting, but a similar methodology could also be employed for the campus network. In this section, we present an approach to infer the application-specific traffic matrix due to *ISP-Community*. Each cell of the matrix corresponds to the volume of traffic related to *ISP-Community* between a

pair of PoPs in the ISP. Inference is required because we have direct measurements available only at two PoPs. We then employ this traffic matrix in Section 4.7 to study the load induced by *ISP-Community* on links of the ISP network, and examine how the load may change under various "what-if" scenarios (e.g., upgrade to higher access technology).

Traffic matrix estimation is a well studied problem [101,102]. However, past work has primarily focused on the estimation of the overall traffic matrix. In contrast, we explicitly target the estimation of the subset of traffic due to the P2P community, and specifically due to *ISP-Community*.

We leverage on the *Simple Gravity Model* [101], according to which the amount of flow exchanged between two objects is proportional to their "size". We consider the network PoPs as objects, whose size is determined by their peer population. Then, the traffic $T_{sent}(s,d)$ sent from PoP s to PoP d is simply defined by:

$$T_{sent}(s,d) = T_{sent}(s) * \frac{population(d)}{\sum_{k=1}^{n} population(k)} \tag{4.1}$$

where $T_{sent}(s)$ is the total P2P traffic sent by users in PoP s, and $population(d)$ is the population of PoP d. n is the total number of PoPs present in the topology. The model assumes that the fraction of traffic from s to d is simply proportional to the relative population of d. We believe this assumption is reasonable if (i) content is uniformly available in each PoP; and (ii) peer selection follows a uniform probability. The first point can be justified, since in *ISP-Community* all users belong to an ISP within the same country and therefore content can be located anywhere in the network. The second point was verified to be the case in *ISP-Community*, since the modified version of the software selects peers with uniform probability (i.e. no tit-for-tat policy is in place). In a more general scenario, where P2P systems may preferentially select nodes in certain PoPs (e.g., PoPs with lower latencies), a "friction factor" may be introduced in the model. We leave this generalization for future work.

4.6.1 Validation

We now describe our methodology to validate the model and present our results. Validating Equation 4.1 requires us to know $T_{sent}(s)$ and the number of *ISP-Community* peers in each PoP. While the former may be directly obtained from our traces, we assume the latter is simply the total number of *ISP-Community* peers in each PoP which contact our monitored *ISP-Community* peers through UDP control messages. We believe this is reasonable because: (i) Kad UDP control messages are sent to a larger number of peers compared to TCP data messages; and (ii) since Kad maps hosts to a DHT network at random, control messages are sent to any destination in the ISP with equal likelihood. To further confirm the validity of our methodology, we found that our monitored *ISP-Community* peers were contacted using UDP control messages by $497.4K$ unique peers in the *ISP-FTTH* trace and $325.1K$ unique peers in the *ISP-ADSL* trace. This is a very representative subset of the total *ISP-Community* population in the ISP. In addition, we noticed that the fraction of the population remains constant for every PoP across our traces and over different periods of time within a single trace. Hence, it may be enough to know the fraction of users per PoP rather than the actual total population of users.

We have predicted the amount of traffic sent to any other PoP in the network from the *ISP-FTTH* PoP and the *ISP-ADSL* PoP. To further show the validity of our model, when we consider the *ISP-FTTH* PoP as the source PoP, the population is estimated using the *ISP-ADSL* dataset and when we consider the *ISP-ADSL* PoP as the source PoP, the population is estimated using the *ISP-FTTH* dataset.

Figures 4.8 and 4.9 show the amount of traffic sent to any other PoP in the network considering the *ISP-FTTH* and *ISP-ADSL* PoPs as the source PoP respectively. Two curves are reported representing the model prediction $T(s,d)$, and the actual data measured from the trace, $\hat{T}(s,d)$. We observe that the model closely follows the real data in both cases.

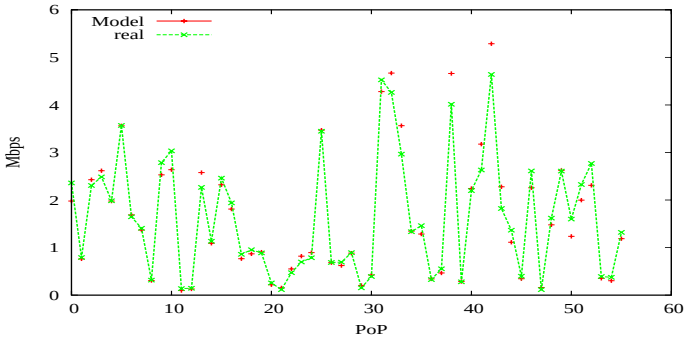


Fig. 4.8. Comparison of output from the gravity model and the real data for the *ISP-FTTH* trace.

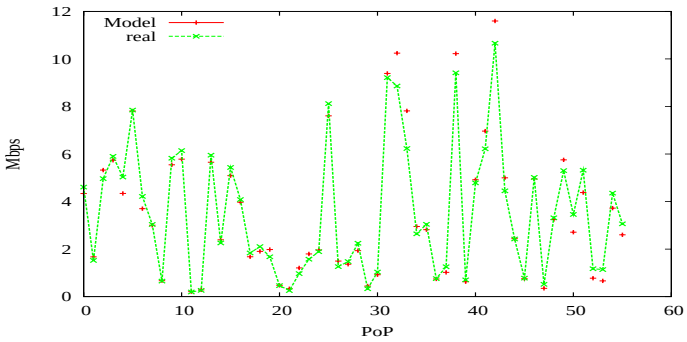


Fig. 4.9. Comparison of output from the gravity model and the real data for the *ISP-ADSL* trace.

To better quantify how close the output of the model is to the real data, we have calculated the relative error of the predictions, defined as $\frac{T(s,d) - \hat{T}(s,d)}{\hat{T}(s,d)}$. Figure 4.10 shows the CDF of the relative error for the *ISP-FTTH* and the *ISP-ADSL* traces. For around 90% of the PoPs, the error is smaller than 20% for both traces, which is an acceptable error margin.

These results confirm that a simple gravity model can be used to predict the traffic matrix of *ISP-Community*.

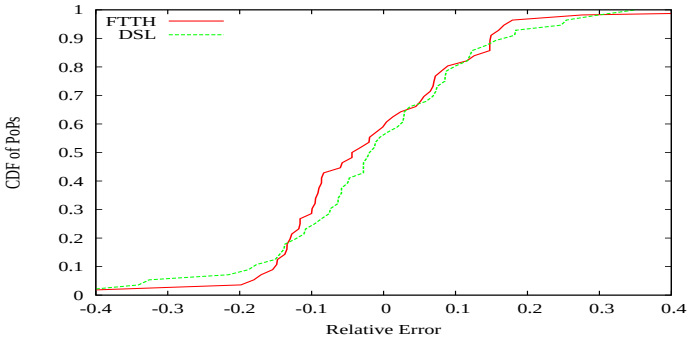


Fig. 4.10. Relative error of the gravity model for the *ISP-ADSL* and the *ISP-FTTH* traces.

4.7 Impact of Communities on the Network

In Section 4.6, we developed an approach to estimate the traffic matrix specific to *ISP-Community*. In this section, we use the estimated traffic matrix to compute the amount of *ISP-Community* traffic carried by individual network links. We also compute the difference in traffic on individual links under various "what-if" scenarios. In particular: (i) we consider a hypothetical scenario in which all clients currently using *ISP-Community* switch to using *ISP-Generic*. The purpose of this scenario is to shed light on how localizing traffic within the ISP impacts the capacity planning decisions of the ISP; and (ii) we evaluate how an upgrade of the access technology of customers may impact the network. We consider a scenario in which all customers in the ISP are upgraded to an FTTH access technology.

To conduct our analysis, we require knowledge of the ISP topology, the routing algorithm, and the traffic matrix corresponding to each scenario. In the rest of this section, we elaborate on our approach to modeling each of these aspects, and present simulation results.

4.7.1 Modeling Approach

Topology and routing: We model the topology based on the actual ISP network through discussions with the operator. In particular, nodes in the topology represent both the PoPs to which customers are connected, and the ISP backbone routers. Four types of links are present in the modeled topology: (i) *PoP-to-PoP* links directly connecting two PoPs in the same city, (ii) *PoP-to-backbone* links connecting a PoP to a backbone router, (iii) *backbone* links connecting two backbone routers typically between two cities and (iv) *peering* links that connect some backbone routers to the Internet. Traffic is routed on the topology using the standard shortest path algorithm, commonly employed in networks today, including the ISP that we consider.

Traffic Matrix estimation from the dataset: In Section 4.6 we have shown that the *ISP-Community* traffic exchanged by PoPs follows a simple gravity model. We leverage this to generate different traffic matrices to model possible scenarios. Besides the knowledge of the population of peers per PoP, Equation 4.1 relies on the availability of the total traffic sent $T_{sent}(s)$ by a given PoP s, which we can directly measure for only two PoPs. Following the gravity model assumption, we model $T_{sent}(s)$ as directly proportional to the population of peers in PoP s.

More in detail, let $n_f(s)$ and $n_a(s)$ be the number of FTTH and ADSL users in PoP s. Let $\hat{t}_f(s)$ and $\hat{t}_a(s)$ be the average amount of traffic that an FTTH and ADSL user in s generates during a given time interval respectively. Assuming that users corresponding to each PoP generate the same amount of average traffic, $\hat{t}_f(s) = \hat{t}_f \ \forall s$ and $\hat{t}_a(s) = \hat{t}_a \ \forall s$. $\hat{t}_f$ and $\hat{t}_a$ can then be estimated by considering the *ISP-FTTH* and *ISP-ADSL* datasets. Finally, the total volume of traffic sent from PoP s is simply proportional to the mix of access technology peers in the PoP, i.e.,

$$T_{sent}(s) = n_f(s)\hat{t}_f + n_a(s)\hat{t}_a \tag{4.2}$$

Equation 4.2 is then used to derive the *ISP-Community* traffic matrix T.

4.7.2 Predicting Traffic Matrix Changes

We now consider traffic matrices that P2P systems generate in different scenarios. We first consider the hypothetical scenario in which users in an ISP switch from *ISP-Community* to *ISP-Generic*. We also consider scenarios in which the ISP upgrades the access link of peers.

Users switch from *ISP-Community* to *ISP-Generic*: To evaluate this, we must construct the *ISP-Generic* traffic matrix T'. This is the scenario in which all the community traffic is directed to the Internet peering node z. We assume that the volume of traffic received by internal peers in this scenario is the same as in the current scenario, i.e., users are willing to download the same amount of data. More formally, $T'_{send}(z,d) = \sum_{s \neq z} T_{send}(s,d)$ and $T'_{send}(s,d) = 0 \ \forall d \neq z$.

In addition, we assume that the amount of traffic each PoP sends to z is the same as currently observed in the *ISP-Generic* system. We believe this is reasonable because independent of the upload capacity of peers inside the ISP, once connections leave the network, their throughput is likely to be limited by the destination access link or by some congested or rate limited intermediate link. More formally, $T'_{send}(s,z) = T'_{send}(s)$, where $T'_{send}(s)$ is computed as in Equation 4.2 considering the *ISP-Generic* dataset to estimate $\hat{t}_a$ and $\hat{t}_f$.

Technology upgrade: To consider the technology upgrade from the current ADSL and FTTH mix to an all-FTTH scenario, we assume that all ADSL clients send and receive *ISP-Community* traffic at the FTTH rate. Therefore $T''_{send}(s) = n_a(s)\hat{t}_f + n_f(s)\hat{t}_f$ and the traffic matrix T'' is computed as in Equation 4.1.

Finally, we consider a scenario in which users switch to the *ISP-Generic* system, while access technology is upgraded to FTTH. This allows us to compare the traffic on individual network links with *ISP-Generic* and *ISP-Community*, in an upgraded access technology setting. The traffic matrix is then obtained similarly to T', but $\hat{t}_a = \hat{t}_f$ are estimated as in T''.

4.7.3 Results

Figure 4.11 shows the CDF of the volume of P2P community related traffic that traverses each ISP link. For ease of presentation we call *GenericOnly* to the scenario where only *ISP-Generic* is present in the ISP, *CurrentTech* to today's mix of access technologies in the ISP and *AllFTTH* to the technology upgrade to an all-FTTH scenario. There are four lines, one for each combination of *ISP-Community* or *GenericOnly* for the P2P system in the network, and *CurrentTech* or *AllFTTH* for the access technology of clients. Logarithmic x-scale is used to better highlight the differences. We draw several observations from this plot. First, as expected, the usage of *ISP-Community* greatly reduces the traffic at peering links. However, more surprisingly, over 90% of the links carry a larger amount of traffic in the presence of *ISP-Community* as compared to the *GenericOnly* scenario. For instance, the median of *ISP-Community-CurrentTech* is 4.6 times larger than the median of *GenericOnly-CurrentTech*. Second, notice that *ISP-Community* makes use of more links in the network. While in the *GenericOnly* scenarios, 30% of the links are unused (mostly PoP-to-PoP links), in the *CurrentTech* scenarios, more than 95% of the links are being used and most of them are carrying more than 100GB per day. Third, when all peers are upgraded to FTTH, the traffic on links increases by almost an order of magnitude when comparing *ISP-Community*-CurrentTech and *ISP-Community*-AllFTTH. This is due to the higher upload capacity of peers. Hence, enhancing the capacity of ISP peers to get the full benefits of localization can probably hurt the network.

To get more insight into this, Figure 4.12 shows the difference in volume per link comparing the *GenericOnly-CurrentTech* and *ISP-Community-CurrentTech* scenarios. A negative value corresponds to an increase of link load when *ISP-Community* is used, while a positive value indicates an increase in link load with *ISP-Generic*. Results are separately reported for each class of links, i.e., backbone (top left plot), backbone-to-PoP (top right plot), PoP-to-PoP (bottom left plot) and peering links (bottom right plot) respectively. We draw several observations. First, we notice that

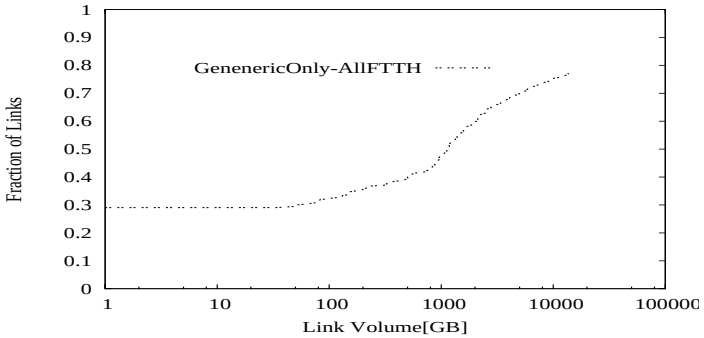


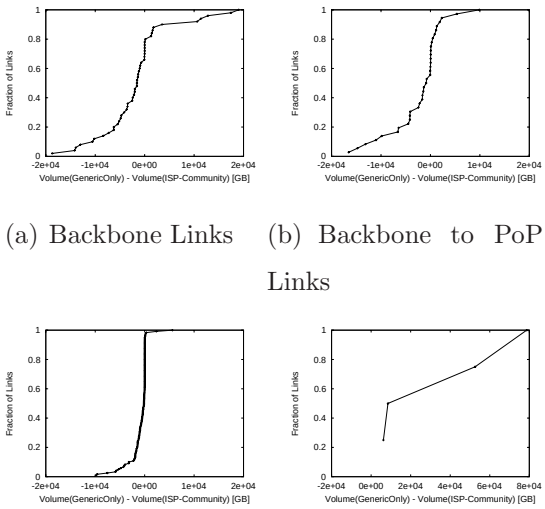
Fig. 4.11. CDF of volume of traffic related to P2P communities, seen by all links in the ISP topology, for the *ISP-Community* and *GenericOnly*, varying the access technology of peers.

more than 60% of backbone links see an increase in the volume of traffic they have to carry in the *ISP-Community* scenario, with some links seeing as much as $2 * 10^4$ GB of additional traffic each day. We make a similar observation for backbone-to-PoP links and for PoP-to-PoP links. However, for the latter, the increase in traffic is not as large in the *ISP-Community* scenario since those links interconnect PoPs which may exchange small amounts of data. Finally, as expected, *ISP-Community* is able to reduce the traffic at the peering links. We note that on average, peering links have to transport 36.5TB less traffic per day, or 1.5TB less traffic each hour in the *ISP-Community* scenario.

Implications: Overall, these results show that the extensive use of the P2P community has very significant implications for network providers in terms of traffic that individual links carry. While the use of the P2P community reduces traffic at peering points, it greatly increases traffic on interior links of the network, and consequently has important implications for capacity planning algorithms of the network provider.

4.8 Discussion and Implications

We discus key aspects of our work and implications below:



(a) Backbone Links (b) Backbone to PoP Links

(c) PoP to PoP Links (d) Peering Links

Fig. 4.12. Difference in traffic per link considering the *ISP-Community-CurrentTech* and *GenericOnly-CurrentTech* scenarios. There is one plot for each link type.

Implications for P2P traffic localization: Our characterization of closed P2P communities offer important lessons for research on localizing traffic of generic P2P systems within ISP boundaries [13–21]. While a majority of research in this area has taken the benefits of localization on users and ISPs for granted, our results support recent works [107, 108] which argue that a more critical examination is essential.

First, our results indicate that the benefits of localization depend on the access bandwidth of peers inside the ISP, as pointed out by recent works [107, 108]. For instance, while the throughput of connections of *Campus-Community* is significantly improved due to the high bandwidth campus LAN, the improvement in throughput of *ISP-Community* connections is limited by the fact that 80% of the users are behind an ADSL link. Further, the RTT of *ISP-Community* connections does not show significant improvement compared to *ISP-Generic* due to both the access technology of users, as well the fact that most users of *ISP-Generic* tend to access content from the same European country.

Second, going beyond [107, 108], our results indicate that in ISPs with *heterogeneous* access technologies, the performance benefit to users on localizing P2P traffic is largely dependent on the *degree of seed-like behavior* of peers behind high-bandwidth access links. For instance, the performance with *ISP-Community* is better than with *ISP-Generic* primarily due to a small number of users behind high-bandwidth FTTH connections, which contribute much more data than they receive. These observations also imply that in ISPs with *heterogeneous* access technologies, not all stakeholders can simultaneously win when P2P traffic is localized. In particular, (i) if high-bandwidth users show seed-like behavior, then the ISP and users behind low-bandwidth access technologies benefit on localization (through reduced transit traffic and improved performance), at the expense of high-bandwidth users; and (ii) alternately, localization benefits the ISP alone, and does not help improve the performance of users (in particular, users behind low-bandwidth access technologies). This situation can potentially be offset through new models for charging for access, where ISPs can incentivize seed-like behavior of high-bandwidth users by reducing charges on them.

Finally, our simulation results show that use of *ISP-Community* rather than *ISP-Generic* does result in lowered traffic on peering links for the ISP as expected. However, more interestingly, over 90% of internal links of the ISP network see higher traffic, with some links seeing as much as $2 * 10^4$ GB of additional traffic each day. This increase may potentially require internal link upgrades to avoid impairing user performance. These results suggest that coarse-grained schemes for localizing P2P traffic like *ISP-Community* (where peers within an ISP are selected at random), may potentially cause significant shifts in ISP traffic patterns. Developing techniques to predict such shifts is an interesting area of future research and we have taken a first step in this direction. Studying the effect of traffic shifts with more fine-grained localization schemes (for e.g., [20]) is another direction for future work.

Implications for traffic studies suggesting decline of P2P traffic in the Internet: Recent work has suggested that P2P traffic is on the decline [30]. However,

this finding is based on analysis of inter-AS traffic data, and traffic internal to ISP networks is not considered. In our traces however, private communities account for $60-90\%$ of all traffic. This indicates that traffic related to such private communities should be taken into consideration before making conclusive statements about the decline of P2P traffic.

Generality of our work: In this paper, we have characterized P2P communities in two networks. We believe our work is an important start in creating an awareness and understanding of such communities, an area that has received little attention to date. Analyzing a wider range of communities is a challenge given that this requires knowledge of which networks contain P2P communities, and involves traffic collection inside each of the networks. Obtaining access to traffic data from additional networks that contain P2P communities, and analyzing P2P communities in a broader range of networks is an important area for future research, and a subject of our ongoing investigations.

4.9 Related Work

We have already extensively discussed how our work relates to work on P2P traffic localization in Section 4.8.

There has been awareness in the research community about the presence of darknets [109–112]. While darknets are related to our work in that they are also typically closed P2P communities, they share important differences from the types of communities we consider. Darknets are motivated by the primary goal of anonymized sharing of content. In contrast, P2P communities are motivated by other factors such as ensuring good application performance and ensuring hosts with private addresses in the same ISP may communicate. Further, we focus on communities localized to particular networks, while darknets could extend across the Internet. A notable recent work [112] characterized operational BitTorrent darknets, focusing the analysis on identifying the type of content being shared and the level of collaboration between

peers. In contrast, our focus is on P2P communities, performance seen by users of the community and the impact of the community traffic patterns on the service provider.

Our own prior work [113] has pointed out the presence of the closed community in the ISP setting. However, the focus of that work was on detecting undesirable traffic patterns of a range of P2P systems, including the P2P community. In contrast, our focus in this work is on characterizing the performance and network localization of these systems and simulations to study implications on network providers. Further, we consider a completely new campus network community in this paper.

Models for traffic matrix estimation (for e.g., [101,102]) have been widely studied. Our efforts are distinguished in that we focus on developing an application-specific traffic matrix model that only considers traffic due to the P2P community application.

4.10 Conclusions

In this paper, we have raised the awareness of the research community to the prevalence of closed P2P communities, and have presented one of the first and most extensive characterizations of such communities. P2P communities are the most popular P2P systems used and generate most of the traffic in the networks we considered. For instance, we identified about $600,000$ unique peers in *ISP-Community*, that exchange 50 times more traffic than *ISP-Generic* and accounts for 60% to 90% of all the traffic observed in our traces. While as expected, users of P2P communities see better performance than users of generic P2P systems, we have shown that the extent of benefits is largely determined by the access technologies of the users, and the degree of seed-like behavior shown by users behind high-speed access technologies. We have developed techniques to enable network providers understand how the growth of P2P communities impacts network capacity planning. and how projected changes in access technologies of users may affect these results. Using the techniques, we show that while use of the communities does lower traffic on peering links compared to the use of generic P2P systems, it could greatly increase traffic on internal network

links. Our characterization of P2P communities while interesting in its own right, offers important lessons for research on localizing P2P traffic within ISP boundaries. Our future work includes studies of a wider range of P2P communities, and exploring refinements to the P2P community traffic models that can result in greater accuracy.

5. DISSECTING VIDEO SERVER SELECTION STRATEGIES IN THE YOUTUBE CDN

5.1 Introduction

Over the last few years, video traffic has become prominent on the Internet. A recent report [30] shows that 15 to 25% of all Inter-Autonomous System traffic today is video. YouTube is probably the main source of video on the Internet today, with 2 billion videos viewed each day and hundreds of thousands of new video uploads daily [114]. It is the third most visited website in the Internet, according to `www.alexa.com`.

The rapid growth in popularity of YouTube has made it the subject of several research studies. Much of the research to date has focused on understanding user behavior, usage patterns and video popularity [10–12], while others [115] have looked at social networking aspects related to YouTube. Relatively fewer works have looked at the YouTube infrastructure itself, and large parts of its architecture and design remain unknown to the research community. A recent notable work [8] has greatly contributed to the understanding of the YouTube Content Distribution Network (CDN) through an in-depth analysis of traffic traces of a tier-1 Internet Service Provider (ISP). However, much of this analysis has focused on the architecture prior to the acquisition of YouTube by Google Inc. It is unclear to what extent these observations continue to hold today.

In this paper, we aim to obtain a detailed understanding of the YouTube CDN and to quantify its effectiveness. Specifically, we are interested in studying how users' video requests are mapped to YouTube data centers. We are interested in exploring the various factors that can influence the decision, such as user proximity, server load, and popularity of video content. Such insights can aid ISPs in their capacity planning

decisions given that YouTube is a large and rapidly growing share of Internet video traffic today. A better understanding could enable researchers to conduct what-if analysis, and explore how changes in video popularity distributions, or changes to the YouTube infrastructure design can impact ISP traffic patterns, as well as user performance.

Obtaining such understanding is challenging given the proprietary nature of the YouTube system. Even information such as the location of the data centers that store content is not publicly known. To tackle these challenges, we conduct an analysis of traffic from the edge of five networks - two university campuses and three ISP networks - located in three different countries and two distinct continents. We consider a one week-long dataset from each vantage point, all collected at the same time. This allows us to study the server selection algorithm under different scenarios, so that different phenomena may appear in some datasets but not in others. While prior work has analyzed traffic at the edge of a single campus network (for e.g., [10, 11]), our work goes far beyond in terms of the number and diversity of vantage points used.

As a first step, we map YouTube server IP addresses obtained from our datasets to the nearest data centers. Prior efforts at doing so [8,9], have either relied on geolocation databases [116], or on reverse Domain Name System (DNS) lookup that can provide information regarding the server location. However, while these techniques worked with the earlier YouTube architecture, we find they do not apply or perform poorly in the new design. Consequently, we use CBG [117], a well known delay-based geolocation algorithm to learn server locations.

Armed with server location information, we evaluate how user requests are mapped to YouTube data centers. We show that there are two mechanisms: The first is based on DNS resolution which returns the server IP address in a data center; the second relies on application-layer mechanisms in which the server initially contacted can redirect the client to another server in a possibly different data center. Our results indicate that, given a network, most requests are directed to a preferred data center. This is in contrast to [8] which indicated that the earlier YouTube infrastructure

would direct requests from a network to a data center proportional to the data center size. Further, our results indicate that the RTT between data centers and clients in a network may play a role in the selection of the preferred data center.

More surprisingly however, our results also show that there do exist a significant number of instances where users are served from a data center that is not the preferred. Our analysis is informed by techniques we employ to identify groups of YouTube flows that correspond to a single video request. A deeper investigation reveals a variety of causes. These include load balancing across data centers, variations across DNS servers within a network, alleviation of hotspots due to popular video content, and accesses of sparse video content that may not be replicated across all data centers. Overall the results point to the complexity of server selection algorithms employed in YouTube, and the myriad factors that must be considered for the successful design of a large video content distribution network.

5.2 YouTube Basics

YouTube is the most popular video-sharing website on which users can watch videos on demand. It was bought by Google Inc. in November 2006 and it is now integrated in the Google offering. In this section we present a high level description of the steps to retrieve a video from the YouTube system as sketched in Figure 5.1.

When accessing videos from the YouTube site at `www.youtube.com`, the user either browses the portal based system looking for the desired content, or accesses directly the video web page following a video page URL (step 1). Until the actual video web page is accessed, mostly static information and small thumbnails of suggested videos are presented.

Once the actual video has been selected, the front-end replies with a HTML page in which the video is embedded using an Adobe Flash Player plugin, that takes care of the download and playback of the video (step 2). The name of the server that will provide the video is among the parameters provided for the plugin and it is encoded

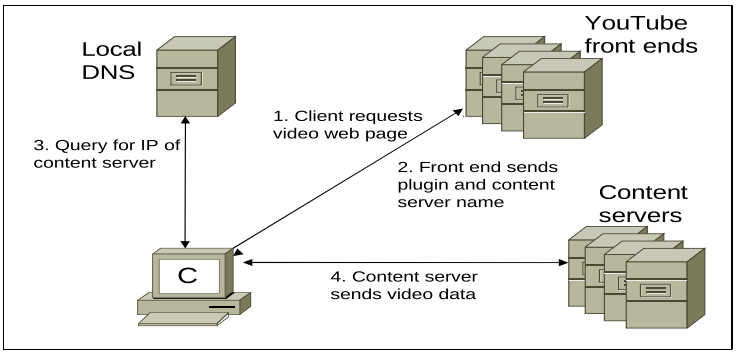


Fig. 5.1. High level sequence of steps to retrieve content.

using a static URL. Then, the content server name is resolved to an IP address by the client via a DNS query to the local DNS server (step 3). Finally, the client will query the content server via HTTP to get the actual video data (step 4).

We further elaborate on steps 3 and 4. First, the selection of the IP address by the local DNS server in step 3 is not arbitrary. In fact, the DNS resolution is exploited by YouTube to route clients to appropriate servers according to various YouTube policies, some of which we will discuss in this paper. Second, it is possible that the preferred server cannot provide the content and the client will be "redirected" by this server to a different one, possibly in a different data center.

5.3 Methodology

To understand the internal mechanisms of the YouTube CDN, we need to analyze the interactions between the user and the content servers. We introduce our data collection tool in Section 5.3.1, and describe our datasets in Section 5.3.2.

5.3.1 Collection Tool

Our traces are collected using Tstat [118], an Open Source passive sniffer with advanced traffic classification capabilities. Tstat identifies the application that gen-

erates TCP/UDP flows using a combination of Deep Packet Inspection (DPI) and statistical classifiers. Tstat was found to perform well in [119].

Tstat has the capability to identify major components of the current HTTP Web 2.0 traffic, including in particular YouTube traffic. Classification is achieved by using DPI technology to inspect the packet payload and then to identify YouTube service specific strings. In this paper we rely on Tstat's ability to identify actual YouTube video traffic, corresponding to the download of the Flash Video (flv) or H.264 (MP4) video file to be played back to the user by the Flash plugin. YouTube video downloads embedded in third party sites such as news sites or blogs are also correctly classified, since the same mechanisms are adopted by the Flash plugin. For more details on the classification algorithm implemented in Tstat, we refer the reader to the source code available from [120].

To uniquely identify a YouTube video, Tstat records the *video identifier* (VideoID), which is a unique 11 characters long string assigned by YouTube to the video. This is the same ID that is used when accessing the video web page in the URL. Furthermore, Tstat also records the actual resolution of the video being requested. At the end, the VideoID and resolution identify the actual video stream served to the player.

5.3.2 Datasets

Using Tstat, we collected datasets corresponding to flow-level logs where each line reports a set of statistics related to each YouTube video flow. Among other metrics, the source and destination IP addresses, the total number of bytes, the starting and ending time and both the VideoID and the resolution of the video requested are available.

We collected datasets from five locations spread across three countries including Points-of Presence (PoP) in nation-wide ISPs and University campuses. In all cases, a high-end PC running Tstat was installed to analyze in real time all the packets going to and coming from all the hosts in the monitored PoPs. For all these datasets,

Table 5.1
Traffic Summary for the Datasets

Dataset	YouTube flows	Volume [GB]	#Servers	#Clients
US-Campus	874649	7061.27	1985	20443
EU1-Campus	134789	580.25	1102	1113
EU1-ADSL	877443	3709.98	1977	8348
EU1-FTTH	91955	463.1	1081	997
EU2	513403	2834.99	1637	6552

we focus on a one week time period, between September 4th and September 10th, 2010. The collection from all vantage points starts at 12:00am, local time.

Table 5.1 summarizes the characteristics of the datasets, reporting the name, the total number of YouTube video flows and corresponding downloaded volume of bytes. Finally, the number of distinct IP addresses considering both YouTube servers and clients in the PoP are reported. In total, more than 2.4 millions YouTube videos have been observed by more than 37,000 users in the whole dataset.

We can divide the 5 datasets collected into two categories:

• **ISP Networks**: The datasets have been collected from nation-wide ISPs in two different European countries. EU1-ADSL and EU1-FTTH refer to data collected from two distinct PoPs within the same ISP. The two PoPs differ in the type of Internet access technology of their hosted customers. In EU1-ADSL, all customers are connected through ADSL links and in EU1-FTTH, all customers are connected through FTTH links. The EU1 ISP is the second largest provider in its country. The EU2 dataset has been collected at a PoP of the largest ISP in a different country.

• **Campus Networks**: The datasets have been collected using a methodology similar to the ISP setting. The Tstat PC is located at the edge of each of the campus networks, and all incoming and outgoing traffic is exposed to the monitor. We collected datasets from two University campus networks, one in the U.S. and one in a European country.

5.4 AS Location of YouTube Servers

We start our analysis studying the Autonomous System (AS) in which YouTube video servers are located. We employ the `whois` tool to map the server IP address to the corresponding AS. Table 5.2 presents our findings for each dataset. The second group of columns shows the percentage of servers and bytes sent from the Google AS. Not surprisingly, most servers are hosted in the Google AS (AS 15169). For instance, for the US-Campus dataset, 82.8% of the servers are located in the Google Inc. AS, serving 98.66% of all bytes. The third group of columns shows that a small percentage of servers (and an even smaller percentage of bytes) are still located in the YouTube-EU AS (AS 43515). We therefore have an evidence that since 2009 Google has migrated most content from the YouTube original infrastructure (that was based on third party CDNs) to its own CDN. The traffic served from the YouTube networks is probably because of legacy configurations. This contrasts with earlier studies such as [8,9], according to which the majority of servers were located in the YouTube AS (AS 36561, now not used anymore).

The fourth group of columns in Table 5.2 shows the percentage of servers and bytes received from within the same AS where the dataset have been collected. Note that the values are 0 for all datasets except EU2. The EU2 dataset indeed shows that a YouTube data center is present inside the ISP network. This data center serves 38.6% of the bytes in the EU2 dataset. This results in the EU2 dataset having fairly different performance than other datasets, as our analysis will reveal later.

Finally, the last groups of columns aggregates the percentage of servers and bytes sent from other ASes, among which CW (AS1273) and GBLX (AS3549) are the most likely one. This confirms therefore that YouTube servers can be both present inside an ISP, or in the Google network.

In the rest of this paper, we only focus on accesses to video servers located in the Google AS. For the EU2 dataset, we include accesses to the data center located inside the corresponding ISP.

Table 5.2
Percentage of Servers and Bytes Received per AS

Dataset	AS 15169 Google Inc.		AS 43515 YouTube-EU		Same AS		Others	
	servers	bytes	servers	bytes	servers	bytes	servers	bytes
US-Campus	82.8	98.96	15.6	1.03	0	0	1.4	0.01
EU1-Campus	72.2	97.8	20.3	1.6	0	0	7.5	0.6
EU1-ADSL	67.7	98.8	28	0.94	0	0	4.3	0.26
EU1-FTTH	70.8	99	24.2	0.83	0	0	5	0.27
EU2	62.9	49.2	28.6	10.4	1.1	38.6	7.4	1.8

5.5 Server Geolocation

In this section we present the techniques used to identify the geographical location of the YouTube servers seen in our datasets. The goal is to later use this information to analyze the video server selection policies.

• **Limitations of IP-to-location databases:** One common way to find the geographical location of an IP address is to rely on public databases [9]. While such databases are fairly accurate for IPs belonging to commercial ISPs, they are known to be inaccurate for geolocation of internal IPs of large corporate networks. For example, according to the Maxmind database [116], all YouTube content servers found in the datasets should be located in *Mountain View, California, USA*. To verify this, we perform RTT measurements from each of our vantage points to all content servers found in our datasets. Figure 5.2 reports the Cumulative Distribution Function (CDF) of the minimum RTT obtained to each server. We clearly observe that there is a lot of variation in the measurements, and in particular, many of the RTT measurements for the European connections are too small to be compatible with intercontinental propagation time constraints [121]. This indicates that all servers cannot be located in the same place.

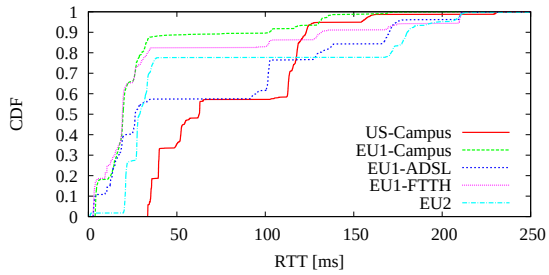


Fig. 5.2. RTT to YouTube content servers from each of our vantage points.

We note that Maxmind was useful in [9], probably because most YouTube servers in the old infrastructure were reported as located in San Mateo and Mountain View, California, USA. Further, a recent work [8] adopts a different approach, where the location of the server is obtained directly from the server name. However, this approach is not applicable to the new YouTube infrastructure, where DNS reverse lookup is not allowed. Therefore we decided to adopt a measurement-based approach to systematically localize YouTube servers.

• **Measurement based geolocation mechanism:** CBG [117] is a well-known geolocation algorithm that is based on simple triangulation. A set of landmarks is used to measure the RTT to a target. A simple linear function is then used to estimate the physical distance between each landmark and the target. This distance will become the radius of a circle around the landmark where the target must be located. The intersection among all circles is the area in which the target can be located.

We obtained the CBG tool from Gueye et al. [117] for our evaluations. We used 215 PlanetLab nodes as landmarks: 97 in North America, 82 in Europe, 24 in Asia, 8 in South America, 3 in Oceania and 1 in Africa. Then, we run RTT measurements from each landmark to each of the YouTube servers that have been found in our dataset, and identified the area in which they are placed.

In Figure 5.3 we evaluate the confidence region of CBG, i.e. the area inside which the target IP should be located. The picture shows the CDF of the radius of the

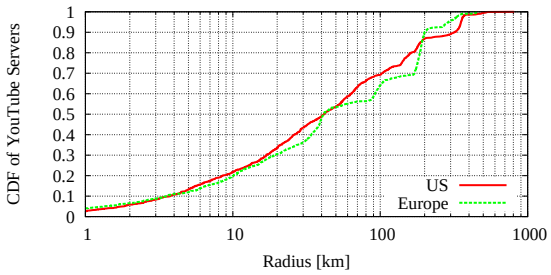


Fig. 5.3. Radius of the CBG confidence region for the YouTube servers found in the datasets.

confidence region for all servers found. Separate curves are shown for IPs in U.S. and Europe. Note that the median for both U.S. and European servers is 41km, while the 90th percentile is 320km and 200km, respectively. This is in the ballpark of the PlanetLab experiments presented in [117], where the 90th percentile for U.S. and Europe was about 400km and 130km. We can therefore consider the results provided by CBG to be more than adequate for our analysis.

• **Geolocation Results:** Table 5.3 details the result of using CBG to identify the location of all the destination IPs found in the datasets. The table shows the number of servers that are located in North America, Europe and other continents. Interestingly in each of the datasets, at least 10% of the accessed servers are in a different continent.

Finally, since several servers actually fall in a very similar area, we consider all the YouTube servers found in all the datasets and aggregate them into the same "data center". In particular, servers are grouped into the same data center if they are located in the same city according to CBG. We note that all servers with IP addresses in the same /24 subnet are always aggregated to the same data center using this approach. We found a total of 33 data centers in our datasets, 14 in Europe, 13 in USA and 6 in other places around the world. These results may not cover the complete set of YouTube servers since we are only considering those servers that appeared in our dataset.

Table 5.3
Google Servers per Continent on each Dataset

Dataset	N. America	Europe	Others
US-Campus	1464	112	84
EU1-Campus	82	713	1
EU1-ADSL	518	769	51
EU1-FTTH	90	631	44
EU2	233	815	0

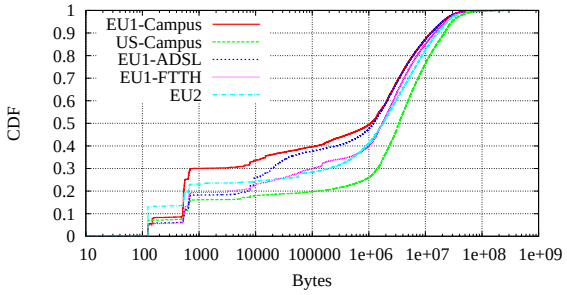


Fig. 5.4. CDF of YouTube flow sizes.

5.6 Evaluating YouTube's Server Selection Algorithm

In the previous section, we have shown how IP addresses of YouTube servers may be mapped to the appropriate YouTube data centers. Armed with such information, we now try to understand how user video requests are mapped to YouTube data centers. We are interested in exploring the various factors that can influence the decision, such as user proximity, server load, and popularity of content. We begin by discussing the various types of flows in a YouTube session, and then discuss how content servers are selected.

5.6.1 Video Flows and Sessions

In conducting our analysis, it is important to note that when a user attempts to download a video, the overall interaction may include a group of distinct flows, not

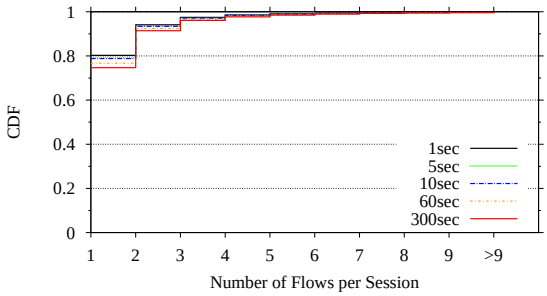


Fig. 5.5. Number of flows per session with different values of T for the US-Campus dataset.

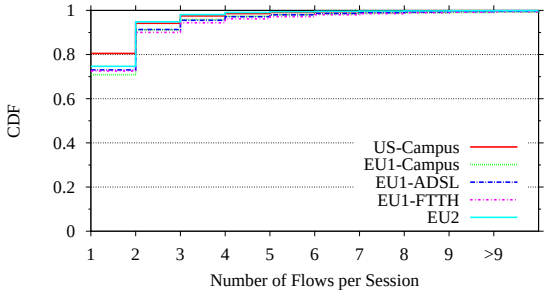


Fig. 5.6. Number of flows per session for all datasets using T=1 second

all of which involve transfer of video. In the normal scenario, each YouTube video request corresponds to a HTTP message exchanged between the Flash Plugin and a content server. If the request succeeds, then the content server starts to deliver the video inside the open connection. It is possible however that the server may not serve the content. In such a case, it would simply *redirect* the user to another content server and close the connection. There may be other possible responses from the server, for e.g., a response indicating that change of video resolution is required. Thus, more generally, according to the reply of the content server, we can distinguish between *video flows*, i.e., long connections carrying the requested video, and *control flows*, i.e., short connections carrying signaling messages.

Knowledge of control flows associated with a video flow can help provide important insights for our analysis. For instance, a video flow from a user to a given server preceded closely (in time) by a control flow to another server is an indication of

redirection. In contrast, an isolated video flow not preceded by other control flows is an indication that the request was directly served by the contacted server. We refer to such a group of related flows as a *video session*. Identification of video sessions aid our analysis as we will see later.

We now discuss how we identify video flows and sessions. Since Tstat classifies YouTube video flows based on the URL in the HTTP requests, it is not able to distinguish between successful video flows and control messages. To overcome this limitation, we employ a simple heuristic based on the size of the flows involved. Figure 5.4 presents a CDF of YouTube video flow sizes. Log-scale is used on the x-axis. We notice the distinct kink in the curve, which is due to the two types of flows. Based on this, we separate flows into two groups according to their size: flows smaller than 1000 bytes, which correspond to *control flows*, and the rest of the flows, which corresponds to *video flows*. We have conducted manual experiments which have confirmed that flows smaller than 1000 bytes are indeed control messages.

A *video session* aggregates all flows that i) have the same source IP address and VideoID, and ii) are overlapped in time. In particular, we consider two flows to overlap in time if the end of the first flow and the beginning of the second flow are separated by less than T seconds. In general, we find that small values of T will group flows triggered by the system, while large values of T may also group flows generated by user interactions with the video player, such as changing the video resolution and pausing or fast-forwarding a video. Since we are interested in capturing server redirections, which are triggered by the system, we want to use a small value of T, but that is large enough to avoid artificially separating related flows. Hence, we perform sensitivity to the value of T in our traces. We show results for the US-Campus dataset in Figure 5.5 and note that other traces show similar trends. Results indicate that values of T equal to 10 seconds or less generate similar number of sessions. So we pick the smallest value of T in our evaluations, $T = 1$ second.

Figure 5.6 reports the CDF of the number of flows per session for each dataset, assuming $T = 1$ second. It shows that $72.5 - 80.5\%$ of the sessions consist of a single

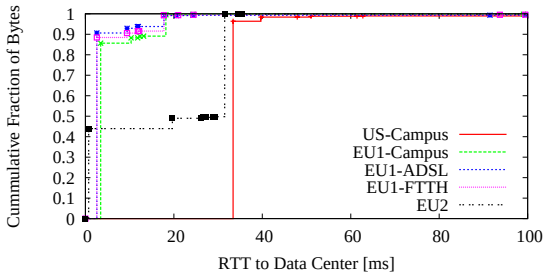


Fig. 5.7. Fraction of the total YouTube video traffic served by a data center with an RTT less than a given value from the dataset collection point.

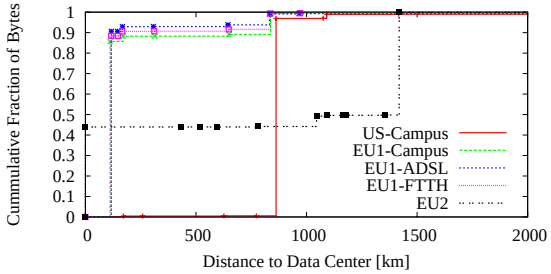


Fig. 5.8. Fraction of the total YouTube video traffic served by a data center with a distance less than a given value from the dataset collection point.

(long) flow. Therefore, normally there is no need to iterate over different servers to download the video data. However, $19.5 - 27.5\%$ of the sessions consist of at least 2 flows, showing that the use of application-layer redirection is not insignificant.

5.6.2 Understanding Server Selection Strategy

In Table 5.3 we have shown that the users in each dataset contact content servers all over the world. It is now interesting to investigate how the volume of traffic downloaded is spread across the different data centers. Figure 5.7 reports the fraction of traffic served by each data center versus the RTT between the vantage points and the data centers itself. In particular, we consider the minimum RTT seen by pinging

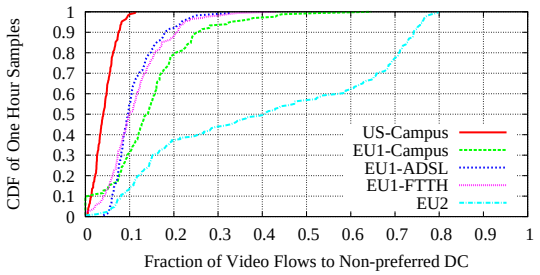


Fig. 5.9. Variation of the fraction of video flows directed to a non-preferred data center over time. One hour long time periods are considered.

all servers in each data center from the probe PC installed in the PoP. We observe that except for EU2, in each dataset one data center provides more than 85% of the traffic. We refer to this primary data center as the *preferred* data center for that particular trace and other data centers will be labeled as *non-preferred*. At EU2, two data centers provide more than 95% of the data, one of them located inside the ISP and the other outside in the Google AS. We label the data center with the smallest RTT in EU2 as the preferred one. We give a closer look to the EU2 case in section 5.7.1.

Further, we notice that the data center that provides most of the traffic is also the data center with the smallest RTT for each dataset. This suggests that RTT does play a role in the selection of YouTube servers. However, we have reason to believe that RTT is not the only criteria and that the preferred data center may change over time. For example, in a more recent dataset collected in February 2011, we found that the majority of US-Campus video requests are directed to a data center with an RTT of more than 100 ms and not to the closest data center, which is around 30 ms away.

Figure 5.8 considers the distance (in kilometers) between users and the data centers they are mapped to. In most cases, the data centers with the smallest delay to the customers are also the physically closest ones. This is not the case for the US-Campus dataset, where the five closest data centers provide less than 2% of all

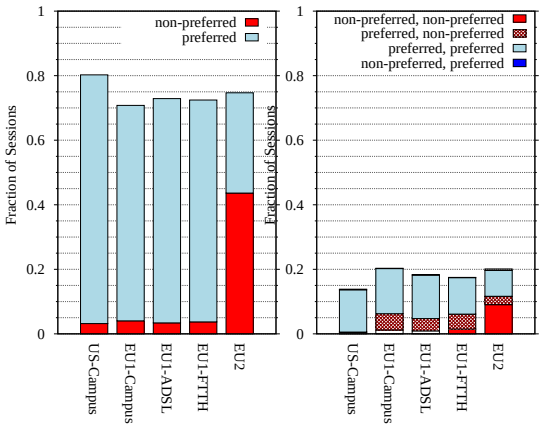
the traffic. Coupled with previous observations about RTT, this is an indication that geographical proximity is not the primary criterion used in mapping user requests to data centers.

The final observation we make is that although most traffic comes from the preferred data center that is typically very close to the customers, there are some exceptions in all datasets. For the US-Campus and the EU1 datasets, between 5% and 15% of the traffic comes from the *non-preferred* data centers. However, in EU2, more than 55% of the traffic comes from *non-preferred* data centers. We now are interested to see the variation over time of the fraction of traffic coming from non-preferred data centers. One hour-long time slots are considered, and the fraction of traffic served by non-preferred data centers in each of these time slots is determined. Figure 5.9 plots a CDF of these fractions. The results indicate that the fraction varies across time for most datasets, the variation being most prominent for the EU2 dataset. In particular for this dataset, 50% of the samples have more than 40% of the accesses directed to the non-preferred data center.

5.6.3 Mechanisms Resulting in Accesses to Non-preferred Data Centers

We have seen that a non-negligible fraction of video flows are downloaded from non-preferred data centers. There are at least two possible causes for this. A first possibility is that the DNS mechanisms direct a request to the non-preferred data center. A second possibility is that the request was redirected to another data center by the preferred data center server.

To disambiguate the two cases, we consider the video session associated with each flow, as discussed in Section 5.6.1. In the case that DNS mapped a request to a non-preferred data center, the video session must consist of a single video flow to a non-preferred data center, or must begin with a control flow to the non-preferred data center. In the other scenario, the session must begin with a control flow to the



(a) 1 flow sessions (b) 2 flows sessions

Fig. 5.10. Breakdown of sessions based on whether flows of the session are sent to preferred data center.

preferred data center (indicating the DNS mapping was as expected), but subsequent flows in the session must be to non-preferred data centers.

To better understand the effectiveness of DNS in mapping requests to the preferred data center, consider Figure 5.10(a). Each bar in the figure shows the fraction of sessions that involve only one flow. Further, each bar shows a break down of the requests sent to the preferred and non-preferred data centers. For instance, for US-Campus, 80% of the sessions involve a single flow; 75% are then served by the preferred data center while 5% of sessions are directly going to the non-preferred data center. Interestingly, about 5% of the single-flow sessions are directly served by the non-preferred data center for EU1 datasets too. For EU2 however, over 40% of the single flow sessions are served by the non-preferred data center. Overall, these results show that DNS is in general effective in mapping requests to the preferred data center. Still DNS mapping mechanisms do account for a significant fraction of video flow accesses to non-preferred data centers.

We next try to understand the extent to which users downloaded video from a non-preferred data center, even though they were directed by DNS to the preferred

data center. Figure 5.10(b) presents the breakdown of sessions involving 2 flows. These sessions group a control flow followed by a video flow. Based on whether each flow involves the preferred or non-preferred data center, we have four possible cases: (i) both preferred; (ii) both non-preferred; (iii) the first preferred and the second non-preferred; and (iv) the first non-preferred and the second preferred. Each bar in Figure 5.10(b) presents the breakdown among these patterns. For all the EU1 datasets, we see a significant fraction of cases where the DNS did map requests to the preferred data center, but application-layer redirection mechanisms resulted in the user receiving video from a server in a non-preferred data center. For the EU2 dataset, we note that a larger fraction of sessions has both flows going to the non-preferred data center, meaning that the DNS is still the primary cause for the user downloading videos from non-preferred data centers. We have also considered sessions with more than 2 flows. They account for $5.18 - 10\%$ of the total number of sessions, and they show similar trends to 2-flow sessions. For instance, for all EU1 datasets, a significant fraction of such sessions involve their first access to the preferred data center, and subsequent accesses to non-preferred data centers. We omit further results for lack of space.

5.7 Causes Underlying Non-preferred Data Center Accesses

In this section, we investigate why accesses to non-preferred data centers occur.

5.7.1 DNS-level Load Balancing

As shown in the previous section, the EU2 dataset exhibits very different behavior compared to other datasets. Over 55% of the video traffic is received from the non-preferred data center, and a vast majority of accesses to non-preferred data centers is due to the DNS mapping mechanisms.

To understand this better, consider Figure 5.11. The top graph presents the evolution over time of the fraction of video flows served by the preferred data center.

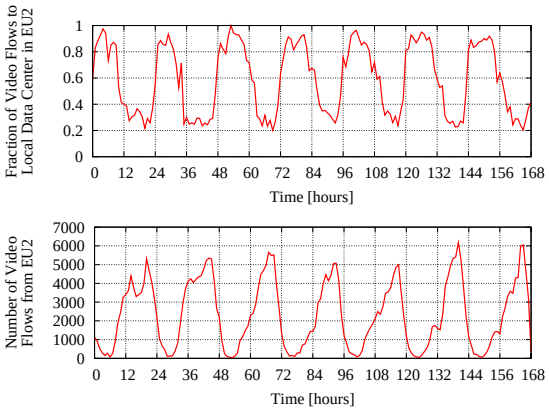


Fig. 5.11. Fraction of the total YouTube video traffic served by the preferred data center (top graph) and total number of video flows (bottom graph) as a function of time for the EU2 dataset.

One hour time slots are considered. The bottom graph shows the total number of video flows seen in the EU2 dataset as a function of time. Note that time 0 represents 12am on Friday. We can clearly see that there is a day/night pattern in this set of requests. During the night, when the total number of accesses from EU2 is small, the internal data center handles almost 100% of the video requests. However, during the day, when the number of requests per hour goes up to around 6000, the fraction of requests handled by the local data center is always around 30% across the whole week. Results for other datasets are not shown for the sake of brevity. Still, all datasets exhibit a clear day/night pattern in the number of requests. However, there is less variation over time of the fraction of flows served by the preferred data center, as already seen in Fig.5.9. Furthermore, there is much less correlation with the number of requests.

We believe the reason for this is the unique setup in the EU2 network. In this network, the data center inside the network serves as the preferred data center. While this data center located inside the ISP is the nearest to the users, it is unable to handle the entire load generated by users inside the EU2 ISP during busy periods. There is strong evidence that adaptive DNS-level load balancing mechanisms are in place,

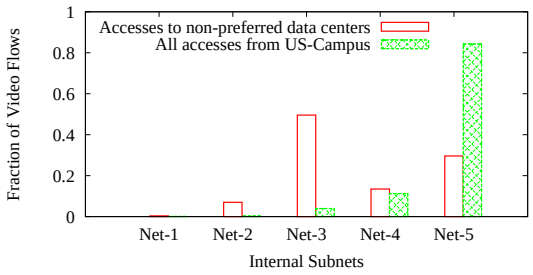


Fig. 5.12. Fraction of all video flows, and video flows to non-preferred data centers for each internal subnet of the US-Campus dataset.

which results in a significant number of accesses to the non-preferred data centers during the high load period of traffic.

5.7.2 Variations Across DNS Servers in a Network

Our results from the previous section indicate that for the US-Campus dataset most of the accesses to the non-preferred data center are caused by DNS. Deeper investigation indicates that most of these accesses may be attributed to clients from a specific internal subnet within the US-Campus network. Those clients indeed request significantly higher fraction of videos from non-preferred data centers than clients in other subnets. To see this, consider Figure 5.12. Each set of bars corresponds to an internal subnet at US-Campus. The bars on the left and right respectively show the fraction of accesses to non-preferred data centers, and the fraction of all accesses, which may be attributed to the subnet. Net-3 shows a clear bias: though this subnet only accounts for around 4% of the total video flows in the dataset, it accounts for almost 50% of all the flows served by non-preferred data centers.

Further investigation shows that hosts in the Net-3 subnet use different DNS servers that map YouTube server names to a different preferred data center. In other words, when the authoritative DNS servers for the YouTube domain are queried by the local DNS servers in Net-3, the mapping provided is to a different preferred data center than the other subnets on US-Campus. We believe this behavior is

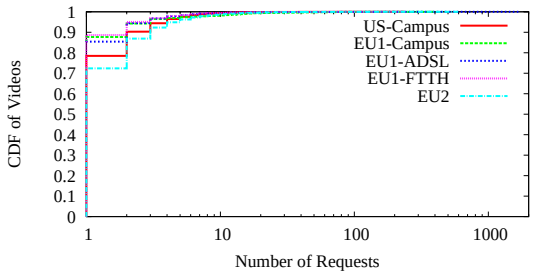


Fig. 5.13. Number of requests for a video to the non-preferred data centers.

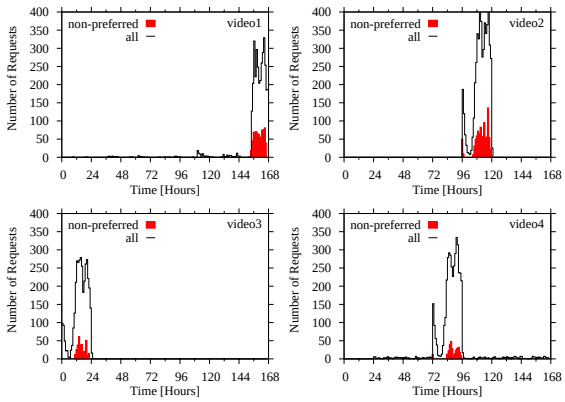


Fig. 5.14. Load related to the top 4 videos with the highest number of accesses to the non-preferred data centers for the EU1-ADSL dataset.

not a misconfiguration in the YouTube servers or the Net-3 servers, but we rather hypothesize that this is the result of a DNS-level assignment policy employed by YouTube, probably for load balancing purposes, which can vary between DNS servers and thus subnets that belong to the same campus or ISP network.

5.7.3 Investigating Redirection at the Application Layer

We now consider cases where users download video from non-preferred data centers, even though DNS mapped them to the preferred data center.

To get more insights into this, consider Figure 5.13 which reports the CDF of the number of times a video is downloaded from a non-preferred data center. Only

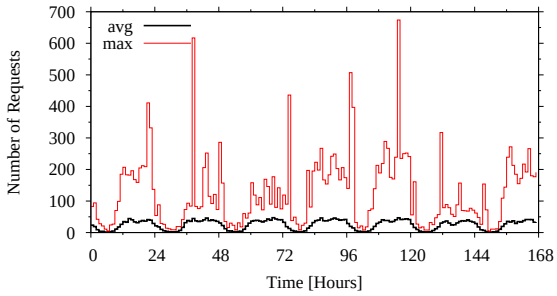


Fig. 5.15. Average and maximum number of requests per server in the preferred data center of EU1-ADSL dataset.

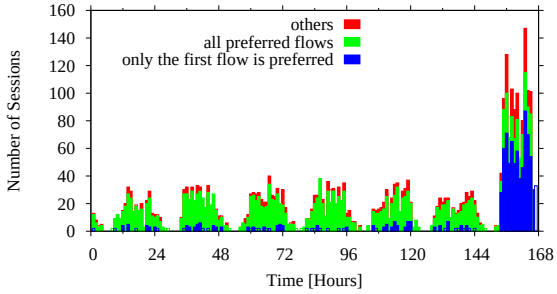


Fig. 5.16. Number of video sessions per hour seen by the server handling *video1* in the preferred data center of the EU1-ADSL dataset. A breakdown of sessions based on whether flows are directed to preferred data centers is also shown.

videos that are downloaded at least once from a non-preferred data center are considered. The results show two trends. First, a large fraction of videos are downloaded exactly once from the non-preferred data center. For example, for the EU1-Campus dataset, around 85% of the videos are downloaded only once from the non-preferred data center. Second, there is a long tail in the distributions. In fact, some videos are downloaded more than 1000 times from non-preferred data centers. We consider the impact of popular and unpopular videos on server selection in the next few paragraphs.

• **Alleviating hot-spots due to popular videos:** Let us focus first on the tail in Figure 5.13. Figure 5.14 considers the four videos with the highest number of accesses to the non-preferred data centers for the EU1-ADSL dataset. Each graph corresponds

to one of the videos, and shows (i) the total number of accesses to that video; and (ii) the number of times the video is downloaded from the non-preferred data center, as a function of time. We see that there are spikes indicating that some videos are more popular during certain limited periods of time. Most accesses to non-preferred data centers occur during these periods. In particular, all these videos were played by default when accessing the `www.youtube.com` web page for exactly 24 hours, i.e., they are the "video of the day".

Those are therefore very popular videos, which possibly generate a workload that can exceed the preferred data center capacity. Therefore, application-layer redirection is used to handle the peaks. As further evidence, Figure 5.15 shows the average and the maximum number of requests served by each server (identified by its IP address) in the preferred data center as a function of time. The figure shows that at several times, the maximum number of requests a single server has to handle is by far larger than the average load. For instance at time 115, the average load is about 50 video flows, but there is one server that answers more than 650 requests. Interestingly, we note that the servers suffering the peak loads are those serving the majority of the top videos of Figure 5.14.

Further investigation reveals that DNS correctly forwards the request to a server in the preferred data center, but since its load is too high, the server redirects part of the requests to another server in a non-preferred data center. Consider Figure 5.16, which shows the load in terms of sessions, handled by the server receiving the requests for *video1* for the EU1-ADSL dataset. Different colors are used to show the breakdown of the total number of sessions according to the preferred/non-preferred patterns. For example, we can see that in the first 6 days, the majority of the sessions involves only flows served by the preferred data center. On the last day however, a larger number of requests is received, which leads to an increase in application-layer redirections to a non-preferred data center. Overall, these results show that local and possibly persistent overload situations are handled by the YouTube CDN via application-layer redirection mechanisms.

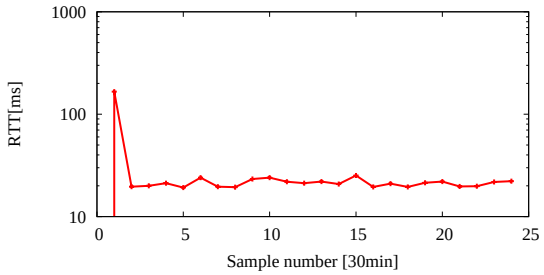


Fig. 5.17. Variation over time of the RTT between a PlanetLab node and the content servers for requests of the same test video.

• **Availability of unpopular videos:** Consider again Figure 5.13. Let us now focus on the observation that several videos are downloaded exactly once from the non-preferred data center. Further analysis indicated that for most datasets, over 99% of these videos were accessed exactly once in the entire dataset, with this access being to non-preferred data centers. However, when the videos were accessed more than once, only the first access was redirected to a non-preferred data center.

This observation leads us to hypothesize that downloads from non-preferred data centers can occur because of the limited popularity of the videos. In particular, videos that are rarely accessed may be unavailable at the preferred data center, causing the user requests to be redirected to non-preferred data centers until the video is found.

Since our datasets only contain a limited view of the accesses seen by a data center, it is difficult to validate this claim using only our datasets. We therefore conducted controlled active experiments using PlanetLab nodes. In particular, we uploaded a test video to YouTube. The video was then downloaded from 45 PlanetLab nodes around the world. Nodes were carefully selected so that most of them had different preferred data centers. From each node, we also measured the RTT to the server being used to download the content. We repeated this experiment every 30 minutes for 12 hours.

Figure 5.17 shows an example of the variation of RTT samples considering a PlanetLab node located in California. Observe that the very first sample has an RTT

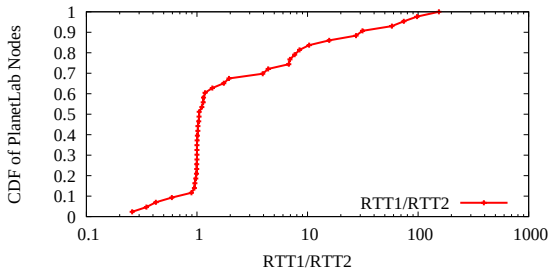


Fig. 5.18. Reduction in RTT from PlanetLab nodes to the content servers when a test video is downloaded twice. The first access may incur a higher RTT due to unavailability of content in the preferred data center.

of around 200 ms. In contrast, later samples exhibit RTT of about 20 ms. Further investigations showed that the first time, the video was served by a data center in the Netherlands while subsequent requests were served by a data center in California.

Figure 5.18 shows the CDF of the ratio of the RTT to the server that handled the first video request (RTT1) to the RTT to the server that handled the second video request (RTT2) for all the PlanetLab nodes. A ratio greater than 1 means that the video was obtained from a closer data center in the second attempt than in the first attempt. A ratio with a value close to 1 shows that the first request went to the same server or a server in the same data center as the second request. For over 40% of the PlanetLab nodes, the ratio was larger than 1, and in 20% of the cases the ratio was greater than 10. Interestingly, we have also found the RTT of subsequent samples is comparable to the RTT of the second sample. Overall, these results indicate that the first access to an unpopular video may indeed be directed to a non-preferred data center, but subsequent accesses are typically handled from the preferred data center.

5.8 Impact of Redirections on User Performance

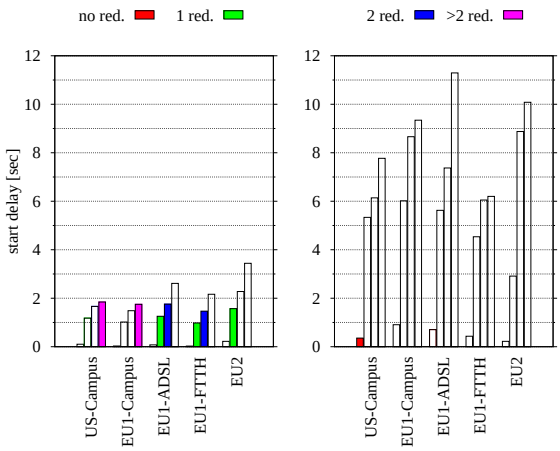
In this section, we evaluate the impact that the YouTube infrastructure decisions may have on user performance. We evaluate performance considering two metrics:

- Startup delay (SD): this metric captures how long the user have to wait before the video is ready to be played. Ideally, we would like to measure this as the time from when the user selects the video of interest until the time the video starts playing in the video web page. However, we cannot get this information from our flow level records. Instead, we approximate this metric by measuring the difference between the time when the client initiates the TCP connection to the first content server for a video, to the time when the first video packet from the content server is received by the client.

- Ratio of download rate to playback rate (RDP): this metric captures the cases in which clients suffer stalls while watching a video. Ideally, we would like to measure this by detecting the times in which the video actually stalls (possibly because the playback buffer is empty). Instead, we approximate this metric by taking the ratio of the average TCP connection download rate to the video playback rate, which is the declared rate for the video to be played without interruption. We can obtain the video playback rate from the header of Flash videos. We consider videos with RDP<1 as those where users have suffered stalls.

5.8.1 Analyzing the Startup Delay

As described in Section 5.6, a video request can be redirected from an initial content server to a different one for various reasons such as server load or content unavailability. In these cases, the SD corresponds to the time between the first SYN of the first request of the client and the first data packet of the video coming from the final server. Since several redirections can occur, the clients are affected by higher delays than when they are immediately served by the first server. Figure 5.19 reports the 50th and 90th of the SD distribution considering different number of redirections. There are five sets of bars, one for each trace, and four bars in each set indicating a number of redirections for a groups of sessions. As we can see, each redirection step



(a) 50th percentile (b) 95th percentile

Fig. 5.19. Start delays with redirection.

increases the SD. We observe that when there are no redirections, the SD is typically small, with less than 100 milliseconds for 50% of the cases in all traces. However, with more redirections, the SD can increase to more than 10 seconds in some cases. We dug deeper and found that the increase in SD comes mainly from: (i) an increase in the number of DNS resolutions, since every server that causes a redirection provides a new content server name that needs to be resolved; (ii) an increase in the time that the final content server contacted takes to start delivering the video to clients.

A natural question is how often redirections occur, since if they happen frequently, clients may prefer not to use the system. In Table 5.4, we report on the fraction of sessions with respect to the number of redirections. We note that although having sessions with two or more redirection is not common, one redirection sessions are not negligible. Hence clients may still suffer from high SD in 6-15% of the cases.

5.8.2 Analyzing the Ratio of Download Rate to Playback Rate

At a high level, videos could be slowly downloaded and stall because of various reasons such as problems at the client side or problems at the server side or problems

Table 5.4
Fraction of Sessions with Redirections

Trace	No red	1 red	2 red	>2 red
US-Campus	91.0	7.7	1.1	0.1
EU1-Campus	83.8	14.1	1.9	0.1
EU1-ADSL	85.9	12.5	1.5	0.1
EU1-FTTH	83.6	14.3	2.0	0.1
EU2	92.2	6.4	1.2	0.2

in the network. In this section we first show how predominant is having video stalls, i.e. video flows with RDP less than one, in our traces. Then, we present a possible reason for this problem and finally we show how users react to slow or interrupted videos. For convenience, we call slow flows, those video flows with RDP less than one.

Figure 5.20 shows the fraction of slow video flows from the total flows for all the traces. We can observe that EU2 has the largest number of slow flows with 16% while US-Campus has the lowest at 2%. Also, note that there is a correlation between the access bandwidth of clients and the fraction of slow flows. EU2 and EU1-ADSL are both networks with ADSL users and show a larger fraction of slow flows than US-Campus and EU1-FTTH where clients have much higher access bandwidth. EU1-Campus, as opposed to US-Campus, shows a relatively high fraction of slow flow. We found that this mainly happens because at peak hours, the NAT box at the edge of a dorm network inside the Campus network, gets congested and causes most of the YouTube flows to become slow.

A possible explanation for the existence of slow flows is that it is for flows to non-preferred data centers. Since non-preferred data centers are farther from the networks we monitor, it is possible that the extra network delay may cause the download rate to be less than the playback rate. Figure 5.21 shows the fraction of slow flows from the total. There are five sets of bars, one for each trace. Each set has one bar for preferred and one for non-preferred flows. We can see that in general, a larger fraction

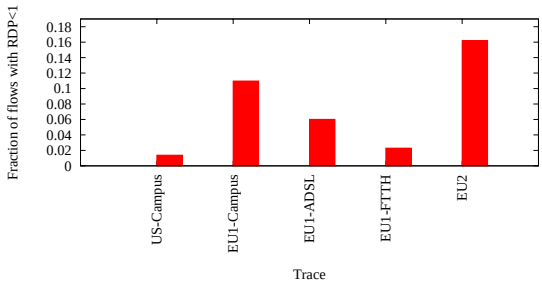


Fig. 5.20. Fraction of flows with slow download rate

of the slow flows are non-preferred, but the difference is not prominent. For instance, in EU1-Campus, there is a 4% difference between preferred and non-preferred slow flows. In EU2, the fraction of slow flows to the preferred data center is slightly higher. This may be because of network congestion or server load at the cache internal to EU2.

Finally, we investigate if there is a change in user behavior because of slow flows. In particular, we compare the fraction from the full video that users download for slow flows and for flows that do not suffer any stalling. Figure 5.22 present our results. There are five sets of bars, one for each trace. There are two bars in each set, one for the fraction of the full video downloaded in slow flows and one the fraction of the full video downloaded in normal videos (e.g. videos with $download\_ratio \geq 1$). We can observe that in general, clients that experience good performance download a larger fraction of the video than clients with bad performance. For instance, on average, clients experiencing good performance download more than 60% of the video. On the other, for slow flows, the fraction of the videos downloaded can be as low as 30%.

5.9 Discussion

While we have performed an extensive study on the various mechanisms used by YouTube for the selection of content servers for a client, we recognize that there are still a few issues that require more experimentation. We discuss two issues in this

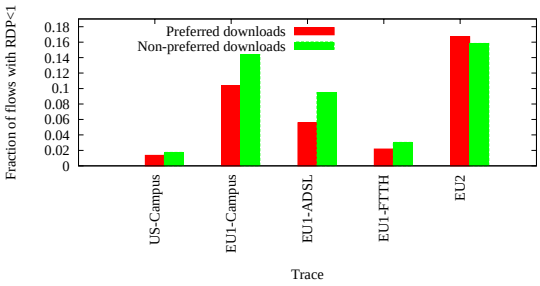


Fig. 5.21. Fraction of flows with slow download rate, for preferred and non-preferred data centers

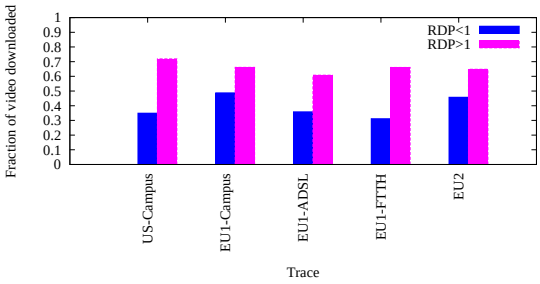


Fig. 5.22. Average fraction of video download for slow flows and not-slow flows.

section. First, we found that, in general, the closest data center in RTT is also the data center that provides most of the videos to clients in the networks monitored. However, a trace collected more recently showed that for US-Campus most videos were coming from a farther away data center. Hence, we believe that in some cases, RTT may not be the only factor considered in server selection and more complex mechanisms may be in place. Second, in our active experiments with planetlab, we found that a single request for a rare video from a client, would cause a video replication to the closest data center to the client. We believe that while our observations may hold in many cases, this caching policy seems simplistic and requires further experimentation to confirm it.

5.10 Related Work

The attention of the research community on YouTube has grown in the last few years. We can coarsely group works in two categories:

• **YouTube Infrastructure Studies:** Recently, a few works analyzed the YouTube video delivery infrastructure ([8, 9]). Both works focus on the "old" YouTube infrastructure. In [8], the authors collected traces at the backbone of a large ISP. Using DNS name resolution of servers, they discovered eight data centers around the U.S. that provided most videos to clients around the world. Further, they found that the YouTube server selection algorithm does not consider geographical location of clients and that requests are directed to data centers proportionally to the data center size. In contrast, our work focuses on the "new" YouTube infrastructure; we have evidence that requests are now redirected to servers in a preferred data center particular to each network and that RTT between data centers and clients plays a role in the server selection strategy. In [9] the authors perform PlanetLab experiments to download YouTube videos and measure user performance. The authors found that most videos are being sent from a few locations in the U.S. and that YouTube pushes popular videos to more data centers. In contrast, in our work, we study traces from several large ISP and campus networks in two continents, which capture actual user behavior; we found that most videos are being delivered from a preferred data center, typically close to the client and that, while popularity of videos may play a role on the redirection of clients to non-preferred data centers, it is not a prominent reason for it. Finally, we also differ from [8, 9] in that we analyze key factors that affect server selection in the YouTube CDN. More recently, a concurrent and preliminary work [122] has started analyzing the new YouTube infrastructure. Our work clearly differs in various aspects. In particular, we use more systematic state-of-the-art algorithms for server geolocation; we also rely on a trace-based analysis instead of active PlanetLab experiments and finally we dig deeper into identifying the various causes underlying content server redirection.

• **YouTube Videos Characterization:** Several works have focused on characterizing various aspects of videos existing in YouTube as well as usage patterns. [10] and [11] collected traces at the edge of a single campus network and characterized per video statistics such as popularity, duration, file size and playback bitrate, as well as usage pattern statistics such as day versus night accesses and volume of traffic seen from the Campus. [12] and [115] crawled the YouTube site for an extended period of time and performed video popularity and user behavior analysis. Further, [12] compares YouTube to other video providers such as Netflix and [115] investigates social networking in YouTube videos. We differ from all these works since we study the video distribution infrastructure. In particular we focus on understanding the content server selection mechanisms used by YouTube. In addition, we analyze datasets from five distinct vantage points ranging from campus networks to nationwide ISPs.

5.11 Conclusion

In this paper we have obtained a deeper understanding into the factors impacting how YouTube video requests are served by data centers. Our understanding has been based on week-long datasets collected from the edge of five networks including two university campuses and two national ISPs, located in three different countries. Our analysis indicates that the YouTube infrastructure has been completely redesigned compared to the one previously analyzed in the literature. In the new design, most YouTube requests are directed to a preferred data center and the RTT between users and data centers plays a role in the video server selection process. More surprisingly, however, our analysis also indicates a significant number of instances (at least 10% in all our datasets) where videos are served from non-preferred data centers. We identified a variety of causes underlying accesses to non-preferred data centers including: (i) load balancing; (ii) variations across DNS servers within a network; (iii) alleviation of load hot spots due to popular video content; and (iv) availability of unpopular

video content in a given data center. Overall these results point to the complexity of factors that govern server selection in the YouTube CDN.

6. CONCLUSIONS AND FUTURE WORK

In this chapter, we present a summary of our key contributions and future work.

6.1 Summary of Contributions

In this section, we summarize the key contribution of this thesis.

6.1.1 Detecting and Preventing Undesirable Behavior of P2P Clients

• *Characterization of undesirable behavior of P2P clients:* Our work is one of the first to show that undesirable behavior exists and is prevalent in real networks. Our analysis shows several examples of undesirable behavior including evidence of DDoS attacks exploiting live P2P clients, significant amounts of unwanted traffic that may harm network performance, and instances where the performance of participating peers may be subverted due to maliciously deployed servers. We systematically study traces collected from a PoP within a nationwide ISP with more than 5 million users. This is a very unique network, where 70% of the inbound traffic and 95% of the outbound traffic are due to P2P data. We found it is hard to distinguish undesirable behavior from normal behavior in an automatic fashion due to the heterogeneity of the P2P traffic and the difficulty in characterizing normal behavior of clients. Hence, we develop mechanisms to detect undesirable behavior, that combine data mining techniques and manual inspection through domain knowledge.

• *Design and evaluation of mechanisms to prevent P2P systems from being exploited to create DDoS attacks:* We study in detail a type of undesirable behavior which we found with our detection methodology, P2P clients participating on DDoS

attacks. Further, we propose and extensively evaluate active-probing based membership validation mechanisms to make P2P systems robust to these types of attacks.

6.1.2 Implications of Localizing P2P Traffic through a Characterization of Private P2P Systems

We present the first measurement study of communities of P2P clients that are localized to a network. We study two such systems used for content sharing, one hosted in a large nationwide ISP and the other in a campus network. In addition, we show the performance benefits experienced by clients of these systems and present a study of the effect of these systems in the traffic volumes carried by links in the host network. We draw lessons from our characterization study that apply to recent research on localization of P2P traffic to within ISP boundaries. In particular, our results indicate that (i) in ISPs with heterogeneous access technologies, the performance benefits to users on localizing P2P traffic is largely dependent on the degree of altruism of peers behind high-bandwidth access technologies; and (ii) while localization can reduce the traffic on Internet peering links, it has the potential to cause a significant increase in traffic on internal links of providers, potentially requiring upgrades of network links.

In a joint parallel effort [27], we study the Internet-wide impact of P2P traffic localization on ISP profitability. Our contributions include a methodology to perform what-if analysis on the adoption of P2P localization by all ISPs in the Internet or by a limited number of ISPs. Some of our key findings include: (i) residential ISPs can actually lose money when localization is employed; (ii) the reduction in costs due to localization will be limited for ISPs with small P2P populations; and (iii) some ISPs can better increase profitability through alternate strategies to localization by taking advantage of the business relationships they have with other ISPs.

6.1.3 Dissecting Video Server Selection Strategies in the YouTube CDN

Our analysis indicates that the YouTube infrastructure has been completely redesigned compared to the one previously analyzed in the literature. In the new design, most YouTube requests are directed to a preferred data center and the RTT between users and data centers plays a role in the video server selection process. More surprisingly, however, our analysis also indicates a significant number of instances where users are served from a data center that is not the preferred one. In one of our datasets, up to 55% of video requests were not served from the preferred data center, while in most datasets at least 10% of requests were not served from the preferred data center.

We identified a variety of causes underlying accesses to non-preferred data centers. In some cases DNS mechanisms resulted in users being directed to non-preferred data centers for reasons including load balancing to handle variations in system load due to the day/night patterns in YouTube usage. Interestingly, we found other cases where video was streamed to the user from a non-preferred data center, even though DNS directs the user to the preferred data center. The common causes underlying such cases included (i) alleviation of load hot spots due to popular video content; and (ii) accesses to unpopular video content that may not be available in a given data center.

6.2 Future Work

Motivated by our current research on the study of video distribution networks, our future work is related to two relevant areas. The first one involves studying video distribution networks and their interactions with mobile devices and other digital media player devices. The second area of research is on performing what-if analysis on changes that may occur in video distribution networks.

In this section, we elaborate more on concrete ideas for future work.

6.2.1 Video Content Distribution in Mobile Devices

We want to study video distribution networks and their interactions with mobile devices and other digital media player devices. The popularity of new ways to access Internet content is creating new opportunities for content providers but also possibly some new problems. In particular, we enhance our YouTube analysis to distinguish between PC and mobile accesses. Our comparison study will focus on two main areas, the mechanisms used by YouTube to deliver video to PC and mobile clients and the behavior of users behind PCs and mobile devices. A preliminary analysis on our YouTube traces show that the way video is delivered to mobile devices is different than PCs. In particular, while PCs obtain the video in a single TCP connection, mobile devices request the video in chunks, which are carried in separate TCP connections. This may have some implications in the network. For example, the more TCP connections open, the more state that need to be kept at middle-boxes such as NATs and firewalls. A second preliminary observation shows that mobile clients mostly watch videos encoded in an MPEG-4 format, as opposed to Flash videos downloaded by PC users. It will be interesting to investigate if this difference has any implications on the location of the caches that deliver the videos to mobile devices. Finally, more related to user behavior, we want to investigate interesting questions such as: (i) what fraction of the the video do mobile users watch? Conventional wisdom would say that mobile users will watch a smaller fraction of the video because of the limited screen size and device resources; (ii) Do mobile users interact with the video player for fast-forwarding or backward seeks? (iii) What are the implications of these type of behavior on the network?

6.2.2 Modeling "What-if" Scenarios of Changes in Video Distribution Systems

We want to perform what-if analysis on changes that may occur at the service provider level and the impact those changes may have on residential ISPs and users

performance. We may consider changes such as: (i) What if the videos distributed by the service provider (i.e. YouTube) increase in quality (high definition quality) or in length (full movies instead of short videos)? We can think on two possible impacts to this change. First, since videos will be bigger in size, there may be less opportunity of caching them at the closest location to a client, which may hurt user performance. Second, in the case of high definition videos, since the playback rate is higher, the corresponding download rate must also be higher so that the user does not experience any interruptions. This may cause congestion in the network (i.e. the client's access network or even the provider's network); (ii) What if YouTube changes from a distributed setting with tens of data centers spread around the world, to a more centralized setting where only a few data centers exist? or to the other extreme of an even more distributed setting with hundreds or thousands of data centers? In the former case, clients performance may suffer because the video must be downloaded from farther locations and residential ISPs may be affected with more transit traffic costs. In the latter case, the service provider may incur higher infrastructure maintenance costs. (iii) As shown in Section 5.8, users watch 50% to 70% of videos. What if YouTube changes its caching scheme to leverage this fact, where the CDN only caches a fraction of the video in the closest data center to the user and only provides other chunks of the video if they are requested by the client? This could possibly improve user performance, since more videos will be cached at a close location, and reduce service provider costs since less inter data center traffic may be required. The various what-if analysis proposed in this section, will require building a simulation infrastructure composed of various models such as the provider data center locations, the popularity of videos requested, the client population and content interest per ISP and the inter autonomous system connectivity among others. Our work presented in Section 5 and our parallel joint work [27], can be key stepping stones in building this simulation infrastructure.

LIST OF REFERENCES

LIST OF REFERENCES

[1] Skype, "Global P2P Telephony Company." http://www.skype.com.

[2] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing skype traffic: when randomness plays with you," in *SIGCOMM*, 2007.

[3] BitTorrent. http://www.bittorrent.org.

[4] "YouTube." `http://www.youtube.com`.

[5] "Hulu." `http://www.hulu.com`.

[6] "Netflix." `http://www.netflix.com`.

[7] A. TV. http://www.apple.com/appletv/.

[8] V. K. Adhikari, S. Jain, and Z.-L. Zhang, "YouTube Traffic Dynamics and Its Interplay With a Tier-1 ISP: an ISP Perspective," in *IMC '10: Proceedings of the 10th ACM SIGCOMM Internet Measurement Conference*, (New York, NY, USA), pp. 431–443, ACM, 2010.

[9] M. Saxena, U. Sharan, and S. Fahmy, "Analyzing Video Services in Web 2.0: a Global Perspective," in *NOSSDAV '08: Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, (New York, NY, USA), pp. 39–44, ACM, 2008.

[10] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "YouTube Traffic Characterization: A View From The Edge," in *IMC '07: Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference*, (New York, NY, USA), pp. 15–28, ACM, 2007.

[11] M. Zink, K. Suh, Y. Gu, and J. Kurose, "Characteristics of YouTube Network Traffic at a Campus Network - Measurements, Models, and Implications," *Computer Networks*, vol. 53, no. 4, pp. 501–514, 2009.

[12] M. Cha, H. Kwak, P. Rodriguez, Y.-Y. Ahn, and S. Moon, "I Tube, You Tube, Everybody Tubes: Analyzing The World's Largest User Generated Content Video System," in *IMC '07: Proceedings of the 7th ACM SIGCOMM Internet Measurement Conference*, (New York, NY, USA), pp. 1–14, ACM, 2007.

[13] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz, "P4P: Provider Portal for Applications," in *SIGCOMM*, 2008.

[14] D. R. Choffnes and F. E. Bustamante, "Taming the Torrent: a Practical Approach to Reducing Cross-ISP Traffic in Peer-to-Peer Systems.," in *SIGCOMM*, 2008.

[15] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should Internet Service Providers Fear Peer-Assisted Content Distribution?," in *IMC*, 2005.

[16] G. Shen, Y. Wang, Y. Xiong, B. Zhao, and Z. Zhang, "HPTP: Relieving the Tension between ISPs and P2P," in *IPTPS*, 2007.

[17] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPs and P2P Users Cooperate for Improved Performance?," *SIGCOMM Comput. Commun. Rev.*, 2007.

[18] R. Bindal, P. Cao, and W. Chan, "Improving Traffic Locality in BitTorrent via Biased Neighbor Selection," in *ICDCS*, 2006.

[19] B. Liu, Y. Cui, Y. Lu, and Y. Xue, "Locality-Awareness in BitTorrent-Like P2P Applications," *IEEE/ACM Trans. Multimedia*, 2009.

[20] M. S. J. Seedorf, S. Kiesel, "Traffic Localization for P2P-Applications: The ALTO Approach," in *IEEE P2P*, September 2009.

[21] "ETSI TISPAN - Defining the Next Generation Network."

[22] "Is the Skype outage really a big deal?." http://news.cnet.com/8301-10784_3-9761673-7.html.

[23] S. Bellovin, "Security Aspects of Napster and Gnutella." Invited Talk at USENIX Annual Technical Conference, 2001.

[24] E. Athanasopoulos, K.G.Anagnostakis, and E. Markatos, "Misusing Unstructured P2P Systems to Perform DoS Attacks: The Network That Never Forgets," in *ACNS*, 2006.

[25] N. Naoumov and K. Ross, "Exploiting P2P systems for DDoS attacks," in *International Workshop on Peer-to-Peer Information Management*, 2006.

[26] K. Defrawy, M. Gjoka, and A. Markopoulou, ""BotTorrent: Misusing BitTorrent to launch DDoS attacks"," in *SRUTI*, 2007.

[27] J. Seibert, R. Torres, M. Mellia, M. Munafo, C. Nita-Rotaru, and S. Rao, "The internet-wide impact of p2p traffic localization on isp profitability," in *Under submission*, 2011.

[28] I. Sandvine, "2009 global broadband phenomena." `http://www.sandvine.com/news/global_broadband_trends.asp`, 2009.

[29] Cisco, "Cisco visual networking index: Usage," October 2010.

[30] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet Inter-domain Traffic," in *Proceedings of the ACM SIGCOMM 2010 Conference*, (New York, NY, USA), pp. 75–86, ACM, 2010.

[31] K. Cho, K. Fukuda, and H. Esaki, "The Impact and Implications of the Growth in Residential User-to-User Traffic," in *SIGCOMM*, 2006.

[32] eMule. http://www.emule-project.net.

[33] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions," in *SIGCOMM*, 2005.

[34] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *IMW*, 2002.

[35] B. Krishnamurthy, S. Sen, Y. Zhang, and Y. Chen, "Sketch-based change detection: Methods, evaluation, and applications," in *IMC*, 2003.

[36] DC++. http://dcplusplus.sourceforge.net/.

[37] CISCO, "Cisco IOS NetFlow," `http://www.cisco.com/web/go/netflow`.

[38] F. Constantinou and P. Mavrommatis, "Identifying Known and Unknown Peer-to-Peer Traffic," in *NCA*, 2006.

[39] T. Karagiannis, D. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel Traffic Classification in the Dark," in *SIGCOMM*, 2006.

[40] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese, "Network Monitoring using Traffic Dispersion Graphs (TDGs)," in *IMC*, 2007.

[41] M. Collins and M. Reiter, "Finding Peer-To-Peer File-sharing Using Coarse Network Behaviors," in *ESORICS*, 2006.

[42] L. Bernaille, R. Teixeira, and K. Salamatian, "Early Application Identification," in *CONEXT*, 2006.

[43] T. Karagiannis, A. Broido, M. Faloutsos, and K. Claffy, "Transport Layer Identification of P2P traffic," in *IMC*, 2004.

[44] A. Moore and D. Zuev, "Internet Traffic Classification Using Bayesian Analysis Techniques," in *SIGMETRICS*, 2005.

[45] M. Mellia, R. L. Cigno, and F. Neri, "Measuring IP and TCP behavior on edge nodes with Tstat," *Computer Networks*, vol. 47, no. 1, pp. 1–21, 2005.

[46] R. Birke, M. Mellia, M. Petracca, and D. Rossi, "Understanding VoIP from backbone measurements," in *INFOCOM*, 2007.

[47] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "Simple Traversal of User Datagram Protocol Through Network Address Translation STUN," 2003. RFC-3489.

[48] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the XOR metric," in *IPTPS*, 2002.

[49] Adunanza. http://www.adunanza.net/.

[50] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to Data Mining*. Pearson, 2006.

[51] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *KDD-96*, 1996.

[52] M. Zhou, Y. Dai, and X. Li, "A Measurement Study of the Structured Overlay Network in P2P File-Sharing Applications," in *ISM*, 2006.

[53] "eMule forum: Repeated Kad Errors." http://forum.emule-project.net/index.php?showtopic=133799.

[54] "eMule forum :: Fake Server List And Ip Numbers." http://forum.emule-project.net/index.php?showtopic=120066.

[55] A. Banerjee, M. Faloutsos, and L. Bhuyan, "The P2P war: Someone is monitoring your activities," in *ScienceDirect*, pp. 1272–1280, 2008.

[56] Prolexic. http://www.prolexic.com/content/moduleId/tPjJLKRF/ article/aRQNVcBH.html.

[57] IPP2P. http://www.ipp2p.org.

[58] Y. Zhang, Z. Ge, A. Greenberg, and M. Roughan, "Network anomography," in *IMC*, 2005.

[59] Phoenix Labs, "PeerGuardian," http://phoenixlabs.org/pg2/.

[60] "I-BlockList," http://www.IBlocklist.com.

[61] "Windows peer-to-peer networking." http://www.microsoft.com/p2p.

[62] S. Ali, A. Mathur, and H. Zhang, "Measurement of commercial peer-to-peer live video streaming," in *WRAIPS*, 2006.

[63] X. Hei, C. Liang, J. Liang, Y. Liu, and K. Ross, "Insights into pplive: A measurement study of a large-scale p2p iptv system," in *Workshop on Internet Protocol TV services over World Wide Web in conjunction with WWW2006*, 2006.

[64] E. Sit and R.Morris, "Security Considerations for Peer-to-Peer Distributed Hash Tables," in *Proc. IPTPS*, March 2002.

[65] J. Douceur, "The Sybil Attack," in *Proc. IPTPS*, March 2002.

[66] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D.S.Wallach, "Security for structured peer-to-peer overlay networks," in *Proc. OSDI*, December 2002.

[67] A. Singh, T.-W. Ngan, D. P, and D. Wallach, "Eclipse Attacks on Overlays: Threats and Defenses," in *Proc. INFOCOM*, 2006.

[68] D. Wallach, "A Survey of Peer-to-Peer Security Issues," in *International Symposium on Software Security*, November 2002.

[69] K. C. Sia, "DDoS Vulnerability Analysis of BitTorrent Protocol." Technical Report, UCLA, 2007.

[70] X. Sun, R. Torres and S. Rao, "DDoS Attacks by Subverting Membership Management in P2P Systems," in *Proc. NPSec*, 2007.

[71] J. Yu, Z. Li, and X. Chen, "Misusing Kademlia Protocol to Perform DDoS Attacks," in *Proc. of the International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, 2008.

[72] V. Paxson, "An analysis of using reflectors for distributed denial-of-service attacks, Computer Communication Review 31(3), 2001."

[73] D. D. J.Mirkovic, S. Dietrich and P. Reiher, "Internet Denial of Service: Attack and Defense Mechanisms," 2005.

[74] Y. Chu, A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early Experience with an Internet Broadcast System Based on Overlay Multicast," in *Proc. USENIX*, June 2004.

[75] J. Harrington, C. Kuwanoe, and C. Zou, "A BitTorrent-Driven Distributed Denial-of-Service Attack," in *Proc. of 3rd International Conference of Security and Privacy in Communication Networks(SecureComm)*, 2007.

[76] "eDonkey." http://en.wikipedia.org/wiki/EDonkey2000.

[77] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in *Proceedings of ACM SIGCOMM*, 2001.

[78] D. Malkhi, Y. Mansour, and M. Reiter, "Diffusing without false rumors: On propagating updates in a byzantine environment," *Theoretical Computer Science*, vol. 299, no. (1-3), pp. 289–306, 2003.

[79] D. Malkhi, E. Pavlov, and Y. Sella, "Optimal unconditional information diffusion," in *15th International Symposium on DIStributed Computing (DISC 2001)*, 2001.

[80] D. Malkhi, M. Reiter, O. Rodeh, and Y. Sella, "Efficient update diffusion in byzantine environments," in *20th Symposium on Reliable Distributed Systems (SRDS 2001)*, 2001.

[81] Y. Minsky and F. Schneider, "Tolerating malicious gossip," *Distributed Computing*, vol. 16, no. 1, pp. 49–68, 2003.

[82] X. Sun, R. Torres, and S. Rao, "On the Feasibility of Exploiting P2P Systems to Launch DDoS Attacks," in *Journal of Peer-to-Peer Networking and Applications*, Springer New York, 2009.

[83] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The eigentrust algorithm for reputation management in p2p networks," in *WWW*, 2003.

[84] K. Hoffman, D. Zage, and C. Nita-Rotaru, "A Survey of Attack and Defense Techniques for Reputation Systems," *ACM Computing Survey*, March 2008.

[85] "Route views project." http://www.routeviews.org.

[86] B. Krishnamurthy and J. Wang, "On network-aware clustering of web clients," *SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 4, pp. 97–110, 2000.

[87] S. Guha, P. Francis, "Characterization and Measurement of TCP Traversal through NATs and Firewalls," in *Proc. ACM SIGCOMM IMC*, 2005.

[88] B. Ford, P. Srisuresh, D. Kegel, "Peer-to-peer communication across network address translators," in *Proc. USENIX*, 2005.

[89] A. Klemm, C. Lindemann, M. K. Vernon, and O. P. Waldhorst, "Characterizing the query behavior in peer-to-peer file sharing systems," in *Proc. ACM SIGCOMM IMC*, 2004.

[90] D. Stutzbach and R. Rejaie, "Understanding churn in peer-to-peer networks," in *Proc. ACM SIGCOMM IMC*, 2006.

[91] M. J. Freedman, K. Lakshminarayanan, S. Rhea, and I. Stoica, "Non-transitive connectivity and dhts," in *Proc. WORLDS*, 2005.

[92] Y. Liu, X. Liu, C. Wang, and L. Xiao, "Defending P2Ps from Overlay Flooding-based DDoS," in *Proc. of the International Conference on Parallel Processing*, 2007.

[93] V. Lam, S. Antonatos, P. Akritidis, and K. G. Anagnostakis, ""Puppetnets: Misusing web browsers as a distributed attack infrastructure"," in *CCS*, 2006.

[94] "DNS amplification attacks." http://www.isotf.org/news/DNS-Amplification-Attacks.pdf.

[95] J. Yu, Z. Li, H. Chen, and X. Chen, "A Detection and Offense Mechanism to Defend Against Application Layer DDoS Attacks," in *Proc. of the Third International Conference on Networking and Services (ICNS)*, 2007.

[96] K. Gummadi, R. Dunn, S. Saroiu, S. Gribble, H. Levy, and J. Zahorjan, "Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload," in *SOSP*, 2003.

[97] S. Saroiu, K. Gummadi, and S. Gribble, "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts," in *Multimedia System*, 2003.

[98] L. Guo, S. Chen, Z. Xiao, E. Tan, X. Ding, and X. Zhang, "Measurements, Analysis, and Modeling of BitTorrent-like Systems," in *IMC*, 2005.

[99] S. Sen and J. Wang, "Analyzing Peer-to-Peer Traffic Across Large Networks," in *IMW*, 2002.

[100] "ADC protocol." http://adc.sourceforge.net/ADC.html.

[101] Y. Zhang, M. Roughan, N. Duffield, and A. Greenberg, "Fast Accurate Computation of Large-Scale IP Traffic Matrices from Link Loads," in *SIGMETRICS*, 2003.

[102] D. Alderson, H. Chang, M. Roughan, S. Uhlig, and W. Willinger, "The Many Facets of the Internet Topology and Traffic," *Ametican Institute for Mathematical Sciences*, vol. 1, no. 4, pp. 569–600, 2006.

[103] M. Pietrzyk, J.-L. Costeux, G. Urvoy-Keller, and T. En-Najjary, "Challenging statistical classification for operational usage: the adsl case," in *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, (New York, NY, USA), pp. 122–135, ACM, 2009.

[104] M. Mellia, M. Meo, L. Muscariello, and D. Rossi, "Passive Analysis of TCP Anomalies," *Comput. Netw.*, 2008.

[105] M. Mellia, "Tstat home page," `http://www.tstat.polito.it`, 2009.

[106] BitTorrent. http://www.bittorrent.org.

[107] R. Cuevas, N. Laoutaris, X. Yang, G. Siganos, and P. Rodriguez, "Deep diving into bittorrent locality," *CoRR*, vol. abs/0907.3874, 2009.

[108] M. Piatek, H. Madhyastha, J. John, A. Krishnamurthy, and T. Anderson, "Pitfalls for isp-friendly p2p design," in *HotNets-VIII*, 2009.

[109] P. Biddle, P. England, M. Peinado, and B. Willman, "The Darknet and the Future of Content Distribution," in *DRM*, 2002.

[110] M. Rogers and S. Bhatti, "How to Disappear Completely: A Survey of Private Peer-to-Peer Networks," in *SPACE*, 2007.

[111] J. Bethencourt, W. Low, I. Simmons, and M. Williamson, "Establishing Darknet Connections: An Evaluation of Usability and Security," in *SOUPS*, 2007.

[112] C. Zhang, P. Dhungel, D. Wu, Z. Liu, and K. Ross, "BitTorrent Darknets," in *INFOCOM*, 2010.

[113] R. Torres, M. Hajjat, M. Mellia, M. Munafo, and S. Rao, "Inferring Undesirable Behavior from P2P Traffic Analysis," in *SIGMETRICS*, 2009.

[114] "YouTube Fact Sheet." `http://www.youtube.com/t/fact_sheet`.

[115] X. Cheng, C. Dale, and J. Liu, "Statistics and Social Network of YouTube Videos," in *IWQoS*, pp. 229–238, 2008.

[116] "Maxmind GeoLite City Database." `http://www.maxmind.com/app/geolitecity`.

[117] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida, "Constraint-based Geolocation of Internet Hosts," *IEEE/ACM Transactions on Networking*, vol. 14, no. 6, pp. 1219–1232, 2006.

[118] M. Mellia, R. L. Cigno, and F. Neri, "Measuring IP and TCP Behavior on Edge Nodes with Tstat," *Computer Networks*, vol. 47, no. 1, pp. 1–21, 2005.

[119] M. Pietrzyk, J.-L. Costeux, G. Urvoy-Keller, and T. En-Najjary, "Challenging Statistical Classification for Operational Usage: the ADSL Case," in *IMC '09: Proceedings of the 9th ACM SIGCOMM Internet Measurement Conference*, (New York, NY, USA), pp. 122–135, ACM, 2009.

[120] "Tstat home page." `http://tstat.polito.it`.

[121] R. Percacci and A. Vespignani, "Scale-free behavior of the internet global performance," *Eur. Phys. J. B*, vol. 32, no. 4, pp. 411–414, 2003.

[122] V. K. Adhikari, S. Jain, G. Ranjan, and Z.-L. Zhang, "Understanding datacenter driven content distribution," in *ACM Conext Student Workshop 2010*, (New York, NY, USA), ACM, 2010.

VITA

VITA

Ruben Torres was born in Panama City, Panama. He received the B.S. Degree in Electronics and Communications Engineering from the University of Panama, in 2002, and the M.S. and Ph.D. degrees in Electrical and Computer Engineering from Purdue University in 2006 and 2011 respectively.

From January 2002 to July 2004, he worked as a technology consultant for Centaury Technologies Corporation in Panama and as a researcher for the Technological University of Panama. Since summer 2005, he has worked in several projects in the area of Internet content distribution and published his results in prestigious conferences such as ACM Sigmetrics and IEEE Infocom. His research interests are in experimental computer networks, distributed systems and security, with special focus on systems design and implementation, network measurements and performance evaluation.