THESIS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

Characterization and Classification of Internet Backbone Traffic

WOLFGANG JOHN

Division of Networks and Systems
Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2010

Characterization and Classification of Internet Backbone Traffic
Wolfgang John
ISBN 978-91-7385-363-7

Doktorsavhandlingar vid Chalmers tekniska högskola
Ny serie 3044
ISSN 0346-718X

Technical Report 65D
Department of Computer Science and Engineering
Research Group: Computer Communications and Computer Networks

Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 GÖTEBORG, Sweden
Phone: +46 (0)31-772 10 00

Contact Information:
Wolfgang John
Division of Networks and Systems
Department of Computer Science and Engineering
Chalmers University of Technology
SE-412 96 GÖTEBORG, Sweden

Email: `wolfgang.john@chalmers.se`

Printed by Chalmers Reproservice
Göteborg, Sweden, 2010

Characterization and Classification of Internet Backbone Traffic

Wolfgang John

Department of Computer Science and Engineering, Chalmers University of Technology

ABSTRACT

We contribute to an improved understanding of Internet traffic characteristics by measuring and analyzing modern Internet backbone data. We start the thesis with an overview of several important considerations for passive Internet traffic collection on large-scale network links. The lessons learned from a successful measurement project on academic Internet backbone links can serve as guidelines to others setting up and performing similar measurements. The data from these measurements are the basis for the analyses made in this thesis. As a first result we present a detailed characterization of packet headers, which reveals protocol-specific features and provides a systematic survey of packet header anomalies. The packet-level analysis is followed by a characterization on the flow-level, where packets are correlated according to their communication endpoints. We propose a method and accompanying metrics to assess routing symmetry on a flow-level based on passive measurements. This method will help to improve traffic analysis techniques. We used the method on our data, and the results suggest that routing symmetry is uncommon on non-edge Internet links. We then confirm the predominance of TCP as the transport protocol in backbone traffic. However, we observe an increase of UDP traffic during the last few years, which we attribute to P2P signaling traffic. We also analyze further flow characteristics such as connection establishment and termination behavior, which reveals differences among traffic from various classes of applications. These results show that there is a need to make a more detailed analysis, i.e., classification of traffic according to network application. To accomplish this, we review state-of-the-art traffic classification approaches and subsequently propose two new methods. The first method provides a payload-independent classification of aggregated traffic based on connection patterns. This provides a rough traffic decomposition in a privacy sensitive way. Second, we present a classification method for fine-grained protocol identification by utilizing statistical packet and flow features. Preliminary results indicate that this method is capable of accurate classification in a simple and efficient way. We conclude the thesis by discussing limitations in current Internet measurement research. Considering the role of the Internet as a critical infrastructure of global importance, a detailed understanding of Internet traffic is essential. This thesis presents methods and results contributing additional perspectives on global Internet characteristics at different levels of granularity.

Keywords: Internet, Traffic, Backbone, Passive, Measurement, Characterization, Classification

Acknowledgements

First and foremost, I want to express my gratitude to my supervisor, Prof. Sven Tafvelin, who offered me this challenging position and was always a kind and patient adviser. I am also indebted to my co-supervisor, Ass. Prof. Tomas Olovsson, for many encouraging discussions regarding research and education. For putting my progress into perspective and repeatedly lifting my spirits I want to thank my mentors, Ana Bove and Johan Sehlstedt.

A great and sincere thanks goes to kc claffy, the "elfbin crowd" and the rest of the CAIDA folks at UCSD. I am still amazed by the great hospitality and encouragement I felt during my visits to San Diego. Specifically, I want to thank kc for continuously supporting me with valuable advice. I also want to thank Min Zhang, Maurizio Dusi and Erik Hjelmvik, with whom I was fortunate enough to co-author papers. I really enjoyed the possibility to cooperate and exchange ideas with such inspiring researchers.

At Chalmers, I had the great fortune to work with great colleagues. I want to thank Prof. Erland Jonsson and his security group, i.e., Magnus Almgren, Vilhelm Verendel, Pierre Kleberger and Farnaz Moradi, for granting me asylum in their regular lunch crew. For creating a great work environment, I am grateful to my friends and former colleagues Daniel Andersson, Magnus Själander, Martin Thuresson, and Ulf Larson. I also want to thank all other current and former PhD students and personnel at the Department for being helpful and supportive every single day, making my work at Chalmers a real pleasure. I am especially thankful for the continuous support of the kind and helpful administrative staff.

I would like to thank my family in Austria for their support and understanding during my years abroad. Furthermore, I feel very privileged to have fantastic friends both in Sweden and back home in Austria. Great thanks go to the people who make my life in Sweden so enjoyable and who have the patience and understanding to remember me even after long periods without contact. While it is impossible to provide an exhaustive list here, I want to thank Manfred Eckschlager, Bernd Resch, and Peter Romirer-Maierhofer for sharing their opinions about research related issues with me.

Finally, I want to thank SUNET, the Swedish University Network, for sponsoring my research. Specifically, I want to thank Börje Josefsson, Per Nihlen and his team at NUNOC for reliable assistance with both operational and technical issues during the measurements.

THANKS TO ALL OF YOU!

Wolfgang John

Göteborg, February 2010

List of Appended Papers

I: **Wolfgang John**, Sven Tafvelin and Tomas Olovsson, "Passive Internet Measurement: Overview and Guidelines based on Experiences" in *Computer Communications*, Vol. 33(5), Elsevier 2010.

II: **Wolfgang John** and Sven Tafvelin, "Analysis of Internet Backbone Traffic and Anomalies Observed" in *IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, San Diego, California, USA, 2007.

III: **Wolfgang John** and Tomas Olovsson, "Detection of Malicious Traffic on Backbone Links via Packet Header Analysis" in *Campus-Wide Information Systems*, Vol. 25(5), Emerald 2008.

IV: **Wolfgang John**, Maurizio Dusi and kc claffy, "Estimating Routing Symmetry on Single Links by Passive Flow Measurements" , Submitted to Conference, Chalmers University 2009.

V: **Wolfgang John**, Min Zhang and Maurizio Dusi, "Analysis of UDP Traffic Usage on Internet Backbone Links", Extended Report of a Short Paper Published as:
Min Zhang, Maurizio Dusi, Wolfgang John and Changjia Chen, "Analysis of UDP Traffic Usage on Internet Backbone Links" in *SAINT '09: Proceedings of the 9th Annual International Symposium on Applications and the Internet*, Seattle, USA, 2009.

VI: **Wolfgang John** and Sven Tafvelin, "Differences between In- and Outbound Internet Backbone Traffic" at *TNC '07: TERENA Networking Conference*, Copenhagen, Denmark, 2007.

VII: **Wolfgang John**, Sven Tafvelin and Tomas Olovsson, "Trends and Differences in Connection Behavior within Classes of Internet Backbone Traffic" in *PAM '08: Proceedings of the 9th Passive and Active Measurement Conference*, Cleveland, Ohio, USA, 2008.

VIII: Min Zhang, **Wolfgang John**, kc claffy and Nevil Brownlee, "State of the Art in Traffic Classification: A Research Overview" at *PAM '09: the 10th Passive and Active Measurement Conference*, Seoul, Korea, 2009.

IX: **Wolfgang John** and Sven Tafvelin, “Heuristics to Classify Internet Backbone Traffic based on Connection Patterns” in *ICOIN ’08: Proceedings of the 22th International Conference on Information Networking*, Busan, Korea, 2008.

X: Erik Hjelmvik and **Wolfgang John**, “Statistical Protocol IDentification with SPID: Preliminary Results” at *SNCNW ’09: Swedish National Computer Networking Workshop*, Uppsala, Sweden, 2009.

Contents

Part I

INTRODUCTION SUMMARY

1 Introduction

Today, the Internet has emerged as the key component in personal and commercial communication. One contributing factor to the ongoing expansion of the Internet is its versatility and flexibility. In fact, almost any electronic device can be connected to the Internet these days, ranging from traditional desktop computers, servers and supercomputers to all kinds of wireless devices, embedded systems, sensors and even home equipment. Accordingly, the usage of the Internet has changed dramatically since its initial operation in the early 1980s, when it was a research project connecting a handful of computers, facilitating a small set of remote operations. Today (2010), the Internet serves as the data backbone for all kinds of protocols, making it possible to interact and exchange not only text, but also voice, audio, video, and various other forms of digital media between hundreds of millions of nodes.

Traditionally, an illustration of the protocol layers of the Internet pictures an hourglass, with a single Internet Protocol (IP) on the central network layer and an increasingly broader spectrum of protocols above and below. Since the introduction of IP in 1981, a protocol that is basically still unchanged, technology and protocols have developed significantly. Underlying transmission media evolved from copper to fiber optics and wireless technologies, routers and switches became more intelligent, and are now able to handle Gbit/s instead of Kbit/s, and additional middleware devices have been introduced (e.g., Network Address Translation boxes and firewalls). Above the network layer, new applications have also constantly been added, ranging from basic services such as the Domain Name System (DNS) and HyperText Transfer Protocol (HTTP), to recent complex peer-to-peer (P2P) protocols allowing applications for file sharing, video streaming, and IP telephony. With the introduction of IPv6, even the foundation of the Internet, IP, is finally about to be substituted.

This multiplicity of protocols and technologies leads to a continuous increase in the complexity of the Internet as a whole. Of course, individual protocols and network infrastructures are usually well understood and tested in isolated lab environments or simulations. However, their behavior as observed while interacting with the vast diversity of applications and technologies in the Internet environment is often unclear, especially on a global scale.

This lack of understanding is further amplified by the fact that the topology of the Internet was not planned in advance. The current Internet topology is the result of an uncoordinated extension process, where heterogeneous networks of independent organizations have been connected one by one to the main Internet (*INTERconnected NETworks*). As a consequence, the Internet today is built up of independent, autonomous network systems, where each autonomous system (AS) has its own set of usage and pricing policies, quality of service (QoS) measures and resulting traffic mix. Thus, the usage of Internet protocols and applications is not only changing over time but also with geographical locations [1].

Finally, higher connectivity bandwidths, growing numbers of users and increasing economical importance of the Internet also lead to an increase in misuse and anomalous behavior [2]. Not only do the numbers of malicious incidents continue to rise, but also the level of sophistication of attack methods and available tools. Today, automated attack tools employ advanced attack patterns and react on the deployment of firewalls and intrusion detection systems by cleverly obfuscating their malicious actions. Malicious activities range from host- and port-scanning to more sophisticated attack types, such as worms and various denial of service attacks. Unfortunately, the Internet, initially meant to be a friendly place, eventually became a very hostile environment that needs to be studied continuously in order to develop suitable counter strategies.

For the reasons mentioned above, network researchers and engineers currently have limited understanding of the modern Internet, despite its emergence as a critical infrastructure of global importance [3]. We identified a number of important open questions that Internet measurements help to answer. We grouped them into four rough categories:

(i) *Scalability and sustainability* issues regarding fundamental Internet services, including routing scalability, AS level topology evolution, IP address space utilization, DNS scalability and security;

(ii) *Internet performance*, e.g., the impact of new protocols and applications on Internet performance characteristics such as per-flow throughput, jitter, latency and packet loss/reordering;

(iii) *Evolution of Internet traffic*, such as traffic growth trends, protocol and application mix at different times and different locations;

(iv) *Network security*, including anomaly detection and mitigation of network attacks and other unwanted/unsolicited traffic, such as email spam, botnet and scanning traffic.

Given the possibility to collect Internet traffic data on a wide-area network backbone link, this thesis addresses the latter two categories. We also discuss methodological aspects of passive Internet measurement and data collection, which form the basis for our results. Specifically, the thesis sets out to provide a better understanding of the modern Internet

by presenting current characteristics of Internet traffic based on a large amount of empirical data. We claim that it is crucial for the Internet community to understand the nature and detailed behavior of modern network traffic. A deeper understanding would support optimization and development of network protocols and devices, and further improve the security of network applications and the protection of Internet users.

1.1 Organization

The thesis is based on a collection of published papers, where reprints of the papers are appended in Part II. Part I provides the introductory summary. In Chapter 1 we present the main topics of the thesis by introducing traffic characterization and classification. We discuss Internet measurement in general, define the thesis objectives and discuss limitations of the obtained results. Chapter 2 describes the MonNet project, which provided the collection framework for most of the network data analyzed in this thesis. This chapter includes a description of the measurement location, the technical solution for data collection, the analysis procedures and a summary of the resulting datasets. Chapter 3 provides an overview of related measurement projects and lists publicly available datasets and data sharing approaches. Chapter 4 groups the papers listed in Part II into a logical structure and provides a short summary of each paper. In Chapter 5 we list the main thesis contributions and summarize our findings. Chapter 6 concludes the thesis with general lessons learned and a discussion of the status quo of the research field and its continuing struggle with issues regarding data sharing and access.

1.2 Analysis of Internet Data

Before discussing issues surrounding passive collection of Internet data traces, we want to introduce the main topic of this thesis: a general characterization of backbone traces on the packet and flow-levels. We continue with a discussion of traffic classification methods, which can complement traffic characterization efforts by providing insight into the type of traffic analyzed.

1.2.1 Traffic Characterization

We define *traffic characterization* as the analysis of Internet data resulting in a description of traffic properties. These properties can range from the features of aggregate network traffic (e.g., flow size distribution [4]) to detailed features of single packets and flows [5]. Specifically, traffic characterization in this thesis covers a detailed, fine grained traffic analysis of packet and flow-level data. Since the Internet is a moving and continually evolving

target [6], some of our results revise or update previous studies that are based on outdated data sets collected years ago. Most results in this thesis are based on contemporary data from a previously unstudied measurement location on the Internet and hence contribute to a global picture of current Internet traffic characteristics.

Our packet-level characterization reveals general traffic properties such as packet size distribution and transport protocol breakdown and also shows the current deployment of protocol-specific features such as IP and TCP options and flags (*Paper II*), which is relevant input to Internet simulation models [7]. Our packet analysis furthermore includes a systematic listing of packet header anomalies together with their frequencies as seen "in the wild" on the observed Internet backbone links (*Paper III*), which provides an empirical background for the development and refinement of traffic filters, firewalls and intrusion detection systems. Furthermore, we believe that knowledge of such detailed Internet traffic characteristics can help researchers and practitioners in designing networked devices and applications and in improving their performance and robustness.

Flow-level analysis aggregates individual packets into flows, which can provide additional insights into traffic characteristics. We propose a method and accompanying metrics to assess routing symmetry flow measurements from a specific link, and the results suggest that routing symmetry is uncommon on non-edge Internet links. We then confirm the predominance of TCP as transport protocol in backbone traffic, but note an increase of UDP traffic during the last few years. These results verify common assumptions about Internet traffic, which are often embedded into traffic analysis or classification tools [8–10]. Consequently, the results can impact advanced Internet analysis efforts and provide further measurement support for Internet modeling. We also provide a detailed analysis of TCP flows to reveal network properties such as connection lifetime, size and establishment/termination behavior (*Paper VI*). The results of this flow analysis highlight the need for *traffic classification* according to application as a next step towards a better understanding of Internet traffic behavior. The following analysis of classified traffic reveals trends and differences in connection properties of Internet traffic and shows how different classes are behaving "in the wild" (*Paper VII*). These results enable the Internet community to see how current transport protocols are utilized by application developers, facilitating the improved design of network devices, software, and protocols.

1.2.2 Traffic Classification

We define *traffic classification* as the analysis of Internet data resulting in a decomposition of the traffic according to network applications/application layer protocols or classes thereof (e.g., bulk, interactive, WWW, etc. [11]). In other words, the goal of traffic classification

is to understand the type of traffic carried on Internet links [12–14]. Traffic classification results can be useful for traffic management purposes (such as QoS and traffic shaping mechanisms [15]) and traffic engineering purposes (such as optimization of network design and resource provisioning). Furthermore, understanding the type of traffic carried on networks supports security monitoring by facilitating the detection of illicit traffic, such as network attacks and other security violations. Modern firewalls, NAT boxes, and Intrusion Detection Systems (IDSs) need to be able to reliably classify network protocols in order to implement fine grained and secure access policies. Apart from the apparent interest of operators and researchers in understanding trends and changes in network usage, there have also been a number of political and legal discussions about Internet usage, further highlighting the need for accurate traffic classification methods. These political discussions include the ongoing debate between intellectual property representatives[1] and the P2P file sharing community[2] [16, 17]. There are also network neutrality discussions between Internet Service Providers (ISPs) and content providers[3] [15, 18, 19].

Historically, network applications have been designed to use well-known port numbers to communicate with servers or peers, making traffic classification relatively straightforward. However, in the early 2000s, developers of upcoming file sharing applications (e.g., KaZaA [20]) started to deviate from the standard behavior by using dynamic port numbers, thus diminishing the accuracy of port-based classification [11, 21, 22]. Since then, there has been an ongoing arms race between application developers trying to avoid traffic filtering or classification, and operators, network researchers, and other institutions interested in accurate traffic classification (*Paper VIII*). Researchers first used static payload examination to classify applications using unpredictable ports [11, 22–24], an approach also used in commercial tools [25, 26]. Application developers then reacted by using proprietary protocols and payload encryption, which means that modern traffic classification methods cannot rely solely on port number information and static payload signatures [8, 24, 27]. In *Paper IX* we propose a payload independent classification method for aggregated Internet backbone data. *Paper X* presents an alternative classification method utilizing statistical flow and payload features.

1.3 Internet Measurement

Before we could perform offline analysis of Internet traffic, we had to set up a measurement infrastructure to collect Internet data. Packet-level data collection on large-scale net-

[1] For example the Recording Industry Association of America (RIAA).

[2] For example *The Pirate Bay.*

[3] These include commercial companies such Google and Yahoo, as well as the P2P file sharing community.

work links, however, is a non-trivial task (*Paper I*). One reason for the difficulties is the rapid and decentralized development of the Internet on a competitive market, which historically has left both little time and few resources to integrate measurement and analysis possibilities into Internet infrastructure, applications and protocols. Traditional network management therefore relies on aggregated measurements from individual nodes, such as SNMP Statistics (Simple Network Management Protocol [28]) and statistics of sampled flow data [29, 30] (e.g., flow counts and flow throughputs, size and duration distributions). However, we claim that we in addition need complete, fine grained data in order to obtain a comprehensive and detailed understanding of the modern Internet. Empirically measured Internet datasets constitute an important data source for different purposes:

- *Scientific purpose*: Analysis of actual Internet traffic provides much needed input for scientific simulation and modeling [7]. Ongoing measurements will also reveal longitudinal trends and changes in the usage of network applications and protocols, and thus foster improvement and development of network protocols and services. Finally, security measures should ideally be based on a profound understanding of traffic properties and should rely on fast and reliable methods to detect unwanted traffic and network anomalies. We therefore consider modern, real-life datasets vital for the network research and development community in order to be able to react to changes in traffic properties and behavior (for both benign and malicious reasons) in a timely fashion.
- *Operational purpose*: While traditional network management tools based on SNMP or Netflow [29] mainly cover critical operational requirements for ISPs, such as troubleshooting and provisioning, more advanced traffic engineering tasks (such as QoS measures and traffic shaping) often rely on classification tools and techniques based on packet-level data [15]. Not only the development, but also the validation of these techniques requires modern traffic traces collected by measurement infrastructures. Internet measurements can also be the basis for refinement of network design and provisioning, design of robust protocols and infrastructure, and improvement of network performance and accounting. Furthermore, Internet measurements reflecting network behavior as seen "in the wild" support security measures, such as refinement of rule sets for traffic filters, firewalls, and network intrusion detection systems [31].
- *Legal purpose*: Monitoring and measurement of Internet traffic are also of increasing legal relevance, as manifested in the recently ratified data retention directive of the European Union [32], requiring communication providers to retain connection data[4]

[4]Connection data here include type, source, destination and timing information for communication including Internet access, web and mail activities.

for periods of up to two years with the purpose of enabling network forensics. Implementations of these types of regulations can directly benefit from the achievements of the Internet measurement community, offering experiences in the non-trivial task of efficient collection and analysis of large amounts of traffic. However, such privacy sensitive regulations also evoke discussions about their ethical implications [33, 34].

1.3.1 Internet Measurement Approaches

We categorize network measurement approaches based on different dimensions, as discussed in *Paper I*. In the following paragraphs we briefly outline the five most important axes and, at the same time, highlight the approaches taken to collect that data for this thesis:

- *Active vs. passive measurement approaches*: Active measurement involves injecting traffic into the network to probe certain network devices (e.g., ping) or to measure network properties such as Round Trip Times (RTT), one-way delay and maximum bandwidth. Passive measurement or monitoring based on pure observation of network traffic is non-intrusive and does not change the existing traffic. Network traffic is tapped at a specific location and can then be recorded and processed at different levels of granularity, from complete packet-level traces to only a statistical summary. In this thesis we apply a passive measurement approach to provide analysis of Internet backbone traffic properties.
- *Software-based vs. hardware-based measurement*: Passive measurement tools based on software modify operating systems and device drivers on network hosts to obtain copies of network packets (e.g., BSD packet filter [35]). In contrast, hardware-based methods are designed specifically for collecting and processing network traffic on high-speed links such as an Internet backbone. Custom-built hardware collects traffic directly on the physical links[5] (e.g., by using optical splitters) or on network interfaces (e.g., mirrored router ports). Specifically, for our measurements we used a hardware-based measurement infrastructure applying optical splitters and Endace DAG cards [36] capable of collecting unsampled, complete packet traces on links with transmission speeds of up to 10 Gbit/s.
- *Online vs. offline processing*: Online processing refers to immediate processing of network data in "real time", which is essential for applications such as traffic filters and intrusion detection systems. Offline processing, on the other hand, is performed on network data after it is stored on a data medium. Offline processing is not time critical

[5]Internet measurement can also be performed on wireless networks. However, wireless measurement techniques are beyond the scope of this thesis and will therefore not be discussed.

and offers the possibility to process, compare, and validate network traffic collected at different times or different locations. Furthermore, stored network data can be re-analyzed on the basis of different criteria. Because of these advantages, we chose offline processing for the packet and flow-level characterization presented in this thesis, since they include complex and time-consuming analysis.

- *Data granularity*: The coarsest granularity is provided by aggregate traffic summaries and statistics, such as packet counts or data volumes, typically provided by SNMP. Another common practice is to condense network data into *network flows*. A flow can be described as a sequence of packets exchanged between common end points, defined by certain fields within network and transport headers (e.g., TCP connections). Packet-level traces provide the finest level of granularity, which can include all information of each packet observed on a specific host or link. Thus, they implicitly include the information contained in less granular data. Since packet-level traces furthermore offer the best analysis and validation possibilities, we chose to use packet-level measurements as raw data.

- *Sampled vs. unsampled data collection*: Given the large amount of data on modern high-speed links, a method for reducing it is to sample data instead of recording all the data observed (i.e., unsampled data collection). Sampling can be done on a packet and flow-level. Basic sampling approaches include count or interval-based systematic sampling (i.e., in static intervals, e.g., recording every *N*th packet/flow), random sampling (i.e., in random intervals), and stratified sampling (i.e., sampling different subpopulations independently). More sophisticated packet sampling approaches have also been proposed, such as adaptive packet sampling [37]. Good overviews of sampling and filtering techniques for passive Internet measurement can be found in Zseby et al. [38] and Duffield [39].

  The packet traces used in this thesis were collected to enable a range of different analysis tasks. However, when packet-level traces are collected without having an exact analysis in mind, sampling is difficult to apply because it requires a very deliberate choice of the sampling strategy to make sure that no sampling bias is introduced. While there are successful sampling techniques for inference of many packet and flow statistics, some characteristics cannot be accurately estimated based on sampled data. As an example, Mai et al. [40] showed that current sampling techniques introduce fundamental bias when used for detection of network anomalies such as volume anomalies and port scans. In this thesis, we decided to take advantage of our capability to collect complete, unsampled packet-header traces.

1.4 Thesis Objectives

As pointed out in this chapter, there are various reasons why we need to understand the nature and detailed behavior of Internet traffic. The datasets used in this thesis (mainly collected within the MonNet project, introduced in Chapter 2) allow investigation of Internet traffic characteristics on Internet backbone links. The overall objective of this thesis is:

> *Improving the understanding of Internet traffic characteristics*
> *by measuring and analyzing modern Internet backbone data.*

More specifically, this thesis sets out to answer the following research questions:

Question 1: *What are the main challenges in passive data collection on large-scale network links and how can we manage these challenges?*

Question 2: *How can packet header traces be used to analyze and characterize traffic from large-scale networks?*

Here we concentrate on the following three subquestions:

a. What traffic characteristics and what types of inconsistent traffic behavior can be revealed through the analysis of packet header information?

b. Are common assumptions about transport protocol usage and traffic symmetry valid when actually investigated using empirical data?

c. Can differences between classes of network traffic be found by analyzing detailed connection properties of TCP flows?

Question 3: *Can we find effective methods for classifying modern network traffic based on network application in order to support traffic characterization efforts?*

1.5 Thesis Limitations

While this thesis contributes to a better understanding of Internet traffic characteristics, we want to note some constraints regarding the results that are caused by deficiencies in the raw data:

- The results provided are snapshots of Internet traffic from *single vantage points* during a limited time period. Where possible, we complemented the MonNet datasets with traces from other measurement locations (i.e., US traces from CAIDA), but the snapshot character remains. We acknowledge that modern Internet traffic is too heterogeneous to allow broad generalization of such results, which is a general limitation of the research discipline of traffic analysis. However, the results contribute some additional perspectives on global Internet characteristics, which is important for identifying trends and possible invariances when compared to other snapshots of different environments. Furthermore, most methods and lessons learned are valid and applicable to other Internet measurement and analysis projects, independent of the exact nature of the network measured.
- The *duration of the traffic traces* collected within the MonNet project does not exceed 20 minutes due to hardware limitations of the measurement cards. Since Internet traffic distributions are heavy-tailed and substantially composed of long-lived flows [4], we were not able to provide a conclusive study of classical flow characteristics such as flow duration and flow sizes. Instead, we deliberately decided to focus on detailed packet-level characterization and flow characteristics such as connection establishment and termination behavior.
- The backbone traces analyzed in this thesis are *anonymized* and *contain no payload*, i.e., IP addresses have been anonymized in a prefix-preserving fashion (CryptoPAn [41]) and packets do not contain payload beyond transport layer protocols. These de-sensitization tasks were required by an ethics committee to permit the MonNet research to proceed. While anonymized packet header data enables a number of interesting research tasks, they have some limitations regarding their utility. Anonymized IP addresses do not allow us to pinpoint and further investigate hosts with anomalous behavior. Such investigations could include active measurements (e.g., Operating Systems fingerprinting) or contacting site administrators (e.g., for manual inspection of misbehaving servers). Missing payload also limits a number of further analysis possibilities. Access to (full, or at least partial) packet payload would allow validation of non-payload-based traffic classification results on solid ground truth. It would also facilitate

very fine grained traffic classification and thus allow studies of isolated characteristics of single applications. Payload data could furthermore enable studies of user behavior to find better answers to questions such as “what are people using the Internet for?”, which could yield results relevant not only for network research but also for social and legal sciences. Other research topics benefiting from payload access include network security related issues, such as anomaly detection. As a specific example, some malicious activities are not visible through packet headers but are embedded in anomalous payloads, e.g., code injection attacks. Investigations of botnet traffic and email spam would also benefit from access to packet payload information, allowing researchers to study detailed behavior and characteristics of the traffic. Finally, full packet payload may provide ground truth for detection methods.

2

Background - The MonNet Project

This chapter provides a description of the MonNet project, a project for passive Internet traffic measurement conducted at Chalmers University of Technology. The goal of the project is to support Internet traffic analysis by providing empirical data, i.e., passive measurements of backbone links. Most results presented in this thesis are based on data collected within MonNet. After giving some project background, including a description of the measurement location, we present the technical solution of the MonNet measurement infrastructure by describing the measurement nodes and the processing platform (Section 2.3). Finally, we describe the pre-processing and analysis procedures of the resulting packet-level traces in Sections 2.4, 2.5 and 2.6. Further experiences and lessons learned from this successful project are discussed in *Paper I*.

2.1 Preparatory Tasks and Project Administration

In 2004, the Computer Communications and Networks group at Chalmers University proposed a project regarding measurements of Internet traffic on the Swedish University Network (SUNET) to the SUNET board. The board then required permission of the "central Swedish committee for vetting ethics of research involving humans" (*Etikprövningsnämnden, EPN*) in order to grant the project (which we in the reminder of this thesis refer to as MonNet project). The Swedish EPN is among other things responsible for vetting research that involves dealing with sensitive information about people or personal information, equivalent to institutional review boards (IRB) [42] at US universities and research institutions [43]. The EPN committee carries out ethical vetting in six regional boards, where one of these boards is responsible for the region of Göteborg. After two meetings and discussions about the de-sensitization process of the traces, the regional ethics committee finally permitted the MonNet measurements to take place under the conditions that user payload is removed and IP addresses are anonymized (e.g., with CryptoPAn [41]).

The measurement and processing nodes applied have been planned and designed to meet the anticipated requirements of packet-header measurements on Packet over SONET

(PoS) Optical Carrier level 192 (OC192) links, i.e., links with line rates of 10 Gbit/s. During the planning phase, previous related measurement projects, such as NLANR PMA's OC48MON [44] and Sprint's IPMON [45], provided valuable inspiration. The resulting technical solution will be described in detail in section 2.3.

Even if we listed these preparatory tasks very briefly, it is important to note that they required significant monetary costs and time expenses. The two types of costs are common obstacles (besides privacy concerns of operators and their users) for passive large-scale network measurements, as acknowledged by many members of the Internet measurement research community [46]. The pure monetary costs of the MonNet measurement equipment correspond to the employment costs for a researcher (e.g., a PhD student) over a substantial time period. The following timeline of the project will highlight the additional time investments required: The MonNet project was proposed to the SUNET board in summer 2004. After a waiting period for permission by the ethics committee, problems with delayed delivery of crucial equipment and unexpected early hardware failures, the measurement nodes were not in place and operational until fall 2005, more than one year after the project began. It took another six months to gain experience in conducting sound Internet measurement [47], when we finally could collect the first usable dataset in April 2006. After 2006, SUNET changed the topology and technology significantly, which rendered our previous measurement location inoperative. During 2007, SUNET focused on launching and trouble-shooting the new network. In this time, SUNET staff was busy with many urgent operational tasks. For the MonNet project this implied that the required operator support for new data collection was not available until the operation of the new network had stabilized. During summer 2008, SUNET had some time resources available again, so that the measurement equipment could be re-installated at a new location.

2.2 Description of the Networks Measured

2.2.1 GigaSUNET

The first measurement traces we analyzed were collected on the previous generation of the SUNET backbone network, called GigaSUNET [48]. GigaSUNET was officially in operation until January 2007, when it was replaced by the current generation, called OptoSUNET [49]. The GigaSUNET backbone consisted of four core rings joining together at a central Internet exchange point in Stockholm. Each ring used Cisco OC192 PoS technology over Dense Wavelength Division Multiplexing (DWDM) channels to interconnect all Points of Presence (POP), i.e., all university cities in Sweden. We illustrate the topology of the internal GigaSUNET backbone in Fig. 2.1. Core routers (illustrated by circles) are fur-

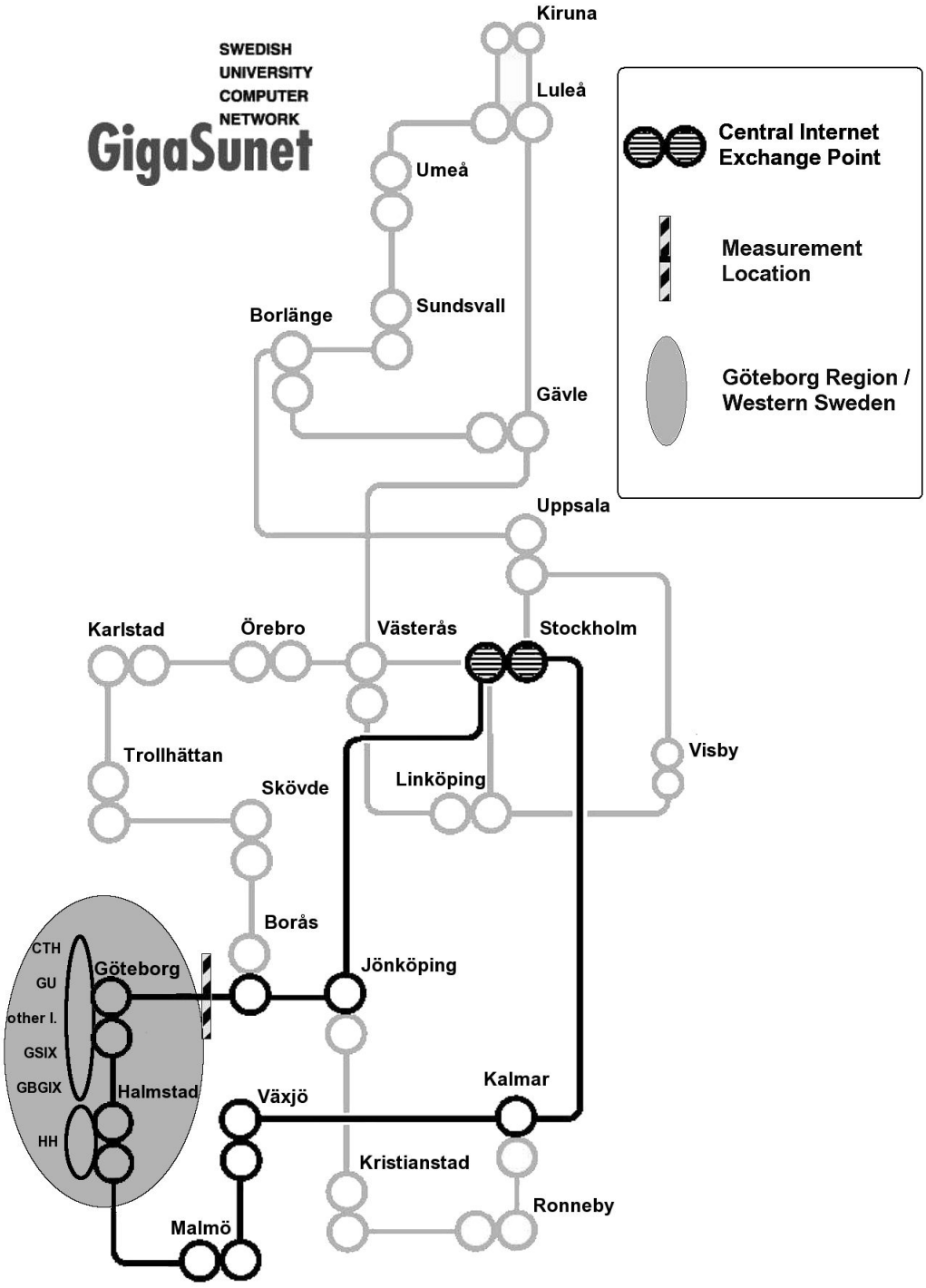


Figure 2.1: *Internal GigaSUNET topology with POPs displayed as two circles in order to indicate the two core routers connecting that POP with the ones in the neighboring cities. The network ring measured is colored in black. At the measurement point we collect data between the region of western Sweden (shaded in grey) and the main Internet outside Sweden (connected via a central Internet exchange point in Stockholm). We observed the traffic to and from Chalmers (CTH), Göteborg University (GU), Halmstad University (HH), smaller research Institutes (other I.), the student dormitory network (GSIX), and local ISPs via an Internet exchange point (GBGIX).*

thermore connected to an access network within the region, providing access to the SUNET backbone for regional SUNET customers such as universities and student networks. We illustrate the OC192 links connecting POPs as grey lines, with exception of the ring on which the measurements were performed, which we colored in black. We collected the traffic traces on the link between the cities of Göteborg and Borås, on the outermost part of the ring. This means that traffic passing the ring between the region of Göteborg (the grey shaded area) and the main Internet outside Sweden (peering with SUNET in Stockholm) was primarily routed via the tapped link, taking Borås as the next hop. SNMP statistics confirmed this behavior, showing that traffic amounts between Göteborg and Borås were an order of magnitude larger than the amounts of traffic transferred between Halmstad and Malmö [50].

We collected backbone traffic on the OC192 (10Gbit/s) link (i.e., one measurement card for each direction) at the measurement location between the core routers Göteborg and Borås. The link measured provides the Internet backbone for two major universities, Chalmers (CTH) [51] and Göteborg University (GU) [52], a substantial number of student dormitories (GSIX) [53] and a number of research institutes and smaller universities (other I.) such as Halmstad University (HH) [54]. Furthermore, around 14% of the collected traffic is exchange traffic with a local Internet exchange point in Göteborg (GBGIX) [55], providing peering between regional ISPs and SUNET. Thus, a significant part of the traffic is transit traffic. Summarized, the resulting traffic traces constitute a medium level of aggregation, between campus-wide traffic and tier-1 backbone traffic.

2.2.2 OptoSUNET

The ring architecture described above was during 2007 upgraded to OptoSUNET, a star structure over leased fiber. OptoSUNET connects all SUNET customers redundantly to a core network in Stockholm, as depicted in Fig. 2.2. While SUNET core routers are also directly connected to smaller Swedish ISPs generating some local exchange traffic, the traffic routed to the international commodity Internet is carried on two links between SUNET and NORDUnet, with capacities of 40Gbit/s and 10Gbit/s respectively. NORDUnet peers with Tier-1 backbone providers, large CDNs (Content Distribution Networks) and other academic networks. Since 40Gbit/s measurement equipment was economically unfeasible[1], we re-used the 10Gbit/s measurement infrastructure from GigaSUNET. We chose to collect on the 10Gbit/s link with the highest possible level of traffic aggregation: the 10Gbit/s link between SUNET and NORDUnet, indicated in black color in Fig. 2.2. According to SNMP statistics [56], the applied load-balancing mechanism assigned half of all inbound but only 15% of the outbound traffic volume to the 10 Gbit/s link observed, and the rest of the traffic

[1] A 40Gbit/s measurement infrastructure essentially requires measurement equipment for 4x10Gbit/s links.

to the 40 Gbit/s link. During July 2009 an additional 10 Gbit/s link in parallel with the existing one was installed in order to keep up with increasing traffic volumes. We illustrated this link as dashed line in the figure. Since this upgrade, the link observed carries about one third of all the inbound traffic but still about 15% of the outbound traffic volumes.

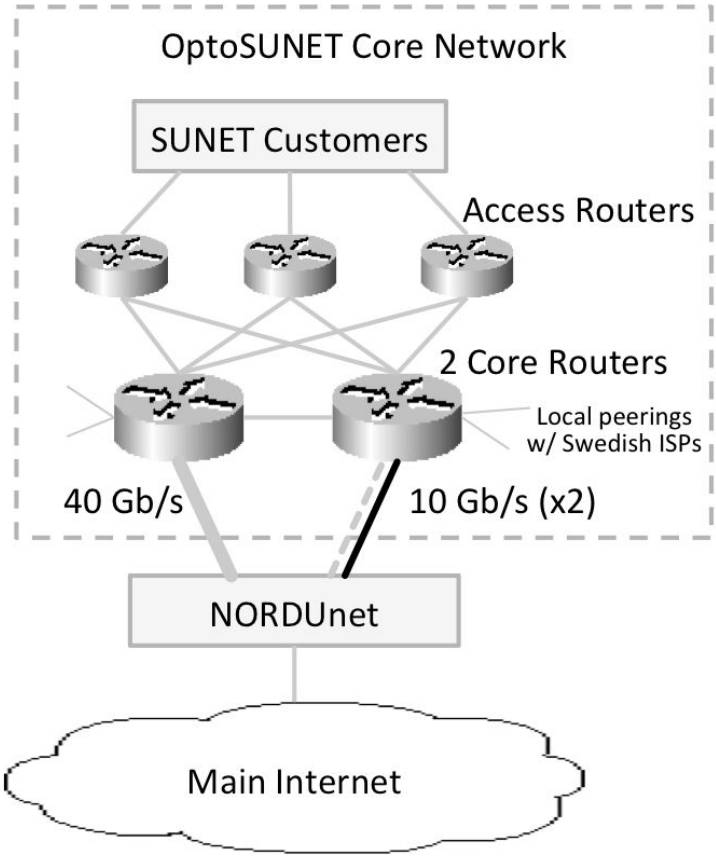


Figure 2.2: *OptoSUNET core topology. All SUNET customers are via access routers connected to two core routers. The SUNET core routers have local peering with Swedish ISPs, and are connected to the international commodity Internet via NORDUnet. SUNET is connected to NORDUnet via 3 links: a 40Gbit/s link and two 10Gbit/s links (one of them installed in July 2009). Our measurement equipment collects data on the first of the two 10Gbit/s links (black) between SUNET and NORDUnet.*

2.3 Technical Solution

In the following paragraphs, we briefly describe the hardware of the measurement and analysis infrastructure we used within the MonNet project. Section 8.3 of *Paper I* provides a detailed and more technical discussion of the hardware solution. Our hardware includes two measurement nodes and one additional processing platform. We use the latter as storage, analysis platform, and database for the network traces collected on SUNET. We ap-

ply optical splitters to tap the two OC192 links, one for each direction. With support from SUNET operators we installed the splitters and attached them to two measurement nodes on-site, which also pre-processed the traces (see Section 2.4). We always collected traces simultaneously for both directions. For the final analysis, we transferred the network traces to the processing platform at the Department of Computer Science and Engineering at Chalmers University.

The two measurement nodes are designed and configured identically, and are based on state-of-the-art hardware available during the design phase in 2004. Each optical splitter, tapping either the inbound or outbound OC192 link, is attached to an Endace DAG6.2SE card sitting in one of the measurement nodes. The cards are capable of collecting data on PoS and 10Gbit-Ethernet links with bandwidths of up to 10Gbit/s. We configured the cards to capture the first 120 bytes of each PoS frame to ensure that the entire network and transport header information is preserved. During pre-processing of the traces we then removed the remaining payload fractions for each packet.

After data collection and completion of the pre-processing procedures on the measurement nodes, we transferred the resulting traces to the storage and processing server located in the secured server room of the department. Besides storage of packet-level traces, the processing platform with external storage also houses a MYSQL database system, which we used for organizing the results obtained by the different analyses of the raw traces, as described in Section 2.6.

2.4 Trace Pre-processing

After storing the truncated data packets on the disks of the measurement nodes, we de-sensitized and sanitized the traces in offline fashion. Batch jobs carried out the de-sensitization and sanitization immediately after collection of the traces, in order to minimize the storage time of unprocessed and privacy sensitive data. We describe the pre-processing steps in detail in Section 8.4 of *Paper I*, but the following paragraphs provide a brief overview.

By trace de-sensitization we mean the removing of sensitive information to ensure privacy and confidentiality according to the requirements of the ethics committee. We did this by removing packet payload, which we define as all data following transport layer headers. As a next step, we anonymized IP addresses in the IPv4 headers based on the prefix-preserving CryptoPAN [41]. Throughout all MonNet measurements campaigns we used a single, unique encryption-key to allow us to track specific hosts and IP ranges between all measurements. Note however that CryptoPAN has its weaknesses and can be subject of de-anonymization attacks [57, 58], which we further discuss in Section 4.2.2 of *Paper I*.

Trace sanitization refers to the process of ensuring that the collected traces are free from logical inconsistencies and are suitable for further analysis [45, 47]. We applied sanity checks before and after each de-sensitization process. A detailed list of the sanity checks can be found in Section 8.3.4 of *Paper I*, where we differentiate between three types of errors: *Major errors; minor errors;* and *errors with unknown severity*.

In the common cases, when no inconsistencies or errors were detected, the original, unprocessed traces were deleted upon completion of the pre-processing procedures, and we kept only de-sensitized and sanitized versions of the traces. Sanity checks for *major errors* included examination of frame types, IP version numbers, DAG timestamps, and inspection of critical internal error counters and log files of the DAG card. While we did not encounter any cases of packet loss or truncation reported by the DAG cards, about 3% of the traces recorded showed garbled trace data or packet arrival rates of zero after a certain time. We suspect that these corrupt traces are the result of the DAG cards loosing framing due to a hardware failure. Over time, this problem got worse on one measurement node, resulting in about 20% corrupt traces on this particular host during the measurements in 2009. Upon detection of major errors, we discarded the specific trace and also deleted the corresponding trace in the opposite direction.

In the *minor error* category we included IP header checksum errors and frames discarded by the DAG due to receiver errors reported by the DAG cards. Receiver errors include link errors such as incorrect light levels on the fiber and HDLC checksum errors. Both errors are rather rare: in our traces (2006 and 2009), one out of 300 Million frames resulted in an IP header checksum error, and also receiver errors are rare (one out of 191 Million frames). Traces with minor errors have been kept and used for analysis, but missing packets and IP checksum errors have been documented in the attached meta-data files.

All remaining checks, such as parsing the system log during measurement intervals and inspecting other internal errors reported by the DAG cards, did not trigger any errors and are therefore classified as errors with *unknown severity*. However, if a novel error were to show up, the sanitization procedure would be interrupted, requesting manual inspection and categorization of the problem.

2.5 Resulting Datasets

We documented the collection process and the different pre-processing steps for each single trace. We stored the resulting meta-data in a file together with a checksum digest of the particular trace, in order to provide distinctive association in case the trace and its meta-data file get separated, e.g., when moving data. Meta-data includes a short description of the mea-

Location	Period Year-Month	Trace Count	Interval Minutes	Pkts 10^9	Data TB	Active IPs, Src to Dst (10^3) Outbound	Inbound	Used in *Paper #*
GigaSUNET	2006-04	74 (x2)	20	10.8	7.6	25 to 590	613 to 120	*II,IV-VII,IX*
	2006-09 to 2006-11	277 (x2)	10	27.9	19.5	24 to 620	635 to 80	*III-V,VII*
OptoSUNET	2008-12 to 2009-11	151 (x2)	10	33.0	19.5	6 to 800	2270 to 360	*IV,V*

Table 2.1: *Summary of MonNet traffic traces. For each measurement series, the table summarizes: number of traces collected (two unidirectional traces during each interval); measurement interval duration in minutes; sum of packets in billions; total amount of data carried by the packets in TB; average number of unique active IPs in thousands, i.e., sources sending packets to destinations during a measurement interval, listed for outbound and inbound direction; List of appended papers using traces from the specific series.*

surement location, direction of the link, timing information, status information of the DAG card and results of the three trace sanitization passes (before and after payload removal, and between payload removal and anonymization). The meta-data provides a summary about errors detected, which includes counts of occasionally observed receiver errors (HDLC CRC errors) and the exact positions of frames including IP header checksum errors.

At present (January 2010), the MonNet datasets represent 95 hours of backbone traffic, collected on 156 different days mainly during 2006 and 2009. As listed in Table 2.1, the data includes in total 72 billion IPv4 packets, carrying 47 TB of data. The table furthermore contains average numbers of unique IPs seen within each measurement series in order to provide a measure for the level of aggregation. The traces contain mainly IPv4 packets (99.98%). The remaining traffic consists of IPv6 BGP Multicast messages, CLNP routing updates (IS-IS) and Cisco Discovery Protocol (CDP) messages. Furthermore, on GigaSUNET we observed around 40 currently unidentified frames each minute. These frames seem to have random address and control bytes in their Cisco HDLC headers, with non-standard ethertypes of 0x4000 or 0x0000. The purpose of these frames is still unclear.

We recorded data traces at a GigaSUNET facility in Göteborg (i.e., the measurement location in Fig. 2.1) in two measurement series during 2006, as also documented in DatCat, the Internet Measurement Data Catalog [59]. We collected datasets in April (Spring dataset) and in the time from September to November 2006 (Fall dataset) on the measurement location on GigaSUNET. At each measurement, we simultaneously stored traces for both directions on the two measurement nodes. In Spring, we collected four traces of 20 minutes duration each day at identical times (2AM, 10AM, 2PM, 8PM) for a period of 20 days. We chose the times to cover business, non-business, and nighttime hours.

On the same location we collected GigaSUNET data at 277 randomized times during 80 days in Fall 2006. At each random time, we stored a trace of 10 minutes duration. We

here chose randomized times in order to provide a good statistical representation of Internet traffic characteristics at the specific time-period and location.

We collected data on OptoSUNET between December 2008 and November 2009 at the SUNET Network Control Center (NOC) in Stockholm, which is the physical location of the link highlighted in Fig. 2.2. During a number of smaller, scattered campaigns, summing up to 151 10-minute intervals, we collected traces at randomized times, again simultaneously for both directions. The data from OptoSUNET has been part of *Papers IV and V*, and is currently used as raw data for ongoing studies.

2.6 Analysis Approaches

So far, we only discussed the measurement process including data pre-processing. In this section, we outline the analysis approaches used to extract the scientific results we presented in the papers included. Following the development of our understanding leading to investigation of our observations in a bigger context, the three analysis methodologies presented are packet-level analysis, flow-level analysis, and traffic classification.

Packet-level Analysis

We ran different packet-level analysis programs on individual traces to extract statistical data into a database. A challenge in these analysis programs was to provide sufficient robustness, i.e., being able to deal with any possible kind of header inconsistency or anomaly. The resulting database consists of tables for specifically interesting features according to our research objectives, such as IP header length, IP packet length, TCP options and different kinds of anomalous behavior. In the database tables, data was summarized per direction and per measurement interval (i.e., trace time), which allowed us to analyze the data in different dimensions by issuing respective SQL queries. We summarized the results of the packet-level analysis in Section 4.2.1. The complete results are presented in *Papers II and III*, and to some extent in Sections 3 and 4 of *Paper VI*.

Flow-level Analysis

To conduct a detailed connection-level analysis, we merged the tightly synchronized unidirectional traces according to their timestamps. The ERF trace headers preserved the directional information in the resulting bidirectional traces. As a next step, our analysis program collected per-flow information of the packet-level traces. We summarized packet streams to flows by the use of a hash-table structure in memory. The gathered per-flow information includes packet and data counts for both directions, start- and end times, TCP flags and counters for erroneous packet headers and multiple occurrences of special flags like

RST or FIN. We inserted this information into one database table for each transport protocol (TCP and UDP), with each row representing a summary of one flow (or connection in the case of TCP).

We define a flow by the traditional 5-tuple of source/destination IP and port numbers as well as the transport protocol [45]. As transport protocols, we only considered TCP and UDP since they are together responsible for more than 99% of the data and the packets carried on SUNET links (see *Paper II*). TCP flows represent connections, and are therefore further separated by SYN, FIN and RST packets. Additional SYN segments for a specific tuple can sometimes be observed in the same direction within short time intervals, as often the case during scanning campaigns. In such cases, we opened further "connections" within the analysis program since each SYN qualifies as a new flow. We then add following non-SYN packets to the most recently opened connection of the particular tuple. Since UDP offers no connection establishment or termination, we define UDP flows as the sum of bidirectional packets observed between a specific 5-tuple during a specified time interval. For *Paper VI*, we specified this timeout as 20 minutes, i.e., the complete measurement duration of the traces in the Spring dataset. For *Papers VII and IX* we separated UDP flows by the commonly accepted timeout of 64 seconds [45, 60], which allows comparison of the results to related work [21]. We also used this bidirectional flow-level processing as ground-truth verification in *Paper IV*.

Traffic Classification

We classified our backbone data based on a set of heuristics regarding connection patterns of individual flows. A detailed description and verification of the heuristics can be found in *Paper IX*. We performed almost the entire the traffic classification by complex SQL statements within the database, starting with the flow tables resulting from the flow-level analysis. We applied the heuristics to the flow tables in 10-minute intervals, which means that every interval is analyzed in isolation, without memory of previous intervals. We first applied the 15 heuristics independently to all flows. For each flow, we set a bit-mask in a separate table in the database according to matching rules. This approach made it possible to verify each heuristic separately and to investigate the effects of different priority rankings of the heuristics. After empirical exploration of the most suitable prioritization scheme for the heuristics, we set an additional bit mask associated with each flow, indicating the final traffic classification into classes such as Web, P2P and attack traffic. The original flow tables together with the associated classification tables allow a convenient way to analyze and compare flow and connection characteristics among traffic of different network applications, which we did successfully for *Paper VII*.

3
Related Measurement Projects

Even though access to data from large-scale passive Internet measurements is still rare, the research community put significant amounts of effort into passive Internet measurement activities in recent years [46]. These activities include development of active and passive measurement methodologies and tools, targeting aspects such as network performance [61–63], traffic classification and quantification [13, 14, 64], reliability and security [65, 66]. In this section, we give an overview of measurement projects dealing with passive collection of Internet traces and packet-level analysis thereof, which is closely related to the topic of this thesis. This overview first presents the most prominent passive measurement projects, which have access to backbone measurement facilities and resulting packet-level traces. Second, we point out some smaller traffic analysis projects, some of which have access to their own packet traces, but many of them depending on shared and often slightly outdated datasets or flow-level data from cooperating service providers. The MonNet project, which we described in this thesis, provides new, contemporary data from a previously unmeasured network. The novelty of the data, the high aggregation level of the measured links, and the packet-level granularity of the traces contribute to a global picture of the current Internet.

NLANR PMA

The Passive Measurement and Analysis Project (PMA) [67] of the National Laboratory for Applied Network Research (NLANR) ended officially in 2006. The goal of NLANR PMA was to gain better understanding of the operation and behavior of the Internet by studying passive header traces. NLANR collected traces by daily measurements at different backbone and access network locations across the USA with speeds of up to OC48 (2.5Gbit/s). The measurements have been performed by specially designed nodes, the OC3MON and OC48MON systems [44], which have been based on Endace DAG4.2 cards [36]. The OC48MON system also influenced the design of the IPMON system of Sprint [45]. NLANR PMA made packet header traces publicly available, which lead to a number of analysis studies by other researchers based on NLANR PMA data. Jiang and Dovrolis used NLANR

traces to evaluate a passive measurement methodology that estimates the RTT distribution for TCP connections on a given network link [68]. Lan and Heidemann show that there are strong correlations between some combinations of size, rate and burstiness in "heavy-hitter" flows on NLANR PMA data [69, 70]. Pentikousis and Badr presented a comparative study of TCP option deployment, showing that the majority of senders employ the maximum segment size option, large windows do not accompany SACK deployment, and ECN usage is negligible on NLANR traces from 12 measurement locations [71].

CAIDA

The Cooperative Association for Internet Data Analysis (CAIDA) [72] was launched in 1997 and is based at the University of California (UCSD) on the San Diego Super Computer Center (SDSC). CAIDA sets out to provide tools and analyses in order to promote maintenance of a robust, scalable global Internet Infrastructure. The broad research activities include routing and addressing, topology, DNS, security, performance, visualization and traffic analysis. Researchers at CAIDA published a number of relevant studies of passive Internet measurement and traffic analysis [73]. These publications include the transport layer identification of P2P traffic in Karagiannis et al. [21, 74], analyses of passively collected Internet traffic in Fomenkov et al. [75] and McCreary and claffy [76], and the observations on fragmented traffic in Shannon et al. [77]. CAIDA also developed popular measurement tools, such as NeTraMet or CoralReef [78, 79], and founded the Internet measurement data catalog DatCat [80]. Furthermore, CAIDA shares datasets with the research community, such as security-related data traces from their network telescope and packet-level header traces from US peering points. Due to the high similarity in research focus, we carried out recent MonNet activities (including *Papers IV and V*) on data from both MonNet and CAIDA as joint studies with researchers situated at UCSD.

ITA

The Internet Traffic Archive (ITA) is hosted by the Lawrence Berkeley National Laboratory (LBNL) [81]. ITA was sponsored by ACM SIGCOMM to provide a moderated repository to support widespread access to Internet traffic traces, which are however rather outdated (+10 years). In addition to trace manipulation and analysis software (e.g., *tcpdpriv*, a trace anonymization program [82]), ITA includes LAN and WAN packet traces, flow records and HTTP logs collected at various sites between 1989 and 1998. These early packet traces have been used for seminal works on traffic characterization and modeling: Leland et al. [83] demonstrated in 1994 that Ethernet traffic is statistically self-similar and that none of the traditionally used traffic models (e.g., Poisson-related models and packet train models)

are able to capture this fractal-like behavior. Paxson and Floyd [84] then showed in 1994 that Poisson processes are valid only for modeling the arrival of user sessions, but that WAN packet arrival processes are better modeled using self-similar processes. Paxson in the same year furthermore found exponential growth trends in wide-area TCP connections from many applications with an explosive growth of then new protocols such as gopher and HTTP during 1992 and 1993 [85], and empirically derived analytic models to describe characteristics of wide-area TCP connections for TELNET, NNTP, SMTP and FTP [86].

LBNL/ICSI

The enterprise tracing project of the Lawrence Berkeley National Laboratory (LBNL) collected more than 100 hours of LBNL's internal enterprise traffic in 2004 and 2005 to characterize traffic recorded at a medium-sized site. They anonymized the traces by *tcpmkpub* [87] and made them publicly available [88]. Researchers at the LBNL presented a first look at this data [89] and provide a high-level view of many aspects of enterprise network traffic, including characterization of applications used only within enterprises and not on WAN environments (e.g., Windows protocols).

The LBNL enterprise dataset has unfortunately also been used in a study investigating the quality of anonymization techniques (including *tcpmkpub* [87]). Coull et al. [90] presented techniques to infer network topology and to de-anonymize servers in anonymized network data, using only the data itself and public information. While the authors intended to inform the community about the weaknesses of anonymization methods, they failed to consult with the data providers about the de-anonymization attempt[1]. As a reaction, Allman and Paxson [91] proposed a code of conduct for data providers and data seekers. Furthermore, they stress the importance of sharing policies in addition to technological data anonymization techniques, an issue that is also supported by other researchers in the Internet measurement community [92].

WAND Network Research Group

The WAND network research group [93] is located at the University of Waikato Computer Science Department. WAND is a network measurement research group, performing among other things collection of long trace sets, network analysis, development of analysis software, and network simulation and visualization. In the field of passive network measurements, WAND is best known for the Waikato Internet Traffic Storage archive (WITS) [94] and the development of the DAG measurement cards. The WITS archive contains about

[1] Later comparison of the de-anonymization results with ground truth showed that most of the IP addresses in the LBNL dataset have in fact been incorrectly de-anonymized [91].

200GB of traces taken at different locations starting in 1999, with the most recent trace dating from January 2009. To-date, only statistical summaries of the traces are publicly available, but traces are planned to be shared in the near future according to their webpage [94] (as viewed in November 2009). WAND developed the DAG measurement cards, a flexible and efficient hardware solutions for network measurements. Today, Endace [36] is responsible for the support and development of DAG equipment, founded in 2001 as spin-off company. In addition to publications describing the development of DAG cards, WAND also contributed scientific measurement results based on WITS data traces, like the analysis of long duration traces by Nelson et al. [95] covering protocol mix, network trip times and TCP flag analysis.

WIDE Project and MAWI

The Widely Integrated Distributed Environment (WIDE) project [96] was launched in 1988 in Japan and is made up of more than 100 loosely bound organizations from all over the world. The visionary goal of WIDE is to construct a dependable Internet *"that can be used by people from all walks of life in any situation with a sense of security"*. WIDE research activities cover all different layers of the Internet, including activities such as flow measurements with sFlow/NetFlow and analysis of IPv6, DNS and BGP routing information. The "Measurement and Analysis on the WIDE Internet" (MAWI) working group furthermore provides a traffic repository of data captured on the WIDE backbone [97], focusing mainly on DNS and IPv6 traffic measurements. The MAWI repository shares their anonymized packet-header traces from trans-pacific backbone links (limited to 18 Mbit/s, later updated to 100 and finally 150 Mbit/s) with other researchers. This dataset contains traces of 15 minutes collected once per day at 2PM from 2001 to 2009, allowing longitudinal studies such as sketching the evolution of Internet traffic during 7 years in Borgnat et al. [98].

EU Framework Projects SCAMPI and LOBSTER

SCAMPI [99] was a two-and-a-half year European project sponsored by the Framework Project (FP) 5 Information Society Technologies (IST) program of the European Commission, starting in April 2002. SCAMPI involved ten European partner organizations, with the goal to develop a scalable monitoring platform for the Internet in order to promote the use of monitoring tools for improving services and technology. The original project was succeeded by another IST project under FP6, the LOBSTER [100] project. LOBSTER continued the deployment of an European Traffic Monitoring Infrastructure based on distributed monitoring sensors capable of collecting on link speeds of up to 10Gbit/s. Besides the deployment of a monitoring infrastructure, LOBSTER developed a number of monitoring and visual-

ization tools, such as Stager [101], a tool for aggregating and presenting network statistics. Researchers from both SCAMPI and LOBSTER were actively involved in the development of the IPFIX flow format standard [30]. LOBSTER also made a number of network attack traffic traces available for download [102]. The attack data includes complete packet traces of individual network flows including polymorphic shellcode, detected by an emulation-based method [103]. Other activities included development of a generic anonymization framework for network traffic [104], which was developed after revealing vulnerabilities in existing pseudonymization approaches [57, 58]. The LOBSTER project concluded on 30 June 2007, but the sensor network remains operational. Since then, the FP7 project MOMENT [105] works on an unified interface for representation and retrieval of passive measurement data provided by LOBSTER nodes, but also of active measurement information from related projects such as ETOMIC [106] and DIMES [107]. MOMENT sets out to create added value from single measurement infrastructures by integrating these results.

SPRINT ATL

In early 2000, Sprint's Advanced Technology Labs (Sprint ATL) started with the design and deployment of a passive monitoring Infrastructure, called IPMON [45]. The IPMON system consisted of a number of measurement nodes, a central data repository and an analysis platform for offline analysis of the data. The measurement nodes are technically similar to the OC48MON systems and were located at geographically distributed Points of Presence (POPs) in order to collect data on different peering and backbone links, with bandwidths up to OC192 (10Gbit/s). As a result, IPMON was able to collect packet-level traces on about 30 bidirectional links in the US Sprint IP backbone. A resulting analysis of 24h traces collected on average every two months between 2000 and 2005 was published online [108]. This analysis reveals general traffic characteristics such as utilization, protocol breakdown and packet size distribution. Sprint's applied research group is also focusing on next-generation wireless systems, data mining and security. The latter research topic includes development of a continuous monitoring platform for high-speed IP backbone links, CMON, the successor of IPMON. CMON [109] was intended to provide a continuous packet stream for detection of anomalies, unusual events and malicious activities. In fact, researchers at Sprint developed an efficient online port scan detection and tracking system utilizing the CMON traffic monitoring architecture [110, 111].

AT&T Labs

The NetScope project [112] is a measurement project that combines active and passive measurements on the AT&T network. In this framework, a monitoring system to collect

and filter packet-level data has been developed, named PacketScope [113]. Researchers at AT&T Labs performed various studies on the resulting data collected from access and backbone links, including quantification of P2P traffic on the Internet during the years 2002-2004 [22, 114, 115]. Recently, Qian et al. presented a fresh look at TCP in the wild on data collected on AT&T backbone and access links during 2008 [116]. This study found no qualitative differences in TCP flow sizes and durations, but higher flow rates compared with an earlier study on AT&T backbone data from 2001 [117]. Sizes of elephant, cheetah, and tortoise flows[2] increased by an order of magnitude compared to a study on backbone data from 2002 and 2003 [70].

Other Measurement Projects

Besides these big measurement projects, various researchers carried out some other passive network measurements. Maier et al. [118] studied residential broadband Internet traffic on data from DSL connections of a large European ISP in 2008 and 2009. This study includes packet- and flow-level characteristics and showed that HTTP, and not P2P traffic dominates the traffic observed. Arlitt and Williamson took a year-long packet-level trace in 2004 on the 100Mbit/s Ethernet campus network at the University of Calgary in order to analyze TCP reset behavior [119], showing that large portions of TCP connections (15-25%) have at least one TCP reset. Also Moore and Papagiannaki used packet-level data collected on a campus network based on Gbps Ethernet to compare network application identification methods [11]. They took these measurements with Nprobe, a passive measurement architecture to perform traffic capturing and processing at full line-rate without packet loss [120]. Crotti et al. [121] and Dusi et al. [122] collected network traces on the edge router of the faculty campus network at the University of Brescia in order to study statistical mechanisms for classification of network traffic. Crotti et al. showed that Internet flows from traditional applications such as HTTP, SMTP and POP3 can be classified by statistical methods without parsing payload data. Dusi et al. showed that statistical methods can even be used to identify applications tunneled in encrypted SSH tunnels. Together with active measurement approaches, Medina et al. [123] used passive measurements of a ICSI lab web server during two weeks in 2004 to track the deployment of transport-related mechanisms in transport protocols. In this work, passive measurements are used to infer usage of specific TCP features at web clients, such as TCP options usage and window size advertisements.

Measurements from networks with higher aggregation are often only available in form of flow data. Perenyi et al. [124] based their identification and analysis method of peer to

[2]Elephant, cheetah, and tortoise flows correspond to heavy-hitter flows in size, rate, and duration respectively. Heavy-hitters are in [116] and [70] defined by flows greater than mean plus three standard deviation.

peer (P2P) traffic on NetFlow data from an ADSL network with around a thousand ADSL subscribers in Hungary. They show that it is possible to identify P2P traffic without access to packet payloads with the help of heuristics on flow properties, such as concurrent TCP and UDP flows between IP pairs, and default P2P port numbers. In the 2009 Internet observatory report, Labovitz et al. [125] studied two years worth of flow data collected on more than 3000 peering routers across 100 global- and regional network and content providers. The study highlighted changes in the logical topology away from the traditional Tier-1 core towards a flatter and more interconnected hierarchy. They also observed changes in application usage, including decline of P2P file sharing traffic at the expense of increasing amounts of streaming media (often via HTTP), confirming recent results by Maier et al. [118].

3.1 Publicly Available Datasets

Generally, access to modern packet-level traces from large-scale Internet links is uncommon. The only publicly available datasets that are frequently updated are published by CAIDA in the form of anonymized OC192 backbone traces [126] and by WIDE in the form of anonymized traces from trans-pacific OC3 links [97].

Since researchers without access to their own measurement infrastructures do not have the possibility to study recent data other than the datasets mentioned above, they need to perform their research on a relatively small set of publicly available, but not updated and thus somewhat outdated network traces. These datasets include the historical datasets of the Internet traffic archive [81], the anonymized enterprise traffic datasets from LBNL of 2005 [88] as well as NLANR's traces from OC3/12/48 links up to 2006 [67]. Note that the NLANR traces do not include complete packet headers, but only contain the first 16 byte of the transport header, which does not suffice to store complete TCP headers and therefore limits the analysis possibilities (such as investigating TCP option deployment).

Besides these general packet header traces, there are also public traces of special network events such as network attacks and worm outbreaks. CAIDA until recently continuously published backscatter traces, including worm outbreaks and Denial of Service (DoS) attacks [126], while the LOBSTER project made code injection attack traffic from 2007 available for the research community [102].

3.2 Data Sharing Approaches

The few data providers sharing datasets applied different anonymization methods to balance privacy requirements of data owners and providers with the information requirements of researchers (i.e., handles the tradeoff between data privacy and utility). Traces available

at the Internet Traffic Archive (ITA) [81] and MAWI Traffic archive [97] are anonymized with *tcpdpriv* [82]; the LBNL enterprise traces [88] with *tcpmkpub* [87]; and CAIDA traces [126] with *CryptoPAn* [41]. Attempts to encourage the Internet measurement community to share data (e.g., Internet measurement data catalog DatCat [127] and PREDICT repository [128]) have been of limited success, probably due to stringent sharing policies[3] which are often stated cautious, not least because of disclosed vulnerabilities of current anonymization methods (such as successful trace de-anonymization attempts [57, 58, 90]).

Recently, a number of researchers acknowledged the need for alternatives to the existing anonymization methods. Many of these approaches could be effective for enabling secure access to even highly sensitive data, i.e., traces including user payload. Parate and Miklau [129] proposed a sharing framework in which trace owners can match an anonymizing transformation of communication data with the requirements of analysts. In contrast to existing anonymization methods, the framework should in this way enable formal reasoning about the impact of anonymization operations on trace utility and privacy.

Alternative ways to anonymized data sharing are "move code to data" solutions[4]. Simple "move code to data" solutions[5] might be a straightforward way to provide researches with access to data (which could even be un-anonymized), considering that analysis tools are usually open-source and thus unproblematic to share from an legal/ethical point of view. However, the success of this approach depends on cooperation efforts of the data providers and thus is not scalable. These efforts include time expenses (e.g., safety-review and installation of the code) and allocation of computational resources. Mogul and Arlitt [130] presented a prototype of SC2D, a framework for shipping flexible analysis code to the data. The proposed solution is based on a layered framework to facilitate usage and verification of privacy and security properties of the received code for the data provider/owner, but the basic scaling issues remain.

As another solution to the privacy/utility tradeoff in data sharing, Mirkovic [131] proposed a privacy safe sharing framework based on secure queries. Similar to Mogul and Arlitt's SC2D, raw traces are not copied and shared. Instead, data access is re-directed through an online interface providing a query language, which allows customized sets of queries to be run on the data and returning de-sensitized, aggregated information fitting the specific research goals. Individual privacy policies can thus be enforced by the query lan-

[3] DatCat mainly shares meta-data, and access to the actual data needs to be negotiated with the respective data owners; PREDICT traces are only available to researchers inside the US.

[4] Instead of the data providers releasing data, the data seekers give their analysis tools to the data providers, who perform the requested analysis and return the results to the seekers.

[5] Basic "move code to data" is the ad hoc solution we currently use to give other researcher access to our MonNet data traces.

guage interpreter. Since the framework only returns aggregated information, its utility is limited and many analysis tasks cannot be supported.

The most advanced "move code to data" approach was recently presented by Mittal et al. [132]. They propose secure mediated trace access using black-box permutation analysis. This approach allows data providers to detect information leaks in the analysis programs received from data seekers. The data provider can verify the policy compliance of the analysis results by repeated permutation of sensitive fields in the input traces inside a sandboxed environment, followed by an analysis and security assessment of the resulting output before it is sent back to the data seeker.

Kenneally and claffy [92] proposed a combination of a policy framework that satisfies obligations of both data seekers and data providers, and a technology framework able to enforce these obligations. The Privacy Sensitive Sharing framework (PS2) should reveal that actual data-sharing is less risky (in form of privacy risks) than *not* sharing data (and inability to understand and anticipate the Internet and its security threads) considering the importance of modern Internet.

4

Thesis Outline and Research Summary

Part I of this thesis comprises the syntheses chapters which include our lessons learned and reflections from passive Internet measurements. Part II is a collection of individual reports and papers published in scientific journals, conferences and workshops as listed in the preface on pages v and vi. In the following sections we describe how the papers conceptually relate to each other and provide short summaries of each specific paper including discussions about their main contributions. Fig. 4.1 depicts a schematic overview of the papers and groups them into a logical structure to guide the reader through this outline.

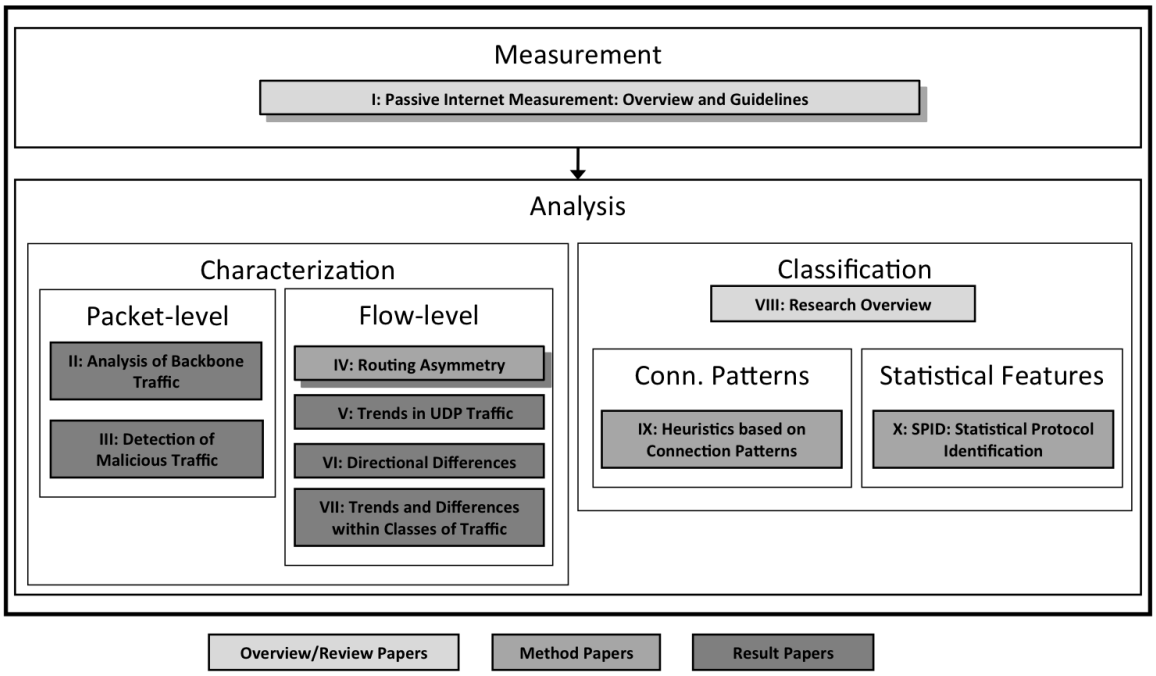


Figure 4.1: *Schematic of the papers included in this thesis. Grey-shaded boxes represent the papers, with grey-shades indicating the type of contribution: Overview and review papers in light-grey; Papers focusing on analysis methods in medium-grey; Papers focusing on the results of characterization in dark-grey. Background shades to two boxes represent the secondary purpose the papers.*

4.1 Internet Measurement: Collecting Backbone Traffic

***Paper I**: Passive Internet Measurement: Overview and Guidelines based on Experiences* discusses the major challenges we encountered during our Internet traffic collection. We first give a detailed overview of different design options and important considerations for backbone measurements. We then discuss the challenges in order of their chronological appearance: we had to sort out a number of legal and ethical issues with legal practitioners and network operators, followed by operational difficulties that needed to be solved. Once we managed these legal and operational obstacles establishing trust relationships with the network operator and an ethics committee, a third challenge is given by various technical difficulties when actually measuring high-speed links. Technical issues range from handling the vast amounts of network data to timing and synchronization issues. Finally, we describe how we navigated the aforementioned issues in MonNet, which we consider a successful Internet measurement project. We therefore have been able to provide concrete lessons learned based on our experiences. One such important lesson to note was our failure to establish policies regarding sharing our data with other researchers. In total, the paper presents an overview and guidelines for setting up and performing passive Internet measurements. We would have saved a lot of time, money and energy if a similar paper had been available five years ago. For this reason, we believe that this type of paper can be of great value for researchers and practitioners planning and designing Internet measurements, currently or in the future.

4.2 Traffic Analysis: Characterization of Internet Traffic

4.2.1 Packet-level Characterization

***Paper II**: Analysis of Internet Backbone Traffic and Anomalies Observed* reflects packet characteristics on SUNET Internet backbone traffic and points out misbehaviors and potential problems. We used the bidirectional traffic collected on GigaSUNET in Spring 2006 to provide a summary of current protocol usage including comparisons to prior studies. The analysis confirmed that IP options and Explicit Congestion Notification (ECN) are virtually not applied. On GigaSUNET, we observed minor fractions of fragmented IP traffic (0.06%), with UDP accounting for a majority of the fragments. The latter observation stems from increased deployment of TCP Path MTU Discovery, which we showed to be dominating. Regarding packet size distribution, three findings should be noted: (i) we found packet size distribution on GigaSUNET to be bimodal, i.e., most packets were either small (44% between 40 and 100 byte) or close to the Ethernet Maximum Transmission Unit (MTU)

size (37% between 1400 and 1500 byte). Earlier measurements (up to 2002) on backbone links [45, 76], and also more recent wide-area measurements in China during 2006 [133] reported of substantial fractions of packets with default datagram sizes (i.e., 576 bytes [134]); (ii) in our data from 2006, IP packet lengths of 628 bytes were even more common (1.8%) than the default datagram size (<1%). We identified these packets to be artifacts of a then popular P2P application (Gnutella [135]); (iii) we do not see any jumbo packets except for BGP updates between routers and one single custom application optimized for bulk transfer. We furthermore identified additional headers introduced by VPN as one cause for the otherwise rare occurrence of IP fragmentation, which should advise application developers to use smaller MSS values. Finally, we highlight several types of misbehaviors within IP and TCP headers, which led to a closer investigation of header anomalies in *Paper III*.

***Paper III**: Detection of Malicious Traffic on Backbone Links via Packet Header Analysis* provides a systematic listing of packet header anomalies together with their frequencies as seen "in the wild". In the paper, we analyzed backbone data recorded on GigaSUNET in Fall 2006 regarding consistency of network and transport layer headers (i.e., IP, TCP, UDP and ICMP) in order to study occurrences of malicious activities in modern Internet traffic. This analysis approach is similar to the header "walk-through" approach used for IDS traffic normalization in Handley and Paxson [136]. In our paper, we focus on the results of this header analysis and present occurrences of header anomalies observed in today's Internet traffic. We also provide detailed discussions about possible causes for the inconsistencies and their security implications for networked devices. The results are interesting for practitioners and researchers, and form valuable input for intrusion detection systems, firewalls and the design of all kinds of networked applications exposed to network attacks.

We found inconsistencies in protocol headers in almost every aspect analyzed, including incorrect or incomplete series of IP fragments, IP address anomalies and other kinds of header fields not following Internet standards. As a general observation, it is surprising to see that we still found many old, well-known attacks. On the upside, our data did not include some former popular attacks, such as Ping-of-death and the IP source route exploit. Generally, we observed a constant noise of malformed or inconsistent packet headers (which are however a very tiny fraction of the total number of packets), consistent with the constant scanning activities we observed on a flow-level in *Paper VII* and the observations of incessant background radiation on IP sinkholes (i.e., unused IP address spaces) [66]. In some cases, this type of background noise is likely to be caused by software implementation or hardware errors, as observed on an 100Mbps Internet access link during 1999 by Paxson [137]. However, we believe that many inconsistencies could also be attributed to

the possibility that even inexperienced hackers today can generate more or less any type of packet header with existing networking tools.

We also observed a number of exceptional events of malicious activity. We identified an ICMP DoS attack with otherwise unsuspicious echo reply messages by analyzing IP addresses regarding reserved IP spaces. We observed a sequence of fragmented datagrams that has been sent in high intensity from a single host for short time intervals. Our detailed analysis of the fragment series revealed a directed Frag attack, using incomplete fragment series with the intention to exhaust resources at the receivers. Filtering IP ID values of zero appeared to be a successful approach to detect different fragmentation anomalies. Observations of the reserved bits field of the TCP header revealed a series of SYN/ACK attacks. Port number values of zero proved effective in detecting port scanning campaigns, both on TCP and on UDP. Finally, our analysis revealed an ICMP DoS attack using ICMP redirect messages.

4.2.2 Flow and Connection-level Characterization

Despite providing a valuable understanding of packet-level behavior of Internet traffic, *Papers II and III* revealed that pure packet-level analysis is often not enough to fully understand some of the observations, such as the origin of unusual packet sizes, fragmented traffic and network attacks. By applying the method described in Section 2.6 to correlate packets into flows or connections/sessions, we will now provide further insights and also re-validate some common assumptions about network traffic.

***Paper IV**: Estimating Routing Symmetry on Single Links by Passive Flow Measurements* sheds light on the assumption of traffic symmetry which researchers and developers often embedded into traffic analysis and classification methods [8–10]. We developed a simple flow-based symmetry estimation method, FSE, a normalized metric allowing to assess and compare traffic symmetry of links on a flow-level. We also published a tool implementing the proposed method[1], and applied it to a heterogeneous dataset, resulting in several valuable reference data points on traffic symmetry. The results confirm anecdotal reports that traffic symmetry typically does not hold for non-edge Internet links, and decreases as one moves toward core backbone links, due to routing policy complexity. Our proposed metric for traffic asymmetry induced by routing policies should help the community to improve traffic characterization techniques and formats, but also support quantitative formalization of routing policy effects on links “in the wild”.

[1] The tool is available at http://www.cse.chalmers.se/ johnwolf/FSE.

***Paper V**: Analysis of UDP Traffic Usage on Internet Backbone Links* re-validates the assumption that TCP is the dominant transport protocol on the Internet, as reported repeatedly ([45, 75] and our own results from 2007 in *Paper II*). We investigated UDP traffic in traffic traces collected in the period 2002-2009 on several backbone links located in the US and Sweden. We assess the fraction of UDP traffic in terms of flows, packets and bytes. According to this data, TCP is still dominant in packets and bytes, but the use of UDP as a transport protocol has gained popularity recently, especially in terms of number of flows. Our first analysis suggests that most UDP flows use random high ports and carry few packets and little content (payload), consistent with its use as a signaling protocol for popular P2P applications (*Paper VIII*). This trend may again change with the advent of IPTV and UDP based P2P applications, which not only signal, but also transport large data volumes via UDP [138, 139]. The effect of increasing fractions of UDP traffic volumes on network stability will depend on the congestion control abilities of these application protocols. The increasing trend of UDP therefore needs to be monitored closely in order to keep track of possible undesirable effects.

***Paper VI**: Differences between In- and Outbound Internet Backbone Traffic* utilized the Spring 2006 dataset from GigaSUNET to highlight significant directional differences in traffic properties between in- and outbound traffic. While some high-level analysis, like cumulated traffic volumes or protocol breakdown, could suggest an even distribution between inbound and outbound traffic, this study reveals that there are a number of significant directional differences found on different protocol levels (IP, TCP and UDP). Traffic properties differing per direction include IP fragmentation, TCP termination behavior and TCP options usage. Our analysis includes a focus on TCP connection properties, yielding two classes of traffic as the main reasons for the directional differences: malicious traffic and P2P (file sharing) traffic. These results highlight the importance of traffic classification techniques (as discussed in *Papers VIII-X*) that allow detailed analysis on isolated classes of traffic.

Our investigation of malicious behavior confirms the suspicion that most anomalies indeed originate on the outside, on the “unfriendly” Internet. We showed that anomalies on GigaSUNET were between 3 and 9 times more common among inbound data. Typical university campus networks, even student networks, are comparably well behaving, probably due to more attention to configuration and administrative efforts.

P2P file sharing traffic was a second source heavily influencing traffic properties. Even a simple analysis based on ports (known to underestimate real P2P traffic numbers) showed that P2P traffic was a major part of the traffic samples, responsible for at least twice as much packets and volume of inbound traffic as of outbound traffic. We found artifacts of

P2P traffic in packet size distribution, TCP connection termination behavior, TCP options and statistical connection properties. We showed that P2P was also a major traffic source for long-duration flows, especially among inbound connections. Additionally, in our traces P2P overlay traffic was responsible for many (at least 18% based on port classification) UDP flows, carrying typically fewer than 3 small sized packets, but responsible for several million distinct IP addresses observed in the traffic.

Paper VII*: Trends and Differences in Connection Behavior within Classes of Internet Backbone Traffic* follows up on the findings of *Paper VI* and presents a connection-level characterization of the GigaSUNET datasets from Spring and Fall 2006. We classified the data according to network application following the coarse grained classification method proposed in *Paper IX*. We then compared three traffic classes (P2P, Web and malicious) in terms of traffic volumes and signaling behavior. The classification allowed us to discuss differences between traffic classes, which we found in many aspects, even if not always expected. Beside diurnal patterns, the time-span of the dataset allowed us to highlight longitudinal trends. These results provide researchers, developers and practitioners with detailed knowledge about trends and influences of different traffic classes observed in our Internet traffic traces from 2006.

Our analysis revealed that overall traffic volumes were increasing for both TCP and UDP traffic, with highest activity in the evenings. On a diurnal basis, P2P and HTTP traffic exhibited different peak times. P2P traffic dominated with 90% of the transfer volumes, especially during evening and overnight. In contrast, HTTP traffic exhibited its peak activities (9% of the data-volumes) during office hours. We observed similar diurnal patterns in terms of connection numbers, even if P2P connections were not as dominating as in terms of bytes. These results indicate that P2P connections typically carried more data than Web traffic. Unsolicited and malicious traffic (including scanning) was responsible for a substantial part of TCP connections (20-30%) and UDP flows (8-12%), but played a minor role in terms of data volumes since it typically consists of 1-packet flows only. It was interesting to observe that the amount of malicious TCP and UDP flows remained constant in absolute numbers both on diurnal and longitudinal basis, even though traffic volumes generally increased. This result suggests that malicious traffic (e.g., scanning attacks) forms a constant background noise on the Internet.

After 2002, measurements from many links reported high fractions of P2P traffic volumes (see *Paper VIII*). P2P was especially dominating European traffic, as reported by Perenyi et al. with 70% P2P on Hungarian ADSL data from 2005 [124], and up to 83% in some regions in Europe during 2007 according to IPOQUE [140]. However, our results

for P2P traffic volumes of up to 90% on the SUNET links are still exceptionally high. An analysis of IP prefixes showed that the main fractions of the P2P volume observed is traffic to and from a large student residential network in Gothenburg (GSIX). The remaining traffic carried relatively low fractions of P2P traffic, which is consistent with the restrictive non-file sharing policies on most Swedish universities. We suspect that the large P2P traffic volumes for student dormitories can be explained by the combination of low risk of legal persecution in Sweden at the time of the measurements[2], inexpensive and high-speed Internet connectivity, the well fitting network demographics (i.e., young, tech-savvy students), and the traditionally central role of Sweden in the international file sharing community (e.g., The Pirate Bay [142]). Since 2008, various studies indicated a decline of P2P traffic in Europe to below 50%, perhaps superseded by one-click hosting and streaming media traffic, which is often carried inside HTTP [1, 118, 125]. An interesting future research task will therefore be reassessing the current fraction of P2P traffic on the SUNET links and comparing them to these recent studies.

In terms of connection signaling behavior, we highlighted major differences between the three traffic classes. The number of unsuccessful P2P connection attempts, which dominated the P2P connection breakdown in Spring, increased further in the Fall traces. The large fraction (43%) of 1-packet flows combined with the large average data amounts per P2P connection manifested a pronounced elephants and mice phenomenon (Pareto principle) within P2P flow sizes. Regarding termination behavior, P2P connections shifted towards higher fractions of proper closings in the Fall traces. HTTP connections on the other hand appear to behave comparably well according to TCP specification at all times.

We showed that TCP option deployment differed significantly between P2P and Web traffic. While P2P traffic reflected an expected behavior considering the default settings in popular operating systems (Windows, Linux), HTTP showed artifacts of a traditional client-server pattern, with some dedicated web servers neglecting negotiation for certain TCP options, especially SACK. We conclude that even though SACK was deployed by almost all P2P hosts and web clients, a number of (high volume) web servers neglect support for it.

4.3 Traffic Analysis: Classification of Internet Traffic

Papers IV to VI showed that correlation of packets into flows is not always sufficient to fully understand traffic properties. An obvious next step toward better understanding is to separately analyze individual flows of certain traffic classes and types, as done in *Paper VII*.

[2]This changed on April 1, 2009, when an anti-piracy law based on the European directive on the enforcement of intellectual property rights (IPRED) [141] came into effect in Sweden.

Note that reliable classification of Internet traffic based on network applications is still an open research issue, partly due to technical challenges caused by the arms-race introduced in Section 1.2.2. However, in this section we identified additional challenges and shortcomings related to validation and reproducibility of current traffic classification efforts. We will then propose two new classification approaches: a coarse grained method for aggregated Internet backbone traffic based on connection patterns, and a fine grained method based on statistical features for traffic traces with some available payload.

Paper VIII*: State of the Art in Traffic Classification: A Research Overview* presents a research review of scientific traffic classification methods published since 1994, including more than 60 scientific papers and more than 80 data sets. We continuously update the complete survey of papers and datasets, that can be found online [64]. *Paper VIII* includes studies published until 2008 and presents a rough taxonomy of traffic classification approaches (based on features, methods, goals and data sets), showing that traffic classification methods have evolved in response to the more sophisticated obfuscation techniques of network applications (i.e., the arms-race). The review also reveals shortcomings with current traffic classification efforts:

- *Lack of shared, modern data sets as reference data*: The variety of data sets used does not allow systematic comparison of methods. Few research groups (can) share their datasets. The field of traffic classification research still needs publicly available, modern data sets as reference data for validating approaches. This need, however, requires clear policies for data sharing, including accepted anonymization and desensitization guidelines. We will further discuss this topic in the concluding Chapter 6.
- *No clear definition for traffic classes*: The poor comparability of results is further amplified by the lack of standardized measures and classification goals. For example, there exists no agreed performance metrics and definitions for traffic classes (e.g., P2P or file sharing). This will also be discussed in Chapter 6.

Despite these shortcomings, we showed how this taxonomy can shed light on questions such as: *"how much of modern Internet traffic is P2P?"* Though we were able to show some trends and indications, we had far too little data available to make conclusive claims beyond *"there is a wide range of P2P traffic on Internet links; see your specific link of interest and classification technique you trust for more details."*

(a) P2P ports

P2P Application(s)	Proto(s)	Port range(s)
BitTorrent	tcp/udp	688[0-9], [1-5]6881, 32459, 49152
eDonkey	tcp/udp	466[0-8], 14662
	udp	4672, 14672
DirectConnect	tcp/udp	41[1,2], 1412
	udp	9183
Fasttrack (Kazaa)	tcp/udp	121[4,5], 4329
Gnutella	tcp/udp	634[6-9]
Napster, WinMX	tcp/udp	5555, 6257, 66[66,77,88,99]
Soulseek	tcp	2234
Soribada	tcp	22321
MP2P	tcp/udp	41170

(b) non-P2P ports

Appl.	Proto(s)	Port(s)
FTP	tcp	20, 21
SSH	tcp	22
Telnet	tcp	23
Mail	tcp	25, 110, 143, 220, 993
DNS	tcp/udp	53
NTP	udp	123
Netbios	tcp/udp	135, 137, 139,445
BGP	tcp/udp	179
RTSP	udp	554

Table 4.1: *Port numbers used for the port-based heuristics in Paper IX*

4.3.1 Classification of Backbone Traffic based on Connection Patterns

***Paper IX**: Heuristics to Classify Internet Backbone Traffic based on Connection Patterns* proposes a set of heuristics for classifying backbone-type data according to applications. The proposed heuristics are mainly based on connection patterns of the Internet hosts observed, but in some cases also take port numbers into account (listed in Table 4.1). As a result, our heuristics do not require packet payloads, and are as such intended to provide researchers and network operators with a comparable simple and yet relatively privacy sensitive method to get insight into the type of data carried by their links. The heuristics work on traces as short as 10 minutes, which allows operators to classify snapshots of their traffic relatively fast, by only adjusting applied thresholds and parameters empirically. The heuristics can be used to classify backbone traffic according to a number of applications, including P2P traffic, web traffic and other common applications. Furthermore, we introduced a rule that successfully identifies network attacks, which is an additional feature for network operators and researchers interested in network security and intrusion detection. We based some of the proposed heuristics on two existing methods [21, 124]. Since our data did not include payload to perform validation with signature based methods, we had to rely on the verification methods of these original heuristics. Additionally, we performed a careful manual analysis of the resulting classification, pinpointing obvious cases of false positives. Both previous sets of heuristics overestimated the number of P2P flows, mainly because attacking traffic is not taken into account accordingly. By combining the successful rules of the two methods and adding additional, necessary rules, we presented a set of refined and updated heuristics, which we applied on the OptoSUNET traces from Spring 2006.

Recently (during 2009), we got access to backbone traces including 40 bytes of packet payload, which allowed us to compare the results of the heuristics proposed in *Paper IX* with a signature-based method. The signature method used as ground-truth method was introduced in Karagiannis et al. [24] and later refined and updated in Kim et al. [13]. The results of a first comparison show that port-based heuristics (Table 4.1.a) perform accurately for P2P traffic, but can only classify a small part of the P2P traffic, which was also confirmed by Kim et al. However, the heuristics based purely on connection patterns in fact overestimate the fraction of P2P traffic on TCP. On the verification trace, the proposed heuristics classified 89% of the traffic volume as P2P and left 2% unclassified. The signature-based classification method however marked only 81% of the data as P2P, but left 5% of the traffic unclassified. We believe that the resulting overestimation can be explained by three main factors: (i) all parameters and thresholds have been optimized for the OptoSUNET 2006 datasets and have not been adjusted in this verification test; (ii) while the heuristics take asymmetrical (thus unidirectional) flows into account, they work more accurately on bidirectional flows. In the verification trace only about 30% of the traffic was carried in symmetrical (i.e., bidirectional) observed flows, in contrast to more than 75% on OptoSUNET 2006 traces used in *VII and IX*; (iii) we found that a part of the overestimate (2.5% of the total traffic volume) was contributed by heuristic *'H3: Port Usage'*. While this heuristic marked a substantial number of actual P2P flows (often accompanied by additional P2P heuristics), it turns out not be a strong indication for P2P applications by itself and needs to be refined based on this ground-truth verification. Furthermore, we believe that heuristic *'H5: unclassified, long flows'*, an optional heuristic regarded as weak to start with, should not be used anymore, since long lasting flows carrying substantial amounts of data can besides P2P be generated by many other applications, foremost all types of streaming media applications.

4.3.2 Classification of Internet Traffic based on Statistical Features

Methods based on connection patterns require observation of multiple flows per communication endpoint in order to be able to infer the application used. Furthermore, the results of the re-validation of the heuristics we presented in *Paper IX* confirmed our suspicion that heuristic methods applied on connection patterns of Internet flows can only provide a very rough traffic decomposition into traffic classes, but are not accurate enough to pinpoint exact applications of single flows. But there are recent advances in traffic classification that are promising to provide accurate and complete protocol identification on flows: statistical methods using flow features such as packet-sizes and inter-arrival times [8, 121]; and payload based approaches, which are often based on manually created payload signatures [22] or automatically created signatures [143, 144]. Signature matching methods however are

of limited use for classification of obfuscated or encrypted protocols, such as the BitTorrent Message Stream Encryption (MSE) [145], encrypted eDonkey [146], and Skype [147]. Another payload based approach was proposed by Dreger et al. [31]. The Dynamic Protocol Detection (DPD), part of the Bro IDS [137], parses byte streams dynamically with chains of protocol analyzers in parallel until the application is identified. Protocol analyzers apply a set of protocol detection heuristics, which currently includes mainly regular expression signatures. In *Paper X* we present an alternative method that combines statistical and payload based methods, which has the goal to provide reliably identification of application layer protocols within the first few packets of a flow without the need for manual creation of signatures.

***Paper X**: Statistical Protocol IDentification with SPID: Preliminary Results* presents SPID, the Statistical Protocol Identification algorithm. The SPID framework utilizes various statistical packet and flow attributes (i.e., features) to identify application layer protocols by comparison of probability vectors of attributes to protocol models of known protocols. We can define these attributes by all sorts of packet and flow data, ranging from traditional statistical flow features to application-level characteristics, such as byte frequencies and offsets for common byte-values. In this sense SPID is a hybrid technique, utilizing efficient generic attributes, which can include deep packet inspection elements by treating them in the same way as statistical flow properties. Even though SPID does not require complete payload, it needs at least some payload from the first packets in each flow[3] to work accurately.

We obtained initial results when identifying a small set of protocols within a pre-classified set of flows collected by Szabo et al. [148] on an access link during 43 hours in October 2007 with capture length of 96 bytes per packet, i.e., 42 bytes of application header data. These results were promising, showing 100% average precision with a recall of 92% for the five protocols tested (BitTorrent, eDonkey, HTTP, SSL, SSH). In the paper we also discussed interesting and relevant future directions with this approach, such as finding the optimal set of the flow features used or testing the robustness of the algorithm against different network environments, ranging from LAN to backbone links.

We believe that SPID has the potential to become a simple and efficient classification algorithm, providing accurate and fine grained identification of network flows on application-protocol level. Additional tests[4] showed that SPID can provide classification accuracy of $>$90% with high recall even for obfuscated and encrypted protocols that are hard to clas-

[3]The exact numbers of the parameters can be adjusted. In the preliminary tests, SPID analyzed 42 bytes of payload in the first 20 packets of each session.

[4]Study under progress, thus indicated only briefly.

sify, such as MSE BitTorrent, Skype, encrypted eDonkey and Spotify [149]. Note that we performed these tests with an already reduced set of 12 attributes meters, combining only a handful of features. First results indicated that attribute meters combining simple statistical flow features (i.e., packet directions and packets sizes) and payload features (i.e., byte frequencies and offsets) are powerful to accurately classify both obfuscated and non-obfuscated traffic.

5

Thesis Contributions and Findings

The objective of the thesis was *to improve the understanding of Internet traffic characteristics by measuring and analyzing modern Internet backbone data*. The main contributions of the thesis are the following:

- *A discussion of design options and important considerations for passive Internet data collection and measurement*:
  We critically discussed aspects of passive measurements on large-scale network links and the lessons that were learned from our successful measurement project. These lessons can serve as guidelines to others setting up and performing (future) passive Internet measurements. Key findings are:
  - We identified major obstacles to Internet data collection and sharing, including legal, ethical, operational, technical, and economic challenges.
  - We gave an overview of how to manage operational and technical challenges by means of subtle engineering practices and trust relations to network operators. However, we identified legal and ethical challenges as unsolved problems for the research community; we managed them ourselves in an ad hoc fashion only.
  - We highlighted the lessons we learned regarding the importance of shared data for the validation and reproducibility of research results, but acknowledge that we did not succeed in establishing clear policies to make our data accessible to other researchers.
- *A detailed header analysis of modern wide-area Internet traffic*:
  We revealed deployment of protocol-specific features and provided a systematic survey of packet header anomalies. Key findings are:
  - IPv4 packet size distribution is bi-modal, rather than tri-modal as it was observed on several backbone links until the early 2000s.
  - IP fragmentation, Explicit Congestion Notification (ECN) and IP options are rarely used, but Path MTU Discovery and Selective Acknowledgement (SACK) are prevalent.

  - We found protocol violating values in almost any header field, some of which can be harmful. These fields include IP header length, address and fragmentation fields; TCP port number, length, flags and option fields; UDP port number and length fields; and ICMP length and type fields.
  - We found many old, well known attack types but did not observe some formerly popular attacks (e.g., Ping-of-death, IP source route exploit) in our data.

- *A validation of common beliefs about Internet traffic on modern Internet data*:
  We provided a simple method and tool to assess and fairly compare flow-based routing symmetry by passive measurements on specific links. We used the method to assess flow routing symmetry on a large heterogeneous set of network traces. We also investigated the usage of UDP on these traces and shed light on the assumption that TCP is the dominant transport protocol on the Internet. While not all of the results are surprising, they represent important data points and indicate global trends. Key findings are:
  - Routing symmetry is uncommon on non-edge Internet links and decreases with higher levels of "coreness", i.e., as paths move toward highly aggregated links. This implies that traffic analysis tools and methods should assume little routing symmetry unless intended only for stub access links with no path diversity.
  - TCP is the dominant transport protocol in terms of packets and bytes, but fractions of UDP traffic show an increasing trend during the last seven years and are already larger than TCP in terms of flow numbers. We attribute this development to popular P2P applications using UDP for their overlay signaling traffic.

- *A detailed analysis of Internet flows and connections*:
  We revealed differences between different classes of network traffic. Key findings are:
  - Inbound traffic to the Swedish academic network includes several times more "hostile" activities than outbound traffic, consistent with a higher degree of attention to system configuration and administration on Swedish campus and student networks.
  - P2P, Web, and malicious traffic differ significantly in diurnal traffic pattern, TCP option deployment, connection establishment and termination behavior.

- *An assessment of state-of-the-art traffic classification methods*:
  We reviewed current traffic classification efforts and revealed shortcomings, i.e., lack of publicly available modern reference data and a lack of standardized measures and classification goals. We then proposed two classification methods: (i) a payload-independent classification method for aggregated backbone traffic based on connec-

tion patterns; and (ii) a classification method for fine grained protocol identification by utilizing statistical packet and flow features. Key findings are:

- Connection pattern heuristics can classify traffic in a privacy sensitive way but accomplish only coarse grained decomposition of backbone traffic. This method is useful for providing quick insight into the type of data carried on large Internet links and for estimating Internet background noise (background radiation [66]).
- Classification based on statistical features is a promising approach to reach sufficiently accurate identification of individual application layer protocols, as required by many traffic management, security, and policy enforcement tasks. Preliminary results indicate that SPID, our method, is capable of accurate classification of most protocols in a simple and efficient way.

6
Conclusions

Considering the role of the Internet as a critical infrastructure of global importance, we claim that it is crucial for the Internet community to understand the nature and detailed behavior of modern network traffic. A deeper understanding supports optimization and development of network protocols and devices, and furthermore helps to improve the security of network applications and the protection of Internet users. In this thesis, we therefore presented methods and results contributing additional perspectives on global Internet behavior at different levels of granularity. We are confident that we advanced the understanding of the modern Internet by presenting current characteristics of Internet traffic based on a large amount of empirical data. Furthermore, we discussed methodological aspects of passive Internet measurement and data collection.

This final section gives overall conclusions and lessons learned in carrying out the research presented. We also emphasize the limitations of Internet measurement research, which point to future research possibilities.

Traffic Analysis and Characterization

We started our traffic analysis efforts with a detailed investigation of individual packet headers. While our study of different aspects of protocol deployment and header consistency revealed answers, it (unsurprisingly) also brougth up new questions. To follow the resulting learning curve, we had to zoom out to higher levels of data aggregation, first to the flow-level and then to classes of Internet traffic. We have presented detailed Internet traffic characteristics on both packet and flow granularities. The results, representing actual, empirically measured properties of network traffic, are important for scientific network simulation and modeling and are relevant as well for operational purposes such as network management, traffic engineering, and network security.

In the course of our research, we gained experience in dealing with large-scale Internet data. Measuring actual Internet traffic revealed a great deal of behaviors that do not follow standards, which was at first somewhat unexpected for us as naive researchers expecting

textbook behavior. Almost every possible inconsistency in protocol headers and connection signaling appears "in the wild", highlighting the need for careful design and robust implementation of network applications and infrastructure to keep them resilient against the multitude of network attack types in the global Internet environment. The changing nature of Internet traffic, with new protocols and applications appearing continuously, requires ongoing revalidation of common assumptions. As indicated in the study of UDP traffic, assumptions that have been valid for a long time can be misleading if they are not revisited periodically and from varying vantage points.

In the MonNet project, we made most of the analyses with in-house tools developed from scratch. Available tools, such as the CoralReef suite [79] or dagtools [36], have only been used to validate the results from our own tools. The positive aspects of this approach are that it allowed us to optimize the tools for each specific research question. Furthermore, we learned a great deal by dealing with the detailed problems and pitfalls of handling vast amounts of data. In fact, our experiences in developing tools to analyze the backbone traffic traces described in *Paper II* inspired us to make a systematic investigation of header anomalies presented in *Paper III*. Relying on in-house tools also has the advantage of high transparency of the complete chain of data handling tasks, whereas using external tools and APIs introduces the risk of misconceptions about data-handling details, which can produce biased or incorrect results.

The choice to use our own tools also had a downside. Developing tools takes considerably more time than building on top of existing tools and APIs. Early versions of our tools could only handle certain header types and trace formats used on the specific link and measurement hardware in our environment. We later increased the versatility of some tools by integrating CoralReef libraries, which can handle many common link layer protocols and trace formats. Another important aspect of standard tools and APIs is the possibility to easily compare results with those from related tools. An example is the choice to use our own bidirectional flow definition in *Papers VI,VII and IX* (described in Section 2.6). While our definition proved to be useful to infer the connection behavior of TCP sessions, it complicated the comparison of results with analysis tools based on standard flow definitions such as Netflow or CoralReef's Coralflow, which treat unidirectional flows, discriminated by timeouts rather than by TCP signaling flags.

Traffic Classification

Our research results indicate that exploitation of statistical features is a promising method for reliable traffic classification. This approach may prove useful especially in the face of non-existing or obfuscated payload and further complicating circumstances such as unidi-

rectional traffic flows and large fractions of UDP traffic[1], but it also requires at least partial access to privacy sensitive packet payload. However, as part of the arms race mentioned in Section 1.2.2, statistical features can also be obfuscated to some degree, e.g., by senders deliberately varying inter-packet delays or randomly padding packet payloads. On he basis of our experiences, we believe that hybrid methods taking advantage of port number information and connection patterns could further improve the performance of statistical fingerprinting. Purely heuristic methods as proposed in *Paper IX* were too course grained and inaccurate to be useful for traffic management purposes such as traffic shaping or traffic differentiation. Due to their simplicity, such methods might still be practical for a basic overview of rough traffic decomposition on a given link.

Besides the lack of common reference data, we found that traffic classification research further suffers from a lack of standardized measures. Classification papers use a wide range of performance metrics such as *overall accuracy, precision, recall, F-measure* [13]; *accuracy, completeness* [24]; *hit ratio, false positive ratio* [121]; and *true positive, false positive* [8]. Some of these metrics describe the same phenomena, but with different names. Others are even defined differently. It is possible to convert some of the metrics to others for purposes of comparison, but raw numbers would sometimes be required to perform this task. In any case, we argue that the Internet measurement community would benefit from a set of well defined performance metrics to facilitate straightforward comparison of classification results. Some researchers proposed traditional statistical classification measures such as accuracy, recall and F-number [13], which we followed in this thesis (*Paper X*).

Traffic classification can be applied for various purposes (e.g., traffic management, traffic engineering, security monitoring, accounting, policy enforcement) that require different classification granularity. We find that the lack of singularly defined traffic classes further amplifies the poor comparability of classification results. To give an example, for some purposes, Skype traffic might be regarded as P2P traffic, but not P2P file sharing, while for other purposes it could be classified as voice-over-IP. We believe that the research community would benefit from a set of common definitions for traffic classes in different granularities depending on the purpose. Traffic granularities could be (i) single protocols (e.g., for security monitoring); (ii) network applications (e.g., for policy enforcement); and (iii) protocols merged into application classes (e.g., for traffic engineering and accounting), which could be inspired by categories used in existing work [11].

[1] UDP traffic has no notion of sessions with a clearly defined start and end (thus unclear initiator/responder), which complicates flow classification efforts.

Internet Measurement: Data Collection and Access

To provide a better picture of the global Internet it is crucial to provide possibilities to continuously monitor and collect Internet traffic from various vantage points. We have presented our experiences from passive backbone data collection by breaking the main obstacles down to separate challenges: economic, legal, ethical, operational, and technical considerations. We conclude that measuring traffic on large-scale Internet links is a tedious task, which can be both very expensive and time consuming. However, Internet measurement research, empirical in nature, depends on the quality and diversity of available network traces. We therefore identify a further challenge: the complications of sharing or providing access to the tediously collected data, which is related to the legal and ethical limbo of scientific Internet data collection. Lack of available datasets is a major shortcoming of current traffic classification efforts as well as any other type of traffic analysis. Researchers should not see sharing or providing access to network data only as a courtesy to the community. In any field of experimental research, reproducibility of results is vital, and thus sharing of data is a prerequisite for the scientific process [150]. Access to diverse datasets from real world networks would allow investigation of traffic properties across measurement times and locations and thus enable fair comparison of competing analysis methods on identical reference data.

The status quo in the Internet measurement community is a few datasets from even fewer vantage points available to researchers (Section 3.1). While packet traces with payload are desirable for maximal research utility (see Section 1.5), they are also the most problematic regarding privacy issues. But even packet-header traces, less problematic from a privacy perspective but still relevant to many research problems, are hardly shared. We assume that many researchers with access to data-collection infrastructure would like to share datasets, both to add credibility to their own research results and as a service to the community. However, economic considerations together with the uncertain payoff result in a reluctance of network operators to install measurement equipment for research usage beyond operational requirements[2]. A further reason for the cautious and defensive attitude of potential data providers is the uncertain legal situation and unclear ethical implications of data-sharing. The MonNet project is one such example: while we managed to install measurement equipment in the SUNET backbone to collect data, we have been subject to restrictions (i.e., data anonymization and payload removal) even for our own data analysis efforts. We furthermore failed to negotiate policies regarding sharing our data and were forced to act cautiously. As a result, we can only offer simple mediated access (i.e.,“move code to data”) in an ad hoc

[2]Commercial Internet providers might additionally be reluctant to share data for competitiveness reasons.

fashion to researchers asking for our data. While we experienced that we do not have enough time and resources to act as data mediator for multiple data seekers at the same time, this scalability problem in our case was diminished by the inert reactions of many data seekers once we offered them the mediated "move code to data" solution.

To increase the credibility of Internet measurement as a research discipline, we therefore argue that it is essential for the research community to agree on ways to facilitate the sharing of network data in a manner that balances the privacy requirements of data owners and providers with the information requirements of researchers (i.e., ways to handle the trade-off between data privacy and utility). We believe that current desensitization and minimization methods (Section 3.2) ultimately need to be complemented by sharing policies in order to mitigate the unavoidable vulnerabilities of these privacy protection schemes [57, 58, 90]. We fear that purely technological approaches can not in the long run sufficiently meet the privacy/utility trade-off without additional policy support [92]. First of all, traffic anonymization, reduction and minimization methods can potentially leak information in unexpected ways [130] or, as Allman and Paxson put it, "*they* [data providers] *are releasing more information than they think"* [91]. And secondly, our experience showed that even basic de-sensitization techniques reduce analysis possibilities. Thus, overcautious application of minimization techniques are likely to limit the utility of the data and hinder viable research.

List of References

[1] H. Schulze and K. Mochalski, "IPOQUE Internet Study 2008/2009," 2009, http://www.ipoque.com/resources/internet-studies/internet-study-2008_2009 (accessed 2009.11.27).

[2] Allen Householder, Kevin Houle, and Chad Dougherty, "Computer Attack Trends Challenge Internet Security," *Computer*, vol. 35, no. 4, pp. 5–7, 2002.

[3] kc claffy, "Internet as Emerging Critical Infrastructure: What Needs to be Measured?," in *JCC'08: Chilean Computing Week*, 2008, http://www.caida.org/publications/presentations/2008/uchile/uchile.pdf (accessed 2009.11.22).

[4] N. Brownlee and kc claffy, "Understanding Internet Traffic Streams: Dragonflies and Tortoises," *IEEE Communications Magazine*, vol. 40, no. 10, pp. 110–117, 2002.

[5] k.c. claffy, "Internet Traffic Characterization," Tech. Rep., University of California at San Diego La Jolla, CA, USA, 1994, PHD Thesis.

[6] Sally Floyd and Vern Paxson, "Difficulties in Simulating the Internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 4, pp. 392–403, 2001.

[7] Sally Floyd and Eddie Kohler, "Internet Research Needs Better Models," 2003, vol. 33 of *Comput. Commun. Rev. (USA)*, pp. 29–34, ACM.

[8] L. Bernaille, R. Teixeira, and K. Salamatian, "Early Application Identification," in *ADETTI/ISCTE CoNEXT Conference*, Lisboa, Portugal, 2006.

[9] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow Clustering Using Machine Learning Techniques," in *Passive and Active Measurement Conference (PAM)*, Antibes Juan-les-Pins, France, 2004.

[10] S. Zander, T.T.T. Nguyen, and G. Armitage, "Automated Traffic Classification and Application Identification using Machine Learning," in *IEEE Conference on Local Computer Networks (LCN)*, Sydney, Australia, 2005.

[11] Andrew Moore and Konstantina Papagiannaki, "Toward the Accurate Identification of Network Applications," in *PAM: Proceedings of the Passive and Active Measurement Workshop*, 2005.

[12] Arthur Callado, Carlos Kamienski Géza Szabó, Balázs P. Gero, Judith Kelner, Stenio Fernandes, and Djamel Sadok, "A Survey on Internet Traffic Identification," *IEEE Communications Surveys & Tutorials*, , no. 3, July 2009.

[13] H. Kim, kc claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K.Y. Lee, "Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices," in *Proceedings of the ACM CoNEXT Conference*, 2008.

[14] T.T.T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[15] S. Jordan, "Four Questions that Determine whether Traffic Management is Reasonable," in *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2009.

[16] U.M. Lewen, "Internet File-Sharing: Swedish Pirates Challenge the US," *Cardozo J. Int'l & Comp. L.*, vol. 16, pp. 173, 2008.

[17] Z.G. O'Leary, "Flying the Pirate Flag: Understanding the Fight Against and Prevalence of the Internet Gift Economy," *Pell Scholars Honors Theses*, p. 37, 2009.

[18] S. Jordan, "Implications of Internet architecture on net neutrality," *ACM Transactions on Internet Technology*, vol. 9, no. 2, 2009.

[19] K. Mochalski and Schulze H., "Deep Packet Insepction: Technology, Applications & Net Neutrality," Tech. Rep., IPOQUE, 2009, White Paper.

[20] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the KAZAA Network," in *WIAPP: 3rd IEEE Workshop on Internet Applications*, 2003, pp. 112–120.

[21] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and kc claffy, "Transport Layer Identification of P2P Traffic," in *IMC: Proceedings of the 4th ACM Conference on Internet Measurement*, Taormina, Sicily, Italy, 2004.

[22] Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang, "Accurate, Scalable In-network Identification of P2P Traffic using Application Signatures," in *WWW: Proceedings of the 13th international Conference on World Wide Web*. 2004, pp. 512–521, ACM.

[23] T. Choi, C. Kim, SH Yoon, J. Park, B. Lee, H. Kim, H. Chung, and T. Jeong, "Content-aware Internet Application Traffic Measurement and Analysis," in *Proc. Network Operations and Management Symposium*, 2004.

[24] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: Multilevel Traffic Classification in the Dark," in *Proceedings of ACM SIGCOMM*, 2005, p. 240.

[25] "Bluecoat Application Monitoring," `http://www.bluecoat.com` (accessed 2009.11.22).

[26] "Sandvine Inc.," `http://www.sandvine.com/` (accessed 2009.11.22).

[27] J. Erman, A. Mahanti, M. Arlitt, and C. Williamson, "Identifying and Discriminating Between Web and Peer-to-peer Traffic in the Network Core," in *WWW: Proceedings of the 16th international Conference on World Wide Web*. ACM, 2007, p. 892.

[28] J.D. Case, M. Fedor, M.L. Schoffstall, and J. Davin, "Simple Network Management Protocol (SNMP)," RFC 1157 (Historic), 1990.

[29] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), 2004.

[30] B.Claise, "IPFIX Protocol Specification," IETF Internet Draft, `http://tools.ietf.org/html/draft-ietf-ipfix-protocol-21` (accessed 2009.11.22).

[31] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, "Dynamic Application-layer Protocol Analysis for Network Intrusion Detection," in *USENIX Security Symposium*, 2006.

[32] "Directive 2006/24/EC of the European Parliament and of the Council," 2006, `http://eur-lex.europa.eu/LexUriServ/site/en/oj/2006/l_105/l_10520060413en00540063.pdf` (accessed 2009.11.22).

[33] L. Pimenidis and E. Kosta, "The Impact of the Retention of Traffic and Location Data on the Internet User," *Datenschutz und Datensicherheit-DuD*, vol. 32, no. 2, pp. 92–97, 2008.

[34] P. Ohm, "The Rise and Fall of Invasive ISP Surveillance," Tech. Rep., University of Illinois Law Review, 2009, Univeristy of Colorado Law Legal Studies Research Paper No. 08-22 `http://ssrn.com/abstract=1261344` (accessed 2009.11.28).

[35] Steven McCanne and Van Jacobson, "The BSD Packet Filter: A New Architecture for User-level Packet Capture," in *USENIX Winter*, 1993, pp. 259–270.

[36] Endace, "DAG Network Monitoring Cards," `http://www.endace.com/our-products/dag-network-monitoring-cards/` (accessed 2009.11.22).

[37] Baek-Young Choi, Jaesung Park, and Zhi-Li Zhang, "Adaptive packet sampling for accurate and scalable flow measurement," *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, vol. 3, pp. 1448–1452 Vol.3, 29 Nov.-3 Dec. 2004.

[38] T.Zseby, M. Molina, N.Duffield, S.Niccolini, and F.Raspall, "Sampling and Filtering Techniques for IP Packet Selection," IETF Internet Draft, `http://www.ietf.org/internet-drafts/draft-ietf-psamp-sample-tech-10.txt`.

[39] N. Duffield, "Sampling for Passive Internet Measurement: A Review," *Statistical Science*, pp. 472–498, 2004.

[40] J. Mai, C.N. Chuah, A. Sridharan, T. Ye, and H. Zang, "Is Sampled Data Sufficient for Anomaly Detection?," in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*. ACM, 2006, p. 176.

[41] Jun Xu, Jinliang Fan, Mostafa H. Ammar, and Sue B. Moon, "Prefix-Preserving IP Address Anonymization: Measurement-Based Security Evaluation and a New Cryptography-Based Scheme," in *ICNP '02: Proceedings of the 10th IEEE International Conference on Network Protocols*, Washington, DC, USA, 2002, pp. 280–289.

[42] United Stated Departement of Health and Human Services (HHS), "CFR Title 45 Part 46: Protection of Human Subjects," 2005, http://www.hhs.gov/ohrp/humansubjects/guidance/45cfr46.htm (accessed 2009.11.27).

[43] "CAIDA Research - Institutional Review Boards (IRB) Approval Process," http://www.caida.org/home/about/irb/ (accessed 2009.11.27).

[44] Joel Apisdorf, k. claffy, Kevin Thompson, and Rick Wilder, "OC3MON: Flexible, Affordable, High Performance Staistics Collection," in *LISA: Proceedings of the 10th USENIX Conference on System administration*, Berkeley, CA, USA, 1996.

[45] Chuck Fraleigh, Sue Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and Christophe Diot, "Packet-Level Traffic Measurements from the Sprint IP Backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, 2003.

[46] kc claffy, Mark Crovella, Timur Friedman, Colleen Shannon, and Neil Spring, "Community-oriented Network Measurement Infrastructure (CONMI) Workshop Report," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 2, pp. 41–48, 2006.

[47] Vern Paxson, "Strategies for Sound Internet Measurement," in *IMC: Proceedings of the 4th ACM SIGCOMM Conference on Internet measurement*, 2004, pp. 263–271.

[48] SUNET, "History of the Swedisch University Computer Network," http://basun.sunet.se/karta/ (accessed 2009.11.28).

[49] SUNET, "The Swedisch University Computer Network OptoSUNET," http://basun.sunet.se/aktuellt/optosunetbroschyr_eng.pdf (accessed 2009.11.28).

[50] SUNET, "Interactive Map of GigaSUNET," 2006, http://stats.sunet.se/stat-q/load-map/gigasunet-map,2006-12-01,traffic,peak (accessed 2009.11.28).

[51] "Chalmers Computer Communication Group," http://www.cdg.chalmers.se/ (accessed 2009.01.20).

[52] "Gothenburg University," http://www.gu.se/ (accessed 2009.01.20).

[53] "GSIX Student Network Aggregation," http://www.gsix.se/ (accessed 2009.01.20).

[54] "Halmstad University," http://www.hh.se/english (accessed 2009.01.20).

[55] "Netnod Swedish Internet Exchange," http://www.netnod.se (accessed 2009.11.28).

[56] SUNET, "Interactive Map of OptoSUNET," http://stats.sunet.se/stat-q/load-map/optosunet-core,,traffic,peak (accessed 2009.11.28).

[57] Tønnes Brekne and André Årnes, "Circumventing IP-address pseudonymization," in *Proceedings of the Third IASTED International Conference on Communications and Computer Networks*, Marina del Rey, CA, USA, 2005.

[58] T. Brekne, A. Arnes, and A. Oslebo, "Anonymization of IP Traffic Monitoring Data: Attacks on Two Prefix-Preserving Anonymization Schemes and Some Proposed Remedies," *Lecture Notes in Computer Science*, vol. 3856, pp. 179, 2006.

[59] Wolfgang John and Sven Tafvelin, "SUNET OC 192 Traces (Collection)," http://imdc.datcat.org/collection/1-04L9-9=SUNET+OC+192+Traces (accessed 2009.11.22).

[60] k.c. claffy K.C., H.-W. Braun, and G.C. Polyzos, "A Parameterizable Methodology for Internet Traffic Flow Profiling," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 8, 1995.

[61] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, "Inferring TCP Connection Characteristics Through Passive Measurements," in *IEEE INFOCOM*, 2004, vol. 3, pp. 1582–1592.

[62] V. Paxson, "End-to-end Routing Behavior in the Internet," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 56, 2006.

[63] B.Y. Choi, S. Moon, Z.L. Zhang, K. Papagiannaki, and C. Diot, "Analysis of Point-to-point Packet Delay in an Operational Network," *Computer Networks*, vol. 51, no. 13, pp. 3812–3827, 2007.

[64] "CAIDA Internet Traffic Classification," http://www.caida.org/research/traffic-analysis/classification-overview/ (accessed 2009.11.22).

[65] D. Moore, C. Shannon, D.J. Brown, G.M. Voelker, and S. Savage, "Inferring Internet Denial-of-service Activity," *ACM Transactions on Computer Systems (TOCS)*, vol. 24, no. 2, pp. 139, 2006.

[66] Ruoming Pang, Vinod Yegneswaran, Paul Barford, Vern Paxson, and Larry Peterson, "Characteristics of Internet Background Radiation," in *ACM Internet Measurement Conference (IMC)*, Taormina, Sicily, Italy, 2004.

[67] "NLANR Passive Measurement and Analysis Project," `http://pma.nlanr.net/` (accessed 2009.11.22).

[68] H. Jiang and C. Dovrolis, "Passive Estimation of TCP Round-trip Times," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 3, pp. 75–88, 2002.

[69] K. Lan and J. Heidemann, "On the Correlation of Internet Flow Characteristics," *USC/ISI, Tech. Rep. ISI-TR-574*, 2003.

[70] K. Lan and J. Heidemann, "A Measurement Study of Correlations of Internet Flow Characteristics," *Computer Networks*, vol. 50, no. 1, pp. 46–62, 2006.

[71] Kostas Pentikousis and Hussein Badr, "Quantifying the Deployment of TCP Options - a Comparative Study," *IEEE Communications Letters*, vol. 8, no. 10, pp. 647–9, 2004.

[72] "The CAIDA Web Site," `http://www.caida.org/` (accessed 2009.11.22).

[73] "List of Papers by CAIDA," `http://www.caida.org/publications/papers/bytopic/` (accessed 2009.11.22).

[74] T. Karagiannis, A. Broido, N. Brownlee, k.c. claffy, and M. Faloutsos, "Is P2P Dying or Just Hiding?," in *GLOBECOM: IEEE Global Telecommunications Conference*, Dallas, TX, USA, 2004, vol. Vol.3, pp. 1532 – 8.

[75] Marina Fomenkov, Ken Keys, David Moore, and k claffy, "Longitudinal Study of Internet Traffic in 1998-2003," in *WISICT: Proceedings of the winter international synposium on Information and communication technologies*, 2004.

[76] Sean McCreary and kc claffy, "Trends in Wide Area IP Traffic Patterns - A View from AMES Internet Exchange," Tech. Rep., CAIDA, San Diego Supercomputer Center, 2000.

[77] Colleen Shannon, David Moore, and kc claffy, "Beyond Folklore: Observations on Fragmented Traffic," *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 709–20, 2002.

[78] "The CAIDA Tools site," `http://www.caida.org/tools/` (accessed 2009.11.22).

[79] Ken Keys, David Moore, Ryan Koga, Edouard Lagache, Michael Tesch, and k claffy, "The Architecture of CoralReef: an Internet Traffic Monitoring Software Suite," in *PAM: A Workshop on Passive and Active Measurements*, 2001.

[80] Colleen Shannon, David Moore, Ken Keys, Marina Fomenkov, Bradley Huffaker, and k claffy, "The Internet Measurement Data Catalog," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 97–100, 2005.

[81] "The Internet Traffic Archive," `http://ita.ee.lbl.gov/` (accessed 2009.11.28).

[82] Greg Minshall, "TCPDPRIV: Program for Eliminating Confidential Information from Traces," `http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html`.

[83] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson, "On the Self-similar Nature of Ethernet Traffic (Extended Version)," *IEEE/ACM Transactions on Networking (ToN)*, vol. 2, no. 1, pp. 1–15, 1994.

[84] Vern Paxson and Sally Floyd, "Wide-area Traffic: the Failure of Poisson Modeling," *SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 257–268, 1994.

[85] V. Paxson, "Growth Trends in Wide-area TCP Connections," *IEEE Network*, vol. 8, no. 4, pp. 8–17, 1994.

[86] Vern Paxson, "Empirically Derived Analytic Models of Wide-area TCP Connections," *IEEE/ACM Trans. Netw.*, vol. 2, no. 4, pp. 316–336, 1994.

[87] Ruoming Pang, Mark Allman, Vern Paxson, and Jason Lee, "The Devil and Packet Trace Anonymization," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 29–38, 2006.

[88] "LBNL/ICSI Enterprice Tracing Project," http://www.icir.org/enterprise-tracing/download.html (accessed 2009.11.22).

[89] Ruoming Pang, Mark Allman, Mike Bennett, Jason Lee, Vern Paxson, and Brian Tierney, "A First Look at Modern Enterprise Traffic," in *IMC: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement*, Berkeley, CA, USA, 2005, pp. 2–2, USENIX Association.

[90] S. Coull, C. Wright, F. Monrose, M. Collins, and M. Reiter, "Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces," in *Proceedings of the Network and Distributed Systems Security Symposium*, San Diego, CA, USA, 2007.

[91] Mark Allman and Vern Paxson, "Issues and Etiquette Concerning Use of Shared Measurement Data," in *IMC: Proceedings of the 7th ACM SIGCOMM Conference on Internet measurement*, 2007, pp. 135–140.

[92] Erin E. Kenneally and kc claffy, "An Internet Data Sharing Framework For Balancing Privacy and Utility," in *Proceedings of Engaging Data: First International Forum on the Application and Management of Personal Electronic Information*, 2009.

[93] "WAND Network Research Group," http://www.wand.net.nz/ (accessed 2009.11.22).

[94] "WITS: Waikato Internet Traffic Storage," http://www.wand.net.nz/wits/ (accessed 2009.11.22).

[95] Richard Nelson, Daniel Lawson, and Perry Lorier, "Analysis of Long Duration Traces," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 1, pp. 45–52, 2005.

[96] "The WIDE Project," http://www.wide.ad.jp/ (accessed 2009.11.22).

[97] "Packet Traces from WIDE Backbone," http://tracer.csl.sony.co.jp/mawi/ (accessed 2009.11.22).

[98] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho, "Seven Years and One Day: Sketching the Evolution of Internet Traffic," in *Proceedings of INFOCOM*, 2009.

[99] "The IST SCAMPI Project," http://www.ist-scampi.org/ (accessed 2009.11.22).

[100] "The IST LOBSTER Project," http://www.ist-lobster.org/ (accessed 2009.11.22).

[101] "The Stager Visualization Package," http://software.uninett.no/stager/ (accessed 2009.11.22).

[102] "LOBSTER Attack Traces," http://lobster.ics.forth.gr/traces/ (accessed 2009.11.22).

[103] M. Polychronakis, K.G. Anagnostakis, and E.P. Markatos, "Emulation-Based Detection of Non-self-contained Polymorphic Shellcode," in *RAID: 10th Int. Symposium on Recent advances in intrusion detection*, 2007.

[104] D. Koukis, S. Antonatos, D. Antoniades, E.P. Markatos, and P. Trimintzios, "A Generic Anonymization Framework for Network Traffic," *ICC: IEEE International Conference on Communications*, vol. 5, June 2006.

[105] "The MOMENT Project," http://www.fp7-moment.eu/ (accessed 2009.11.22).

[106] "ETOMIC: European Traffic Observatory Measurement Infrastructure," http://www.etomic.org (accessed 2009.11.22).

[107] "The DIMES Project," http://www.netdimes.org (accessed 2009.11.22).

[108] "Sprint IP Data Analysis Trace Collection Overview," http://ipmon.sprint.com/packstat/packetoverview.php (accessed 2009.11.22).

[109] K. To, T. Ye, and S. Bhattacharyya, "CMON: A General-purpose Continuous IP Backbone Traffic Analysis Platform," Tech. Rep., Sprint ATL, 2004, Research Report RR04-ATL-110309.

[110] A. Sridharan, T. Ye, and S. Bhattacharyya, "Connectionless Port Scan Detection on the Backbone," in *IPCCC: 25th IEEE International Performance, Computing, and Communications Conference*, 2006, p. 10.

[111] Avinash Sridharan and Tao Ye, "Tracking Port Scanners on the IP Backbone," in *LSAD '07: Proceedings of the 2007 Workshop on Large scale attack defense*, 2007, pp. 137–144.

[112] Anja Feldmann, Albert Greenberg, Carsten Lund, Nick Reingold, and Jennifer Rexford, "NetScope: Traffic Engineering for IP Networks," *IEEE Network*, vol. 14, 2000.

[113] R. Cáceres, N. Duffield, A. Feldmann, J.D. Friedmann, A. Greenberg, R. Greer, T. Johnson, C.R. Kalmanek, B. Krishnamurthy, D. Lavelle, et al., "Measurement and Analysis of IP Network Usage and Behavior," *IEEE Communications Magazine*, vol. 38, no. 5, pp. 144–151, 2000.

[114] Subhabrata Sen and Jia Wang, "Analyzing Peer-to-peer Traffic Across Large Networks," in *IMW: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, New York, NY, USA, 2002, pp. 137–150, ACM.

[115] Alexandre Gerber, Joseph Houle, Han Nguyen, Matthew Roughan, and Subhabrata Soensitivoen, "P2P The Gorilla in the Cable," Chicago, IL, USA, 2003, NCTA National Show.

[116] Feng Qian, Alexandre Gerber, Z. Morley Mao, Subhabrata Sen, Oliver Spatscheck, and Walter Willinger, "TCP Revisited: A Fresh Look at TCP in the Wild," in *IMC: Proceedings of the 9th ACM Internet Measurement Conference*, 2009.

[117] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the Characteristics and Origins of Internet Flow Rates," in *Proceedings of SIGCOMM*, 2002, pp. 309–322.

[118] G. Maier, A. Feldmann, V. Paxson, and M. Allman, "On Dominant Characteristics of Residential Broadband Internet Traffic," in *Proceeding of the 9th ACM SIGCOMM Conference on Internet Measurement*, 2009.

[119] M. Arlitt and C. Williamson, "An Analysis of TCP Reset Behaviour on the Internet," *Computer Communication Review*, vol. 35, no. 1, pp. 37–44, 2005.

[120] Andrew Moore, James Hall, Christian Kreibich, Euan Harris, and Ian Pratt, "Architecture of a Network Monitor," in *Passive and Active Measurement Workshop (PAM)*, 2003.

[121] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic Classification Through Simple Statistical Fingerprinting," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 16, 2007.

[122] M. Dusi, A. Este, F. Gringoli, and L. Salgarelli, "Using GMM and SVM-based Techniques for the Classification of SSH-Encrypted Traffic," in *ICC: Proceedings of the 44rd IEEE International Conference on Communications*, 2009.

[123] Alberto Medina, Mark Allman, and Sally Floyd, "Measuring the Evolution of Transport Protocols in the Internet," *Computer Communication Review*, vol. 35, no. 2, pp. 37–51, 2005.

[124] M. Perenyi, Dang Trang Dinh, A. Gefferth, and S. Molnar, "Identification and Analysis of Peer-to-peer Traffic," *Journal of Communications*, vol. 1, no. 7, pp. 36–46, 2006.

[125] Craig Labovitz, Danny McPherson, and Scott Iekel-Johnson, "2009 Internet Observatory Report," in *NANOG 47*, 2009.

[126] "CAIDA Internet Data - Passive Data Sources," `http://www.caida.org/data/passive/` (accessed 2009.11.22).

[127] CAIDA, "DatCat: Internet Measurement Data Catalog," `http://imdc.datcat.org/` (accessed 2009.12.01).

[128] "PREDICT: Protected Repository for the Defense of Infrastructure against Cyber Threats," `https://www.predict.org/` (accessed 2009.12.01).

[129] Abhinav Parate and Gerome Miklau, "A Framework for Safely Publishing Communication Traces," in *CIKM: Conference on Information and Knowledge Management*, 2009.

[130] J.C. Mogul and M. Arlitt, "SC2D: an Alternative to Trace Anonymization," in *Proceedings of the SIGCOMM Workshop on Mining network data*, 2006, p. 328.

[131] J. Mirkovic, "Privacy-safe Network Trace Sharing via Secure Queries," in *Proceedings of the 1st ACM Workshop on Network data anonymization*, 2008.

[132] P. Mittal, V. Paxson, R. Sommer, and M. Winterrowd, "Securing Mediated Trace Access Using Black-box Permutation Analysis," in *HOTNETS*, 2009.

[133] G. Xie, G. Zhang, J. Yang, Y. Min, V. Issarny, and A. Conte, "Survey on Traffic of Metro Area Network with Measurement On-Line," *Lecture Notes in Computer Science*, vol. 4516, pp. 666, 2007.

[134] J. Postel, "RFC 879: TCP Maximum Segment Size and Related Topics," 1983.

[135] T. Karagiannis, A. Broido, N. Brownlee, k. claffy, and M. Faloutsos, "File-sharing in the Internet: A Characterization of P2P Traffic in the Backbone," *University of California, Riverside, USA, Tech. Rep*, 2003.

[136] M. Handley, V. Paxson, and C. Kreibich, "Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-end Protocol Semantics," in *Proceedings of the 10th conference on USENIX Security Symposium*, 2001, p. 9.

[137] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-time," *Comput. Networks*, vol. 31, no. 23, pp. 2435–2463, 1999.

[138] Pan Pan, Yi Cui, and Bo Liu, "A Measurement Study on Video Acceleration Service," in *IEEE CCNC*, 2009.

[139] Wikipedia.org, "Micro Transport Protocol," Online: "http://en.wikipedia.org/wiki/Micro_Transport_Protocol", accessed April 29, 2009.

[140] H. Schulze and K. Mochalski, "IPOQUE Internet Study 2007," 2007, `http://www.ipoque.com/resources/internet-studies/internet-study-2007` (accessed 2009.11.27).

[141] "Directive 2004/48/EC of the European Parliament and of the Council," 2004, `http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2004:157:0045:0086:EN:PDF` (accessed 2010.01.18).

[142] "The Pirate Bay," `http://thepiratebay.org/` (accessed 2009.01.20).

[143] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "ACAS: Automated Construction of Application Signatures," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Mining network data*, 2005, p. 202.

[144] J. Ma, K. Levchenko, C. Kreibich, S. Savage, and G.M. Voelker, "Unexpected Means of Protocol Inference," in *Proceedings of the 6th ACM SIGCOMM Conference on Internet measurement*, 2006, pp. 313–326.

[145] "Message Stream Enryption (MSE)," `http://azureuswiki.com/index.php/Message_Stream_Encryption` (accessed 2009.11.28).

[146] "eMule Protocol Obfuscation," `http://wiki.emule-web.de/index.php/Protocol_obfuscation` (accessed 2009.11.28).

[147] S.A. Baset and H. Schulzrinne, "An Analysis of the Skype Peer-to-peer Internet Telephony Protocol," in *IEEE infocom*, 2006, vol. 6.

[148] G. Szabo, D. Orincsay, S. Malomsoky, and I. Szabo, "On the Validation of Traffic Classification Algorithms," in *PAM: Proceedings of the Passive and Active Measurement Workshop*, 2008.

[149] "Spotify," `http://www.spotify.com/en/` (accessed 2009.11.28).

[150] T. Henderson, "Sharing is Caring: so Where are Your Data?," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 1, pp. 43–44, 2008.

Part II

APPENDED PAPERS

PAPER I

Wolfgang John, Sven Tafvelin and Tomas Olovsson

Passive Internet Measurement: Overview and Guidelines based on Experiences

Computer Communications
Vol. 33(5), Elsevier, 2010

Passive Internet Measurement: Overview and Guidelines based on Experiences

Wolfgang John, Sven Tafvelin and Tomas Olovsson

Department of Computer Science and Engineering
Chalmers University of Technology, Göteborg, Sweden
{firstname.lastname}@chalmers.se

Abstract

Due to its versatility, flexibility and fast development, the modern Internet is far from being well understood in its entirety. A good way to learn more about how the Internet functions is to collect and analyze real Internet traffic. This paper addresses several major challenges of Internet traffic monitoring, which is a prerequisite for performing traffic analysis. The issues discussed will eventually appear when planning to conduct passive measurements on high-speed network connections, such as Internet backbone links. After giving a brief summary of general network measurement approaches, a detailed overview of different design options and important considerations for backbone measurements is given. The challenges are discussed in order of their chronological appearance: First, a number of legal and ethical issues have to be sorted out with legislators and network operators, followed by operational difficulties that need to be solved. Once these legal and operational obstacles have been overcome, a third challenge is given by various technical difficulties when actually measuring high-speed links. Technical issues range from handling the vast amounts of network data to timing and synchronization issues. Policies regarding public availability of network data need to be established once data is successfully collected. Finally, a successful Internet measurement project is described by addressing the aforementioned issues, providing concrete lessons learned based on experiences. As a result, the paper presents tutorial guidelines for setting up and performing passive Internet measurements.

1 Introduction

The usage of the Internet has changed dramatically since its initial operation in the early-80s, when it was a research project connecting a handful of computers, facilitating a small set of remote operations. Today (2009), the Internet serves as the data backbone for all kinds of protocols, making it possible to exchange not only text, but also voice, audio, video and various other forms of digital data between hundreds of millions of nodes, ranging from traditional desktop computers, servers or supercomputers to all kinds of wireless devices, embedded systems, sensors and even home equipment.

Traditionally, an illustration of the protocol layers of the Internet has the shape of an hourglass, with a single Internet Protocol (IP) on the central network layer and an increasingly wider spectrum of protocols above and below. Since the introduction of IP in 1981, which is basically still unchanged, technology and protocols have developed significantly. Underlying transmission media evolved from copper to fiber optics and WIFI, routers and switches became more and more intelligent and are able to handle Gbit/s instead of Kbit/s and additional middleware boxes have been introduced (e.g., NAT and firewalls). But also above the network layer new applications have constantly been added, ranging from basic services such as DNS and HTTP, to recent, complex P2P protocols allowing applications such as file-sharing, video streaming and telephony. With IPv6, even the foundation of the Internet is finally about to be substituted. This multiplicity of protocols and technologies leads to an ongoing increase in complexity of the Internet as a whole. Of course, individual network protocols and infrastructures are usually well understood when tested in isolated lab environments or network simulations. However, their behavior when observed while interacting with the vast diversity of applications and technologies in the hostile Internet environment is often unclear, especially on global scale.

This lack of understanding is further amplified by the fact that the topology of the Internet was not planned in advance. It is the result of an uncontrolled extension process, where heterogeneous networks of independent organizations have been connected one by one to the main Internet (*INTERconnected NETworks*). This means that each autonomous system (AS) has its own set of usage and pricing policies, QoS measures and resulting traffic mix. Thus usage of Internet protocols and applications is not only changing with time, but also with geographical locations. As an example, Nelson et al. [1] reported about an unusual application mix on a campus uplink in New Zealand due to a restrictive pricing policy, probably caused by higher prices for trans-pacific network capacities at this time.

Finally, higher connectivity bandwidths and growing numbers of Internet users lead to increased misuse and anomalous behavior [2]. Not only the numbers of malicious incidents keep rising, but also the level of sophistication of attack methods and tools has increased. Today, automated attack tools employ more and more advanced attack patterns and react on the deployment of firewalls and intrusion detection systems by clever obfuscation of their malicious intentions. Malicious activities range from scanning to more advanced attack types such as worms and various denial of service attacks. Even well-known or anticipated attack types reappear in modified variants, such as the recent renaissance of cache poisoning attacks [3]. Unfortunately, the Internet, initially meant to be a friendly place, eventually became a hostile environment that needs to be studied continuously in order to develop suitable counter strategies.

Overall, this means that even though the Internet may be considered to be the most important modern communication platform, its behavior is not well understood. It is therefore crucial that the Internet community understands the nature and detailed behavior of modern Internet traffic, in order to be able to improve network applications, protocols and devices and protect its users.

The best way to acquire a better and more detailed understanding of the modern Internet is to monitor and analyze real Internet traffic. Unfortunately, the above described rapid development has left little time or resources to integrate measurement and analysis possibilities into

Internet infrastructure, applications and protocols. To compensate for this lack, the research community has started to launch dedicated Internet measurement projects, usually associated with considerable investment of both time and money. However, the experiences from a successful measurement project showed that measuring large-scale Internet traffic is not simple and involves a number of challenging tasks. In order to help future measurement projects to save some of their initial time expenses, this paper gives an overview of the major challenges which will eventually appear when planning to conduct measurements on high-speed network connections. Experiences from the MonNet project will then provide guidelines based on lessons learned (Section 8).

1.1 How to read this paper

Section 2 gives an overview of different network traffic measurement approaches and methodologies. Sections 3-7 address the main challenges encountered while conducting passive Internet measurements. The challenges are discussed in order of their chronological appearance: First, a number of legal and ethical issues have to be sorted out with legislators and network operators before data collection can be started (Sections 3 and 4). Second, operational difficulties need to be solved (Section 5) such as access privileges to the network operator's premises. Once legal and operational obstacles are overcome, a third challenge is given by various technical difficulties when actually measuring high-speed links (Section 6), ranging from handling of vast data amounts to timing issues. Next public availability of network data are discussed, which should eventually be considered once data are successfully collected (Section 7). Section 8 then outlines the MonNet project, which is the measurement project providing the experience for the present paper. Each point from Sections 3 - 7 will be revisited and the specific problems and solutions as experienced in the MonNet project are presented. These considerations are then summarized presenting the most important lessons learned in each particular section, providing a quick guide for future measurement projects. Finally, Section 9 discusses future challenges of Internet measurement and concludes the paper.

2 Overview of network measurement methodologies

This section gives an overview of general network measurement approaches. The basic approaches are categorized among different axes and the most suitable methods for passive Internet measurements according to current best practice are pointed out.

The most common way to classify traffic measurement methods is to distinguish between **active** and **passive** approaches. Active measurement involves injection of traffic into the network in order to probe certain network devices (e.g., PING) or to measure network properties such as round-trip-times (RTT) (e.g., traceroute), one-way delay and maximum bandwidth. Pure observation of network traffic, referred to as passive measurement or monitoring, is non-intrusive and does not change the existing traffic. Network traffic is tapped at a specific location and can then be recorded and processed at different levels of granularity, from complete packet-level traces to statistical figures. Even if active measurement offers some possibilities that passive approaches cannot provide, in this paper only passive measurement is considered, which is best suitable for analysis of Internet backbone traffic properties.

Passive traffic measurement methods can be further divided into **software-** and **hardware-based** approaches. Software-based tools modify operating systems and device drivers on network hosts in order to obtain copies of network packets (e.g., BSD packet filter [4]). While this approach is inexpensive and offers good adaptability, its possibilities to measure traffic on high-speed networks are limited [5]. In contrast, hardware-based methods are designed specifically for collection and processing of network traffic on high-speed links such as an Internet backbone. Special traffic acquisition hardware collects traffic directly on the physical links (e.g., by using optical splitters) or on network interfaces (e.g., mirrored router ports). Since highly specialized, such equipment is rather expensive and offers limited versatility.

Once network data are collected, it needs to be processed to fulfill its particular purpose.Traffic processing can be done **online**, **offline** or in a combination of both approaches. Online processing refers to immediate processing of network data in "real time", which is essential for applications such as traffic filters or intrusion detection systems. Sometimes only parts of the data processing are done online, as typically done when collecting condensed traffic statistics or flow-level summaries. Offline processing on the other hand is performed on network data after it is stored on a data medium. Offline processing is not time critical and offers the possibility to correlate network traffic collected at different times or different locations. Furthermore, stored network data can be re-analyzed with different perspectives over and over again. These advantages make offline processing a good choice for complex and time consuming Internet analysis.

Internet measurement can furthermore operate on different **protocol layers**, following the Internet reference model [6]. While link-layer protocols dictate the technology used for the data collection (e.g., SONET/HDLC, Ethernet), one of the most studied protocols is naturally the Internet Protocol (IP), located on the network layer. The Internet measurement community also shows great interest in the analysis of transport layer protocols, especially TCP and UDP. Some Internet measurement projects have the possibilities to study all layers, including application layer protocols. In practice, most measurement projects consider mainly network and transport layer protocols due to privacy and legal concerns, as discussed later (Sections 3 and 4)

Data gathered on different protocol layers can present different levels of granularity. The most coarse granularity is provided by cumulated **traffic summaries and statistics**, such as packet counts or data volumes, as typically provided by SNMP [7]. Another common practice is to condense network data into **network flows**. A flow can be described as a sequence of packets exchanged between common endpoints, defined by certain fields within network and transport headers. Instead of recording each individual packet, flow records are stored, containing relevant information about the specific flow. Such flow records can be unidirectional, as in the case of NetFlow [8], or bidirectional, as used in different studies by MonNet [9, 10, 11]. The finest grained level of granularity is provided by **packet-level traces**. Packet-level traces can include all information of each packet observed on a specific host or link. While such **complete packet-level traces** offer the best analysis possibilities, they come along with a number of technical and legal issues, as discussed in Chapters 3 - 6. It is therefore common practice to reduce the stored information to packet headers up to a certain protocol level, e.g., including network and transport protocols only, as done for the MonNet traces.

Finally, packet-level network traces can be stored in different trace formats. Unfortunately, there is no standardized trace format, so developers of trace collection tools historically defined their own trace formats. The most popular trace format, especially common for traces from local area networks (LANs), is the **PCAP format**, the format of the BSD Packet Filter and TCPdump. For traces of wide area networks (WANs), an often used format was defined by Endace, the Endace record format (ERF), formerly also known as **DAG format**. Other trace formats seen in the Internet measurement community include CAIDA's CORALReef format CRL [12] or NLANR's formats FR, FR+ and TSH. This diverseness in trace formats introduces some problems, since publicly available analysis tools usually do not recognize all of these formats, making conversion of traces from one format to another necessary. Since PCAP can be seen as the de-facto standard, almost all conversion tools are able to convert their own format to or from this format. Conversion, however, is usually not without cost. Different timestamp conventions within the trace formats often lead to loss of timestamp precision, which should be considered when performing timing sensitive operations.

3 Legal background

In this section the legal background of Internet measurement is presented, which is somewhat in contrast to actual political developments and common academic practice. Current laws and regulations on electronic communication rarely explicitly consider or mention the recording or measurement of traffic for research purposes, which leaves scientific Internet measurement in some kind of legal limbo. In the following paragraphs the existing regulations for the EU and the US are briefly outlined in order to illustrate the legal complications network research is struggling with.

3.1 European Union (EU) directives

Privacy and protection of personal data in electronic communication in EU countries are regulated by the *Directive 95/46/EC on the protection of personal data* [13] of 1995 and the complementing *Directive 2002/58/EC on Privacy and Electronic Communications* [14] of 2002. Data retention regulations have recently been further amended with the *Directive 2006/24/EC on the retention of data generated or processed in electronic communication* [15].

The Data protection directive (Directive 95/46/EC) defines personal data in Article 2a as *"any information relating to an identified or identifiable natural person (data subject)"*. Besides names, addresses or credit card numbers, this definition thereby also includes email and IP addresses. Furthermore, data are defined as personal as soon as someone can potentially link the information to a person, where this someone not necessarily needs to be the one possessing the data. Processing of personal data are then defined in Article 2b as *"any operation or set of operations which is performed upon personal data, whether or not by automatic means, such as ... collection, recording, ...storage, ..."*, which means that Internet traffic measurement clearly falls into the scope of this directive. Summarized, Directive 95/46/EC defines conditions under which the processing of personal data are lawful. Data processing is e.g., legitimate with consent of the user, for a task of public interest or for compliance with legal obligations (Ar-

ticle 7). Further conditions include the users (or "data subjects") right for transparency of the data processing activities (Articles 10 and 11), the user's right of access to own personal data (Article 12) and principles relating to data quality (Article 6). The latter describes that data are only allowed to be processed for specified, explicit and legitimate purposes. However, further processing or storage of personal data for historical, statistical or scientific purposes is not incompatible with these conditions, as long as appropriate safeguards for this data are provided by individual member states.

The e-privacy directive (Directive 2002/58/EC) complements the data protection directive of 1995, targeting matters which have not been covered earlier. The main subject of this directive is *"the protection of privacy in the electronic communication sector"*, which was required to be updated in order to react on requirements of the fast changing digital age. In contrast to the data protection directive, the e-privacy directive is not only applied to natural but also to legal persons. Besides dealing with issues like treatment of spam or cookies, this directive also includes regulations concerning confidentiality of information and treatment of traffic data. Some of the regulations are especially relevant for Internet measurement. Specifically, Article 5 states that *"listening, tapping, storage or other kinds of interception or surveillance of communications and the related traffic data by persons other than users"* are prohibited, with the exception of given consent by the user or the necessity of measures in order *"to safeguard national security, defense, public security, and the prevention, investigation, detection and prosecution of criminal offenses"* (Article 15(1)). Furthermore, Article 6(1) obliges service providers to erase or anonymize traffic data when no longer needed for transmission or other technical purposes (e.g., billing, provision, etc.), again with the only exception of national security issues (Article 15(1)).

The data retention directive (Directive 2006/24/EC) was among others a reaction on recent terrorist attacks (i.e., July 2005 in London), requiring communication providers to retain connection data for a period of between 6 months and 2 years *"for the purpose of the investigation, detection and prosecution of serious crime""* (Article 1). When this directive was released in March 2006, only 3 EU countries had legal data retention in force. The remaining countries declared to postpone application of this directive regarding Internet access, Internet telephony and Internet email, which was possible until 14 March 2009 according to Article 15(3). At present (October 2009) 22 of the 27 EU countries have transposed the directive (at least partial, as in the case of Luxembourg and the UK) by implementing the different national laws. The remaining five countries (i.e., Austria, Greece, Ireland, Poland, and Sweden) have yet failed to install national laws following the directive. An updated overview of national data retention policies and laws can be found online at "Vorratsdatenspeicherung.de" [16].

For current measurement projects in EU countries these directives basically say that Internet traffic measurement for scientific purposes requires user consent, since such projects are not subject of national security. User content could e.g., be obtained by adding a suitable passage to the "Terms of Service" signed by network users. Additionally, any individual member state has the possibility to permit Internet measurement for scientific purposes if appropriate safeguards are provided. With the introduction of the data retention directive, providers are legally required to store connection data. However, in order to be able to actually execute this directive, a number of technical challenges need to be solved first (Section 6). Experiences and lessons learned from

scientific Internet measurement projects are therefore vital and further underline the relevance of Internet measurement.

3.2 United States (US) laws

In contrast to the EU, privacy in the US is handled by a patchwork of case law, state and federal industry-specific laws [17]. The overview of US privacy laws in the present paper will follow a recent article by Sicker et al. [18], thereby focusing on federal laws of the US only (as opposed to state laws), especially since they are probably best compared to the overarching EU directives. There are two relevant sets of federal US laws applying to Internet measurement: one for real-time monitoring, and one for access to stored data.

When monitoring network traffic in real time, US laws distinguish between monitoring of user content and non-content such as header data. Real-time content monitoring is regulated by the *Wiretap Act* (18 U.S.C. §2511 [19]), basically stating that interception of communications is prohibited. There are, however, some exceptions to this basic rule, including user consent of at least one party of the communication as well as the providers' right to protect their networks and to help tracking culprits. Real-time monitoring of non-content (i.e., header data) was unregulated in the US until 2001, when the 9/11 attacks lead to the USA PATRIOT Act. This law amended the *Pen Register and Trap and Trace Act* (18 U.S.C. §3127 [20]) in order to apply it to recording or capturing of *"dialing, routing, addressing, or signaling information"* in context of electronic communications, which clearly includes non-content such as packet headers and IP address information. Consequently, also recording of packet header traces is prohibited in the US since 2001. Again, user consent and provider monitoring are exceptions stated in the act.

Access to stored network data, i.e., sharing of data traces, is in US federal laws regulated by the *Electronic Communications Privacy Act* (18 U.S.C. §2701-§2703 [21, 22, 23]). Basically, it is prohibited for network providers to give away stored records of network activity, regardless whether or not they include user content. Besides the exception of user consent there are two further exceptions to this basic rule. First, this rule does not apply to non-public providers, which means that data collected at private companies or organizations can be shared with other organizations or researchers. Second, non-content records (e.g., header traces) can be shared with anyone, with exception of the government. This leaves some uncertainty about the definition of "government entities", since scientific projects and researchers might be funded or co-sponsored by governmental money.

3.3 Scientific practice

For researchers it is not always obvious which regulations are in force. The borders between private and public networks as well as the difference between signaling or header data and user content is sometimes blurred and fuzzy, which makes it difficult to relate to the correct piece of law. This is especially true for amateurs in juristic matters, such as typical network scientists. Common privacy protection measures have been surveyed on datasets used in 57 recent Internet measurement related articles in [18], showing that a majority of network traces were collected on public networks and stored as packet headers only. Discussions about trace

anonymization or the difference between content and non-content was brought up in very few articles, probably due to page restrictions. However, it can be assumed that most researchers are aware of their responsibility towards the users and are anxious about privacy concerns, as described in Section 4.

As pointed out by Sicker et al. [18], often there is a *"disconnect between the law and current academic practice"*. Even though leading researchers try to close the gap between Internet researchers and lawyers by clarifying common misconceptions about the Internet [24], laws are not likely to be changed in favor of scientific Internet measurement anytime soon. According to Sicker at al. [18], a first important step towards de-criminalization of Internet measurement could be a community-wide consensus about privacy-protecting strategies formulated in a public document. Furthermore, the authors present some basic strategies for protecting user privacy, ranging from the often impossible task of getting user consent (e.g., signed "Terms of Service") to traditional de-sensitization techniques such as anonymization and data reduction (see Sections 4 and 6.2). The network researcher's motto should first of all be: *Do no Harm!* Even though researchers might sometimes unavoidably operate in legal grey zones, it is likely that no legal prosecution will be started as long as careful measures to avoid privacy violations following "common sense" have been taken and no harm has been done.

In a recent paper, Kenneally and claffy go one step further and propose the Privacy-Sensitive Sharing framework (PS2) [17], a framework supporting proactive management of privacy risks. The proposed model is a combination of a policy framework that satisfies obligations of both data seekers and data providers, and a technology framework able to enforce these obligations. As a result, PS2 should reveal that actual data sharing is less risky (in form of privacy risks) than *not* sharing data (and inability to understand and anticipate the Internet and its security threads), especially when considering the importance of modern Internet as an underlying, critical infrastructure for economical, professional, personal, and political life [25].

4 Ethical and moral considerations

Besides potential conflicts with legal regulations and directives, Internet measurement activities raise also moral and ethical questions when it comes to privacy and security concerns of individual users or organizations using the networks. These considerations include discussions about what to store, how long to store and in which ways to modify stored data. The goal is to fulfill privacy and security requirements of individuals and organizations, while still keeping scientific relevant information intact. Since network data can potentially compromise user privacy or reveal confidential network structures or activities of organizations, operators usually give permission to perform Internet measurement with at least one of the following restrictions:

1. *keep* raw measurement data *secret*;
2. *de-sensitize* the data, which can be done in one or both of the following ways:
   (a) *remove sensitive data (such as packet payload data)* in packet-level traces;
   (b) *anonymize or de-identify* packet traces and flow data.

De-sensitization refers to the process of removing sensitive information to ensure privacy and confidentiality. An example where un-desensitized measurement data are required would be network forensics conducted by governmental authorities. In this case data are kept secret, i.e., it is accessed by a limited number of trusted persons only. Within research projects, however, it is common that de-sensitization is required. Anonymization in this context refers to the process of removing or disguising information which reveals the real identity of communication entities. Some information, such as IP addresses, can be used to pinpoint individual users. This privacy threat makes IP address anonymization a common requirement even for measurements which are only kept internally, inside a network operator's organization.

The above stated de-sensitization actions, payload removal and anonymization, might satisfy both data providers (operators) and data seekers (researchers and developers) analyzing the data. There are, however, a number of detailed questions that are not necessarily answered by often imprecise and broadly stated policies. We discuss some important considerations below.

4.1 What to keep?

Even if it is decided to store packet header traces only, it is not always explicitly stated where user payload really starts. A common way to interpret "packet headers" is to keep IP and TCP (UDP) headers only, stripping off data after transport headers. However, one could argue that application headers are technically not user payload, and therefore could be kept as well. This may lead to problems in some cases (e.g., SMTP and HTTP headers), since a lot of sensitive information can be found there. Other application headers, such as SSH and HTTPS, violate no obvious privacy issues, assuming that IP address anonymization is done for all layers of packet headers. Furthermore, application headers introduce practical problems since the number of network applications is virtually infinite and not all applications use well defined headers. A solution is to store the first N bytes of the payload following transport protocols. Saving the initial bytes of packet payloads is sufficient for classifying traffic using signature matching (shown e.g., by Karagiannis et al.[26]) and offers a number of additional research possibilities, such as surveying frequency and type of packet encryption methods. Even if packets with privacy-sensitive application data (e.g., SMTP) would be treated differently and stored without any payload beyond transport layer, there is still a large degree of uncertainty left about how much sensitive information is included in unknown or undefined application payloads or malformed packets not recognizable for the processing application. This remaining uncertainty might be tolerable if traces are only accessed by a limited number of trusted researches, but is unsuitable for traces intended to become publicly available.

Even if the boundary between packet header and packet payload is clearly defined for most protocols (e.g., payload starts beyond transport layer), the researcher needs to decide how to treat unusual frames, not defined within most available trace processing tools. One such example is routing protocols such as CLNS routing updates (Connectionless Network Protocol) and CDP messages (Cisco Discovery Protocol). Even if routing information is not revealing privacy-sensitive data about individual users, it reveals important information about network layout and topology, which in turn can be important input to de-anonymization attacks. Another example are all kinds of unknown or malformed headers, which might not be recognized

by processing tools, but still contain sensitive information following malicious packet headers [27]. Policies for how to treat this kind of packets include:

1. packet truncation by default after a specified number of bytes;
2. packet dropping (which should be recorded in the meta-data of the specific trace);
3. keeping the un-truncated packet (which might bear security and privacy risks).

Finally, privacy of datasets can be improved by removing network data from hosts with unique, easy distinguishable behavior, as suggested by Coull et al. in [28]. Such hosts can include DNS servers, popular HTTP or SMTP servers or scanning hosts. Obviously, this approach leaves a biased view of network traffic, which might be unsuitable for certain research purposes. It is therefore crucial that removal or special treatment of packets from specially exposed hosts is well documented and commented in the descriptions or the meta-data of the respective network traces.

4.2 How to anonymize?

If anonymization of network traces is required, it still needs to be decided which header fields to anonymize and how. Generally, it should be noted that "*anonymization of packet traces is about managing risk*", as pointed out by Pang et al. [29]. Datasets from smaller, local networks might be more sensitive than data from highly aggregated backbone links when it comes to attacks trying to infer confidential information such as network topologies or identification of single hosts. Coull et al. [28] also showed that hardware addresses in link-layer headers can reveal confidential information, which is a problem for Ethernet-based measurements, but not for Internet measurement on backbone links. Furthermore, the age of the datasets being published plays an important role since the Internet has a very short-lived nature, and network architectures and IP addresses change frequently and are hard to trace back. Generally, anonymization is an important measure to face privacy concerns of users, even though it needs to be noted that all proposed anonymization methods have been shown to be breakable to a certain degree, given an attacker with sufficient know-how, creativity and persistency [28, 30, 31, 32]. This was stated nicely by Allman and Paxson in [33], when saying that publisher of network traces "*are releasing more information than they think*"!

Currently, the most common practice to anonymize packet headers is to anonymize IP address information only, which is often sufficient for internal use (i.e., only results, but no datasets will be published). As discussed above, in some situations when traces are planned to be published, a more complete method is required, offering the possibility to modify each header and payload field with individual methods, including email addresses, URLs and usernames/passwords. Such a framework is publicly available and described by Pang et al. in [29]. However, how different fields are modified has to be decided by the researcher or agreed upon in anonymization policies. The increasing importance of data anonymization for the Internet

measurement community has recently resulted in the organization of a dedicated workshop on Network data anonymization [34], which sets out to advance theory and practice of anonymization as it applies to network data.

4.2.1 Anonymization methods

In the following paragraphs, we list and discuss some common methods to anonymize the most sensitive information in packet headers, namely IP addresses. IP address anonymization is here defined as the irreversible mapping between the real and the anonymized IP addresses.

1. *One constant:* The most simple method is to substitute all IP addresses with one constant, which collapses the entire IP address space to one single constant with no information content. A refined version of this method is to keep the first N bits of addresses unmodified, and replace the remaining bits with a constant (e.g., set them to zero).
2. *Random permutation:* Another rather simple method is random permutation, which creates a one-to-one mapping between real and anonymized addresses. This method is only irreversible given a proper secrecy concerning the permutation table. Furthermore the subnet information implicitly included in the real addresses is lost.
3. *Pseudonymization:* The idea of random permutation is very similar to a method called pseudonymization, where each IP address is mapped to a pseudonym, which might or might not have the form of a valid IP address. It is only important that a one-to-one mapping is provided.
4. *Prefix-preserving anonymization:* A special variation of pseudonymization has the property of preserving prefix information, and is therefore referred to as prefix-preserving anonymization. A prefix-preserving anonymization scheme needs to be impossible or at least very difficult to reverse while maintaining network and subnet information, which is crucial for a many different types of analysis.
   (a) *TCPdpriv:* The first popular prefix-preserving anonymization technique was used in TCPdpriv, developed by Minshall in 1996 [35]. The prefix preserving anonymization function of TCPdpriv applies a table-driven translation based on pairs of real and anonymized IP addresses. When new translations are required, existing pairs are searched for the longest prefix match. The first k bits matching the already translated prefix are then reused, and the remaining $32 - k$ bits are replaced with a pseudo-random number and the address is added to the table. The drawback of this approach is that the translations are inconsistent when used on different traces, since translation depends on the order of appearance of the IP addresses. This problem can be solved if translation tables are stored and reused. The approach, however, still leaves the problem that traces cannot be anonymized in parallel, which is desired practice when dealing with large volumes of Internet data.
   (b) *Crypto-PAn:* The drawback of TCPdpriv was fixed by a Cryptography-based Prefix-preserving Anonymization method, Crypto-PAn, described by Xu et al. in 2002 [30].

Crypto-PAn offers the same prefix-preserving features as TCPdpriv, with the additional advantage of allowing distributed and parallel anonymization of traces. Instead of a table-driven approach, Crypto-PAn establishes a deterministic one-to-one mapping by use of a key and a symmetric block cipher. This anonymization key is the only information which needs to be copied when consistent anonymization is done in parallel. Crypto-Pan is nowadays probably the most widely used anonymization method, and has since been modified in order to suit specific requirements, such as anonymization of flow data [36] or online anonymization of traffic on 1 Gbit/s links [37].

4.2.2 Quality of anonymization

Recently, different successful attacks on IP addresses in anonymized traces have been presented [28, 31, 32, 38]. With the awareness of the weaknesses of anonymization methods, it is important to establish policies and agreements between data-receivers and data-sharer not to carry out de-anonymization attempts [17]. Furthermore, Pang et al. [29] argue that anonymizing IP addresses alone might not be enough to preserve privacy. Consequently, a framework which allows anonymization of each header field according to an anonymization policy was presented. They also propose a novel approach to IP address anonymization. External addresses are anonymized using the widely used Crypto-PAn, while internal addresses are mapped to unused prefixes in the external mapping. Note, however, that this scheme does not preserve prefix relationships between internal and external addresses, but is on the other hand less vulnerable to certain types of attacks, as noted by Coull et al. [28].

At present, however, Crypto-PAn is still widely used and sets an de-facto standard for trace anonymization. Thus proper handling of the anonymization key is another issue that needs to be taken care of by researchers. The key is crucial, because with knowledge of the key is it straight-forward to re-translate anonymized addresses bit by bit, which opens for a complete de-anonymization of the trace. The safest solution is to generate a new key for each trace anonymization procedure, which is destroyed immediately after the anonymization process. Obviously, this approach would not provide consistency between different anonymized traces, which is one of the main features of Crypto-PAn. It is possible to re-use a single key across traces taken on different times or locations. In such setups, access to this key needs to be highly restricted, and clear policies for scenarios involving duplication of the key (e.g., for parallel anonymization purposes) are required.

4.3 Temporary storage

After discussing different considerations regarding payload removal and anonymization, it is still an open question when these operations should be performed. If a policy or an agreement with the network operator states that network data are only allowed to be stored if it is payload-stripped and anonymized, does this mean that unprocessed traces are not allowed to be recorded on mass storage devices at all? If so, is there sufficient computational power to process potentially huge amounts of Internet traffic in “real time” during the collection process? And if temporary storage of raw-traces is necessary for processing purposes, how long does

"temporary" really mean? Does the processing (payload removal and anonymization) need to be started immediately after finishing the collection? And how to proceed in case of processing errors, which might require manual inspection and treatment? When is it safe to finally delete unprocessed raw-traces? Such detailed questions are not always answered by existing policies, so it is often up to the researchers to make adequate, rational choices in order to minimize the risks of violating privacy and confidentiality concerns of users and organizations.

4.4 Access and security

Since network data can contain a number of sensitive and confidential information, it is crucial to prevent unauthorized access to (raw) trace data. In case where traces are regarded as very sensitive, it might even be necessary to encrypt the archived network data. If data needs to be copied, there should be clear hand-over policies, which help to keep track of the distribution of datasets. Additionally, the monitoring equipment and measurement nodes need to be secured carefully, since access to functional measurement nodes is probably an even better source to attackers than already collected traces. For measurement equipment and data the same security measures as for all sensitive data centers should be applied. Besides restricting physical access to facilities housing measurement equipment and storage, also network access needs to be strictly regulated and monitored. Finally, especially in case of discontinuous measurement campaigns, measurement times should be kept secret to minimize the risk of de-anonymization attacks involving identifiable activities during the measurement interval.

5 Operational difficulties

Data centers and similar facilities housing networking equipment are usually well secured and access rights are not granted easily, which is especially true for external, non-operational staff, such as researchers. Often it is required that authorized personnel are present when access to certain premises is needed. This dependency makes planning and coordination difficult and reduces flexibility and time-efficiency. Flexibility constraints are further exaggerated by the geographic location of some premises, since they are not necessarily situated in close proximity to the researcher's institute. Moreover, some significant maintenance tasks, such as installation of optical splitters, require interruption of services, which is undesired by network operators.

The above indicated operational difficulties suggest the need of careful planning of measurement activities, including suitable risk management such as slack time and hardware redundancy when possible. Generally, the sparse on-site time should be utilized with care in order to disturb normal operations as little as possible. A good way of doing so is to use hardware with remote management features, providing maximum control of operating system and hardware of the installed measurement equipment. Such remote management capabilities should include possibilities to reset machines and offer access to the system console, independent from operating systems.

A final challenge in planning Internet measurements is the short-lived nature of network infrastructure, which might influence ongoing measurement projects depending on their specific measurement locations. Generally, measurements are carried out in a fast changing en-

vironment, including frequent modifications in network infrastructure and equipment but also changes in network topologies and layouts. This changeful nature of network infrastructure is especially cumbersome for projects intended to conduct longitudinal measurements. Some changes in network infrastructure might not only require modifications or replacement of measurement equipment, but also hamper unbiased comparison of historical data with contemporary measurement data.

6 Technical aspects

Measurement and analysis of Internet traffic is not only challenging in terms of legal and operational issues, but also it is above all a technical challenge. In the following subsections we first discuss methods to physically access and collect network traffic. We will then provide discussions about other important aspects regarding Internet measurement, including strategies to cope with the tremendous amounts of data and some considerations for how to get confidence in the measured data. Finally, we will discuss the important challenge of timing and synchronization. Timing is an important issue in network measurement, especially when timing-sensitive correlation of different traffic traces is required, such as passive one-way delay (OWD) measurements or when merging network traces measured on links of opposite direction.

6.1 Traffic access methods

On shared-medium protocols such as Ethernet, passive measurements can be carried out by all nodes connected to the medium via commodity network interface cards (NICs) running in promiscuous mode. Unfortunately, NICs are not designed for monitoring purposes and do not offer effective and precise recording of network traffic (e.g., imprecise timestamp generation as discussed in 6.4.2 or unreported packet loss). In Ubik and Zejdl [5] it was shown that it is theoretically possible to monitor 10 Gbit/s links with commodity NICs (which currently can support up to 10 Gbit/s for Gigabit-Ethernet). This, however, comes with the cost of high CPU load[1] and the mentioned precision deficiencies.

Specialized traffic monitoring hardware on the other hand can provide precise traffic collection without putting extra CPU load on the monitoring host, which can then be used to perform online traffic processing instead. Currently, the most common capture cards for high-speed network measurements are Endace DAG cards [39], but also other companies offer such equipment, such as Napatech [40] or Invea-Tech [41]. Modern capture cards provide lossless, full packets data collection with precise timestamping and filtering capabilities for link speeds of up to 10 Gbit/s. These cards also report about collection problems such as dropped packets and checksum errors. Endace recently even released a capture box for 40 Gbit/s linespeed [42], which is essentially splitting 40 Gbit/s input into 4 x 10 Gbit/s output, which can then be stored and processed by 10 Gbit/s measurement nodes.

For measurements on fibre or switched connections running point-to-point protocols (e.g., High-level Data Link Control, HDLC), physical access to the network traffic can be gained in three ways:

[1] Ubik and Zejdl report about CPU usage of up to 64% on two Intel Xeon 5160 dual-core 3.00 GHz CPUs for the recording of full packets.

1. *Port mirroring*: Network devices (typically switches, but also routers) send copies of all packets seen on one or more specific port(s) to a single monitoring port to which a measurement/collection device is connected. The main advantage of this solution is its simplicity and its low cost, since many network devices support this feature out-of-the-box (e.g., Cisco's SPAN-Switch Port ANalyser feature). Furthermore, port mirroring can be remotely administrated and is thus relatively flexible in its configuration. However, there are a number of drawbacks that need to be considered before deciding to use this traffic access method:

   - *Performance*: the mirroring process is an additional burden to the network device's CPU and backplane, which might result in degradation of network performance, especially for heavily utilized devices.
   - *Packet loss*: if the aggregated traffic from the mirrored ports exceeds the capacity of the monitoring ports, packets will be dropped.[2] Even if the capacity of the monitoring port is dimensioned properly, network devices under heavy load might drop packets silently due to the additional CPU burden, which is not of high priority for a network device designed to facilitate traffic transport rather than traffic monitoring.
   - *Packet timing and ordering*: since network devices need to buffer packets from the mirrored links before forwarding them to the monitoring link, timing of packets is affected. As shown in Zhang and Moore [43], port-mirroring on two switches from different vendors introduced significant changes to inter-packet timing and packet-re-ordering, even at very low levels of utilization. These results imply that port-mirroring is likely to introduce bias for all analyzing purposes that include packet inter-arrival time statistics or rely on proper packet arrival order (such as analysis of TCP sequence numbers).
   - *Omitted packets*: packets with errors in the lower layers of the protocol stack (layers 1 and 2) are usually dropped by network devices and thus not mirrored, which disqualifies port-mirroring for low-layer troubleshooting and debugging purposes.

2. *Port mirroring on dedicated box*: small switches dedicated to mirror link(s) are also called *aggregation TAPs* (Test Access Ports). The main advantage of this solution is increased buffer sizes and a dedicated CPU and backplane, offering some protection against packet loss. However, since this solution requires additional hardware expenses while still not resolving many problems with port mirroring on network devices (packet timing and ordering, omitted packets), it is seldom applied in existing studies dealing with measured network data.

3. *Network TAP*: a network TAP is a device intercepting traffic on a network link, analogous to telephone taps. TAPs are available for copper and optical fiber supporting up to 10Gbit/s. TAPs split the incoming signal into two signals, one signal continuing on the network link and the other signal passed on to a measurement/collection device. While copper TAPs use electronic circuits for duplication, fiber TAPs optically split the signal

[2] Mirroring a full-duplex port requires twice the capacity on the monitoring port (one link in each dir.)

and are thus called *optical splitters*. For duplex links, each direction needs to be tapped individually, and the resulting traffic stream might or might not be merged in the attached measurement device, depending on the specific measurement purpose. Such passive TAPs are non-intrusive and do not interfere with the data or the timing of the data, eliminating most of the drawbacks with port-mirroring. However, in addition to the extra cost, installation of passive TAPs results in a service disruption of the links monitored. Furthermore amplifiers for the optical signal might be required in order to compensate for the power loss due to the optical splitter.

6.2 Data amount

The amount of data carried on modern Internet backbone links makes it non-trivial to record. This will continue to be a challenge in the foreseeable future, since backbone link bandwidths increase in at least the same pace as processing and storage capacities, with 10 Gbit/s links established as state-of-the-art, 40 Gbit/s links already operational and 100 Gbit/s links planned to be introduced in 2010.

6.2.1 Hardware requirements

Increasing link speeds will emphasize hardware requirements of measurement nodes. Some examples of possible bottlenecks within measurement hardware are listed below:

1. *I/O bus*: If high-capacity backbone links operate in full speed, contemporary I/O bus capacities (e.g., 8 Gbit/s theoretical throughput for PCI-X or 16 Gbit/s for 8-lane PCIe 1.x) are hardly sufficient to process data from complete packet header traces. This insufficiency is even more severe when the data needs to pass the bus twice, once to the main memory and another time to secondary storage. Cutting edge PCIe 2.0 or the upcoming PCIe 3.0 featuring 16 or 32 lanes with theoretical throughputs of up to 8 Gbit/s per lane might overcome this bottleneck for current link speeds.

2. *Memory speed*: If the measurement host's main memory is used to buffer traffic before writing it to disk (e.g., to handle bursts in link utilization), it needs to be considered that memory access speeds do not develop in the same pace as link capacities. Modern DDR2-1066 SDRAM DIMMs offer theoretical transfer rates of 68 Gbit/s (8533 MB/s), which would not be sufficiently fast to buffer data from 100 Gbit/s links on full capacity. Only DDR3 SDRAM technology might nominally overcome the 100 Gbit/s border, with I/O clock speeds of up to 800 Mhz (offering transfer rates of 12,800 MB/s or 102.4 Gbit/s). DDR3 DIMMs are expected to penetrate the market throughout the year 2010. However, it is not enough to store data in memory, it eventually also needs to be read out on disks, which doubles the data-rate required. On the other hand, utilization of memory interleaving between multiple memory banks is a common technique to increase memory throughput on many motherboards/chipsets.

3. *Storage speed*: Even if the I/O bus bottleneck could be overcome, the speed of storage array systems would not suffice. Modern storage array network (SAN) solutions offer in the best case 10 Gbit/s rates. Traditional SCSI systems provide nominal throughput rates of around 5 Gbit/s (e.g., Ultra-640 SCSI), and cutting-edge serial buses such as SAS-2.0 (serial attached SCSI) and SATA (serial ATA) reach 6 Gbit/s. Transfer rates for single hard disks range from around 110 MB/s for good disks with 7200 RPM (revolutions per minute) to 170 MB/s sustained transfer rates for the latest 15,600 RPM drives, which can be scaled up by deployment of RAID disk arrays (e.g., RAID-0). These throughput rates could potentially cope with complete packet-level traces of 10 Gbit/s links, but cannot keep up with higher link rates.

4. *Storage capacity*: All these considerations still do not take the required storage capacity into account. Longitudinal measurement campaigns, recording up to several Gigabytes of network data per second, are non-trivial tasks and will eventually be limited by storage capacities.

The discussion provided above shows that recording of complete packet-level traces is strictly bounded by hardware performance, even if it may theoretically be matched with today's hardware. Fortunately, backbone links are typically over-provisioned, and average throughput is far from line-speed. Even though this fact alleviates some technical problems (e.g., storage capacity), measurement nodes still need to be able to absorb temporary traffic bursts. If such traffic amounts cannot be handled, random and uncontrolled discarding of packets will take place, resulting in incomplete, biased datasets, which is highly undesirable with respect to the accuracy of scientific results.

6.2.2 Traffic data reduction techniques

As shown, measurement of complete packet-level traces is technically not always feasible. In the following paragraphs some approaches aiming to reduce data amounts while still preserving relevant information are presented.

1. *Filtering*: If network data are collected with a specific, well defined purpose, traffic filtering is a valid solution to reduce data amounts. Traffic can be filtered according to hosts (IP addresses) or port numbers, which is probably the most common way to filter traffic. But also other arbitrary header fields or even payload signatures can be used as filter criteria. This was already successfully demonstrated by a very early study about Internet traffic characteristics, carried out by Paxson [44]. In this work, only TCP packets with SYN, FIN or RST packets were considered for analysis. Filtering only packets with specified properties can be done in software (e.g., BSD packet filter [4]), which is again limited by processing capabilities, or in hardware (e.g., by FPGAs), which can provide traffic classification and filtering according to a set of rules up to 10 Gbit/s line speeds (e.g., Endace DAG cards [39]).

2. *Sampling*: Another method to reduce data amounts of packet-level traces is packet sampling. Packet sampling can be done systematically, in static intervals (record every *N*th packet only) or in random intervals, like proposed by sFlow [45]. Alternatively, also more sophisticated packet sampling approaches have been proposed, such as adaptive packet sampling [46]. A good overview of sampling and filtering techniques for IP packet selections can be found in a recent Internet draft by Zseby et al. [47].
3. *Packet truncation*: A common tradeoff between completeness of packet-level traces and hardware limitations is to truncate recorded packets after a fixed number of bytes. Depending on the chosen byte number, this approach is either not guaranteeing preservation of complete header information or includes potentially privacy-sensitive packet payloads. To address this dilemma, it is common practice to truncate packets in an adaptive fashion, i.e., to record packet headers only. As discussed in Section 4.1, stripping of payload data has also the advantage of addressing privacy concerns. The processing of packets, i.e., the decision of what to keep and where to truncate, can in the best case be done online, especially if hardware support is given. Alternatively, packets can be truncated after a specified packet length of N bytes, and removal of payload is then done during offline processing of the traces.
4. *Flow aggregation*: As discussed in Section 2, a common way to reduce data while still keeping relevant information is to summarize sequences of packets into flows or sessions. The advantage is, that classification of individual packets into flows can be done online, even for high-speed networks due to optimized hardware support of modern measurement equipment. This means that the measurement hosts only need to process and store reduced information in form of flow records, which is no burden even for off-the-shelf servers. Flow records can also be provided by the network infrastructure itself (e.g., by routers), which explains why the most common flow record format NetFlow [48] was developed by Cisco. In 2006, the IETF proposed IPFIX [49] as universal flow standard, which is actually derived from NetFlow v9. Even though usage of flow records is already reducing data amounts, various sampling techniques have been proposed for flow collection as well. Flow sampling approaches include random flow sampling (e.g., NetFlow), sample and hold [50] and other advanced sampling techniques, such as in [46, 51, 52].

6.2.3 Archiving of network data

Since measuring Internet traffic is a laborious and expensive task, measurement projects sometimes want to archive not only their analysis results, but also the raw data, such as packet-level traces or flow data. Archiving raw data can be important for several reasons:

1. keeping scientific results reproducible;
2. allowing comparisons between historical and current data;
3. making additional analysis regarding different aspects possible;
4. sharing datasets with the research community, as discussed in Section 7.

Archiving network traces is not always a trivial task, especially for longitudinal, continuous measurement activities. A complete description of different archiving solutions is not within the scope of this paper, but it is recommended to consider risk management such as error handling and redundancy. Data redundancy can be provided by suitable RAID solutions or by periodic backups on tertiary storage such as tape libraries. To further reduce data amounts, compression of traffic traces and flow data for archiving purposes is common practice. Standard compression methods (e.g., Zip) reduce data amounts to 50%, which can be further optimized to 38% as shown in [53]. When network data are archived, it is also crucial to attach descriptive meta-data to datasets, as argued by Pang et al. [29], Paxson [54], and Cleary et al. [55]. Meta-data should include at least descriptions of the measurement and processing routines along with relevant background information about the nature of the stored data, such as network topology, customer breakdown, known network characteristics or uncommon events during the measurement process. To avoid confusion, Pang et al. recommend to associate meta-data to datasets by adding a checksum digest of the trace to the meta-data file.

6.3 Trace sanitization

We define *trace sanitization* as the process of checking and ensuring that Internet data traces are free from logical inconsistencies and are suitable for further analysis. Hence, the goal of trace sanitization is to build confidence in the data collection and preprocessing routines. It is important to take various error sources into account, such as problems with measurement hardware, bugs in processing software and malformed or invalid packet headers, which need to be handled properly by processing and analysis software. Consistency checks can include checksum verification on different protocol levels, analysis of log files from relevant measurement hard- and software and ensuring timestamp consistency. Furthermore, an early basic analysis of traces can reveal unanticipated errors, which might require manual inspection. Statistical properties and traffic decompositions which highly deviate from "normally" observed behavior very often reveal measurement errors (such as garbled packets) or incorrect interpretation of special packets (such as uncommon or malformed protocol headers). Obviously, the results of the trace sanitization process including a documentation of the sanitization procedure should be included into the meta-data of the dataset. An example of a sanitization procedure is described in Section 8.4. Another example of an automated sanitization process is provided by Fraleigh et al. in [56], and a more general discussion about sanitization can be found in Paxson's guidelines for Internet measurement [54].

6.4 Timing issues

Internet measurement has an increasing need for precise and accurate timing, considering that small packets of e.g., 40 bytes traveling back to back on 10 Gbit/s links arrive with as little as 32 nanoseconds (ns) time difference. For each packet a timestamp is attached when recorded, which forms the basic information resource for analysis of time related properties such as throughput, packet-inter-arrival times and delay measurements. Before discussing different timing and synchronization issues occurring in Internet measurement, it is important to define a common terminology about clock characteristics. Next, an overview of timestamp for-

mats will be given, including the important question of when timestamps should be generated during the measurement process. After presenting common types of clocks, this subsection gives a discussion of how accurate timing and clock synchronization can be provided.

6.4.1 Time and clock terminology

First of all it is important to distinguish between a clock's reported time and the true time as defined by national standards, based on the coordinated universal time (UTC [3]). A perfect clock would report true time according to UTC at any given moment. The clock terminology definitions provided below follow Mills' network time protocol (NTP) version 3 standard [57] and the definitions given by Paxson in [58].

- A clock's *resolution*, called *precision* in the NTP specification, is defined by the smallest unit a clock time can be updated, i.e., the resolution is bounded by a clock "tick".
- A clock's *accuracy* tells how well its frequency and time compare with true time.
- The *stability* of a clock is how well it can maintain a constant frequency.
- The *offset* of a clock is the differences between reported time and true time at one particular moment, i.e., the offset is the time difference between two clocks.
- A clock's *skew* is the first derivative of its offset with respect to true time (or another clock's time). In other words, skew is the frequency difference between two clocks.
- A clock' *drift* furthermore is the second derivative of the clock's offset, which means drift is basically the variation in skew.

6.4.2 Generation and format of timestamps

Regardless of how timing information is stored, it is important to understand which moment in time a timestamp is actually referring to. Packets could be timestamped on packet arrival of the first, the last or any arbitrary bit on the link. Software-based packet filters, such as the BSD packet filter [4], commonly timestamp packets after receiving the end of an arriving packet. Furthermore, software solutions often introduce errors and inaccuracies, since arriving packets need to be transported via a bus into the host's main memory, accompanied by an undefined waiting period for a CPU interrupt. Additionally, buffering of packets in the network card can lead to identical timestamps for a number of consecutive packets. These sources of errors are typically not an issue for hardware solutions, such as Endace DAG cards [39]. Another difference is that dedicated measurement hardware generates timestamps on the beginning of packet arrival. If it is for technical reasons not possible to determine the exact start of a packet, timestamps are generated after arrival of the first byte of the data link header (e.g., HDLC), as done by DAG cards for PoS (Packet over SONET) packets [59].

There are also different definitions of how time is represented in timestamps. The traditional Unix timestamp consists of an integer value of 32 bits (later 64 bits) representing seconds since

[3] UTC is derived from the average of more than 250 Cesium-clocks situated around the world.

the first of January 1970, the beginning of the Unix epoch. The resolution presented by this timestamp format is therefore one second, which is clearly not enough to meet Internet measurement requirements. PCAP, the trace format of the BSD packet filter, originally supported 64 bit timestamps that indicated the number of seconds and microseconds since the beginning of the Unix epoch. A more precise time stamp format was introduced with NTP [57], representing time in a 64 bit fixed-point format. The first 32 bits represent seconds since first of January 1900, the remaining 32 bits represent fractions of a second. In Endace ERF trace format, a very similar timestamp scheme is used, with the only difference that ERF timestamps count seconds from the start of the Unix epoch (January 1st 1970). These formats can store timestamps with a resolution of 232 pico seconds ($1s/2^{32}$). Currently, the most advanced hardware can actually use 27 bits of the fraction part, providing a resolution of 7.5 ns [60]. Future improvements of clock resolutions will require no modification of timestamp or trace formats but only take advantage of the currently unused bits in the fraction part. Note that the different timestamp formats within different trace formats can have negative effects on trace conversion (Section 2). Converting ERF traces into PCAP traces might imply an undesired reduction of precision from nanosecond to microsecond scale.

6.4.3 Types of clocks

Current commodity computers have typically two clocks. One independent, battery powered *hardware clock* and the *system, or software clock*. The hardware clock is used to keep time when the system is turned off. Running systems on the other hand typically use the system clock only. The system clock, however, is neither very precise (with resolutions in the millisecond range), nor very stable, with significant skew. In order to provide higher clock accuracy and stability for network measurements, Pasztor and Veitch [61] therefore proposed to exploit the TSC register, a special register which is available on many modern processor types. Their proposed software clock counts CPU cycles based on the TSC register, which offers nanosecond resolution, but above all a highly improved clock stability, with a skew similar to GPS synchronized solutions.

Since tight synchronization is of increasing importance, modern network measurement hardware incorporates special timing systems, such as the DAG universal clock kit (DUCK) [59, 60] in Endace DAG cards. The most advanced DUCK clocks currently run at frequencies of 134 MHz, providing a resolution of 7.5 ns, which is sufficient for packets on 10 Gbit/s links. The DUCK is furthermore capable of adjusting its frequency according to a reference clock which can be connected to the measurement card. Reference clocks (such as a GPS receiver or another DUCK) provide a pulse per second (PPS) signal, which provides accurate synchronization within two clock ticks. For 134 MHz oscillators this consequently means an accuracy of $\pm$ 15ns, which can be regarded as very high clock stability.

6.4.4 Clock synchronization

How accurate clocks need to be synchronized when performing Internet measurements depends on the situation and the purpose of the intended analysis. For throughput estimation, microsecond accuracy might be sufficient. On the other hand, some properties, such as delay or jitter on

high-speed links, often require higher accuracy. In situations with a single measurement point, instead of accurate timing it might be more important to provide a clock offering sufficient stability. Other situations require tight synchronization with true time, while sometimes it is more important to synchronize two remote clocks, and true time can actually be disregarded. In the following paragraphs, we first present some ways of how to synchronize clocks with each other (where one clock might in fact represent true time). This discussion includes an interesting solution to synchronize measurement hardware located in close proximity, which is especially useful when traces recorded on two unidirectional links need to be merged. Finally, methods allowing correction of timing information retrospectively are presented, which is used to adjust one-way-delay measurements, but can also be applied on passive traffic measurements involving remote measurement locations.

6.4.4.1 Continuous clock synchronization

1. *NTP*: The most common way to synchronize a clock of a computer to a time reference is the network time protocol NTP [57]. NTP is a hierarchical system, with some servers directly attached to a reference clock (e.g., by GPS). Such directly attached servers are called stratum 1 servers. This timing information is then distributed through a tree of NTP servers with increasing stratum numbers after each hop. Depending on the type of the network, the distance to the NTP server and the stratum number of the server, NTP can provide clients with timing accuracy ranging from one millisecond to tens of milliseconds. However, forms of clock skew, drift and jumps despite usage of NTP have been reported by Paxson in [58]. These observations lead to the recommendation to disable NTP synchronization during measurement campaigns, thus providing NTP synchronization only before and after measurement intervals.

2. *GPS*: Since the propagation of timing information over networks obviously limits the accuracy of NTP synchronization, some measurement projects directly attach GPS receivers to their measurement equipment. The global positioning system, GPS, is basically a navigation system based on satellites orbiting the earth. The satellites broadcast timing information of the atomic clocks they carry. GPS receivers, however, can not only be used for positioning, but they can also be used as a time source since highly accurate timing information is received in parallel. GPS receivers can therefore provide clock synchronization within a few hundreds of nanoseconds. Unfortunately, GPS receivers require line of sight to the satellites due to the high frequencies of the signals, which means that GPS antennas normally must be installed outside buildings, ideally on the roof. This can be a severe practical problem, especially for measurement equipment located in data centers in the basement of high buildings.

3. *Cellular networks*: To overcome the practical problems of GPS, it is possible to use signals of cellular telephone networks, such as code division multiple access (CDMA) as synchronization source for measurement nodes (e.g., provided by [62]). Base stations of cellular networks are all equipped with GPS receivers to retrieve timing information.

This information is then broadcasted as a control signal within the network. Since base stations operate on lower frequencies, it is possible to use these base stations as timing sources even inside buildings. The accuracy provided by CDMA time receivers is very close to GPS standards. However, due to the unknown distance to the base station, clocks synchronized by CDMA will have an unknown offset from UTC. Furthermore, the offset is not guaranteed to be constant, since receivers in cellular networks can switch between base stations for various reasons.

4. *SDH protocol*: A recently proposed approach distributes time from an UTC node using existing backbone communication networks, such as OC192 links. This system yields an accuracy of a few nanoseconds, which is done by utilizing the data packages already transmitted in the system [63]. To our knowledge, this novel approach has not been used in Internet measurement yet, but it might be an interesting alternative for upcoming measurement projects.

5. *Daisy-chaining of timestamping clock*: Endace DAG cards offer an additional solution for clock synchronization, which is very attractive for measurement hosts located in close proximity. The DUCK, a clock kit on every DAG cards, offers also output of PPS signals [60]. This feature can be used to chain DAG cards together by simple local cabling in order to keep them tightly synchronized. If no external reference clock is available, at least accurate and consistent timestamping between the connected DAG cards is provided. This approach is often used when two links in opposing directions are measured with two separate measurement hosts, since it allows merging of the traces into one bidirectional trace. In this case, synchronization between the two clocks is of main importance, and accuracy with respect to true time (UTC) is no major concern.

6.4.4.2 Retrospective time correction

In some cases (e.g., for large geographical distances), traffic traces timestamped by different clocks need to be compared. Even if clock synchronization by NTP or GPS is provided, forms of clock skew, drift and jumps cannot be ruled out [58]. To compensate for these errors, retrospective time correction algorithms have been proposed. These algorithms have been designed to remove clock offset and skew from one-way delay (OWD) measurements. For distributed passive traffic measurements a set of passive OWD measurements can be obtained given that sufficient (uniquely identifiable) packets traverse both measurement locations. In this case, the correction algorithms can be applied on passive packet traces collected at different locations.

The observed OWD (OOWD) for the ith packet can be calculated as

$$OOWD(i) = t_r(i) - t_s(i) \tag{1}$$

where t_r and t_s are the timestamps of the ith packet at receiver and sender respectively. Given a relative clock offset δ between the receiver and sender clock, the actual OWD can then be derived by:

$$OWD(i) = OOWD(i) - \delta(i) \tag{2}$$

Early approaches assumed zero clock drift (i.e., constant clock skew) and no instantaneous clock adjustments in order to estimated clock skew. This means that a series of OWD(i) measurements would indicate a trend, following steady increasing or decreasing $\delta(i)$, depending on the sign of the clock skew according to the reference clock (typically the receivers clock). Moon et al. [64] proposed a *linear programming based algorithm* in order to estimate the slope α of the resulting trend starting at an initial offset β, i.e., $\delta(1)$.

Newer approaches also try to take clock dynamics into account by partitioning the measurements into segments representing periods with constant skew between clock jumps or frequency adjustments. However, even these newer approaches assume zero drift between the two clocks. Zhang et al. [65] proposed a set of algorithms based on the computation of *convex hulls* in order to remove skews. A devide-and-conquer approach is used to identify clock resets (i.e., jumps) and a marching algorithm should identify epochs of frequency adjustments, between which the relative clock skew is estimated and removed. However, Zhang's approach has some limitations, such as limitations of how often and frequent clock resets can occur.

Wang et al. [66] tried to generalize previous approaches by converting the clock dynamics detection to a *time series segmentation* problem. The resulting clustering based OTDTS algorithm (Optimized Top-Down Time series Segmentation) segments delay time series at the points at which clock resets or adjustments occur. For each segment, clock skew can then be estimated and removed either by the linear programming based algorithm as in Moon et al. or the convex hull approach as proposed by Zhang et al.

A fuzzy-based approach for estimation and removal of clock skew and reset has been proposed by Lin et al. [67], which is claimed to be more accurate and robust than Zhang's convex-hull approach. The authors combine the fuzzy clustering analysis [68] with the linear programming based or the convex-hull approach, where the fuzzy analysis is used to distinguish between clock resets and temporary delay variations such as traffic congestions.

Khlifi and Gregoire [69] tried to further reduce the complexity of previous skew estimation approaches such as linear programming and convex-hull. Two techniques for offline skew removal are proposed. The *average technique*, which reduces the complexity of previous algorithms from O(N) to O(1) by calculating the average of the delay differences between consecutive packets. The *direct skew removal technique* remains at O(N) complexity while increasing the accuracy by iteratively evaluating the set of possible skews until an optimal value is reached. Furthermore, two online techniques for skew removal are proposed, namely the *sliding window algorithm* which tracks the skew by continual evaluation of variations in the minimum OOWD and a *combined algorithm*, combining the sliding window and convex-hull approaches.

7 Data sharing

The discussions about all the legal, operational and technical difficulties involved in conducting Internet measurement clearly show that proper network traces are the result of a laborious and costly process. This explains why currently only few researchers and research groups have the possibilities to collect Internet backbone data, which makes proper traces a scarce resource. Therefore, the Internet measurement community has repeatedly been encouraged to share their valuable datasets and make them publicly available [70, 54, 33], given that sharing of network

data are legally permitted (see Section 3). Sharing network data are not only a service to the community, but it is also an important factor related to credibility of research results. Generally, sharing and archiving of data are fundamental to scientific progress and help to improve scope and quality of future research.

Sharing network traces adds reliability to research, since it makes results reproducible by the public, which allows verification and in the best case confirmation of earlier results. This should be best practice in research, encouraging fruitful research dialogs and discussions within the research community. Furthermore, releasing measurement data makes it possible to compare competing methods on identical datasets, allowing fair and unbiased comparison of novel methodologies. Publishing of network data also gives the additional benefit of providing the original data owners with supplementary information about their data, yielding a better and more complete understanding of the data. Finally, in order to get a representative view of the Internet, diverse data at different locations and times needs to be collected and shared within the research community. In a note on issues and etiquette concerning use of shared measurement data [33], Allman and Paxson discuss the above-mentioned benefits of data availability, including ethical and privacy considerations, as discussed here in Section 4.

Before data can actually be shared, researchers need to be made aware of existing and available datasets. A system for sharing Internet measurements was proposed by Allmann in 2002 [71]. This system was inspiration for CAIDA to finally implement the Internet measurement data catalog DatCat [72], which allows publication of meta-data about network datasets. The goal of this project was to provide the research community with a central database, providing searchable descriptions of existing datasets.

Actual sharing of data, however, is problematic due to the mentioned uncertain legal situation and ethical considerations. Even if traces are desensitized by technological means (e.g., by payload removal and anonymization), additional sharing policies are required in order to safeguard possible technological shortcomings such as trace de-anonymization (see Section 4.2.2). Kenneally and claffy therefore try to facilitate protected data sharing by proactively implementing management of privacy risks in the Privacy-Sensitive Sharing framework PS2 [17]. PS2 is based on an hybrid model relying on a policy framework applying privacy principles together with a technology framework implementing and enforcing privacy obligations. So far, the authors have only outlined the PS2 framework without focusing on a particular implementation of a data sharing tool.

An alternative approach to data sharing was suggested by Mogul in a presentation in 2002 [73]. He proposes a "move code to the data" solution, where analysis programs are sent to the data owners (e.g., network operators) and executed on-site. In this scenario, only results would be shared, but not the network data itself. This is an interesting approach, but it highly depends on the will of the involved parties to cooperate.

As a solution to the privacy/utility tradeoff in data sharing, Mirkovic [74] proposed a privacy-safe sharing framework based on secure queries. Instead of sharing (copies of) raw traces, data access is re-directed through an online interface providing a query language, allowing customized sets of queries to be run on the data and returning de-sensitized, aggregated information fitting the specific research goals. Individual privacy policies can thus be enforced by the query language interpreter.

Parate and Miklau [75] very recently proposed a sharing framework in which trace owners can match an anonymizing transformation of communication data with the requirements of analysts. The framework should so enable formal reasoning about the impact of anonymization operations on trace utility and privacy.

CASFI (Collect, Analyze, and Share for the Future Internet) is currently working on the CASFI Data Sharing Platform [76], a framework that helps to share not only data, but also the data management platform to facilitate better collaboration between multiple research groups. The platform should help to manage local data, and at the same time provide an interface to remote data, in order to get a consistent overview of relevant data without visiting different web interfaces.

8 Experiences from the MonNet project

This section provides a description and lessons learned from a project for passive Internet traffic monitoring and analysis conducted at Chalmers University of Technology: the MonNet project. The goal of the project is to provide a better understanding of Internet traffic characteristics based on empirical data, i.e., passive measurements on backbone links.

8.1 Legal and ethical approval

In summer 2004, MonNet, as a project regarding Internet and traffic measurements and analysis, was proposed to the SUNET board. In order for the project to be granted, the SUNET board required permission from the "central Swedish committee for vetting ethics of research involving humans" (*Etikprövningsnämnden, EPN*), which is among other things responsible for vetting research that involves dealing with sensitive information about people or personal information. Ethical vetting in this committee is carried out in six regional boards. After elaborate discussions about the de-sensitization process of the traces, the regional ethics committee permitted the MonNet measurements to take place. Traffic monitoring was granted under the conditions that user payload is removed and IP addresses are anonymized, e.g., with prefix-preserving Crypto-PAn. We consider the provided permission from the research ethics committee as an appropriate safeguard, as requested for measurement of Internet traffic for scientific purposes by current EU directives (see Section 3.1).

Lessons learned:

1. During the vetting process, it turned out that the committee had little understanding of the technical background and implications. This resulted in a policy suggested by the MonNet project itself which, after some amendments, was approved by the vetting committee. Researchers therefore need to be aware of how and on which level of detail policies for de-sensitization and sharing are formulated in order not to hinder sound scientific research while still respecting privacy of individuals.

2. Obtaining legal approval can introduce a long and unpredictable time delay, which needs to be considered in the project planning.

8.2 Operational considerations

Operational considerations include choice of measurement location and access to the measurement premises at the network operator:

8.2.1 Measurement location

Before actual measurements could be started, a measurement location needed to be chosen. The MonNet project initially (from 2005 until 2007) recorded traffic on GigaSUNET Internet backbone links. Data were collected on the outside point of an OC192 ring, which was the primary link from the region of Gothenburg to the main Internet outside Sweden. The link carried traffic from major universities, large student residential networks and a regional access point exchanging traffic with local ISPs. This choice was in the first place made to achieve traces with a high level of traffic aggregation, since the link measured carries data transferred between a regional network and the main Internet.

The former ring architecture has during 2007 been upgraded to OptoSUNET, a star structure over leased fiber. All SUNET customers are since then redundantly connected to a central Internet access point in Stockholm. Besides some local exchange traffic, the traffic routed to international commodity Internet is carried on two links (40 Gbit/s and 10 Gbit/s) between SUNET and a Tier-1 provider. Since 40 Gbit/s measurement equipment was economically impossible (it would essentially require measurement equipment for 4 x 10 Gbit/s links), the measurement infrastructure was moved to the 10 Gbit/s link with the highest possible level of traffic aggregation: the 10 Gbit/s link between SUNET and NorduNet, located in Stockholm. According to SNMP statistics, this link carries 50% of all inbound but only 15% of the outbound traffic volume.[4] In July 2009 an additional 10 Gbit/s link was installed in parallel with the existing one in order to keep up with the increasing traffic volumes.

8.2.2 Access to premises

The initial measurement location had the additional feature of being located in the same city as the research group, at the Chalmers University of Technology in central Gothenburg. This feature was of great advantage for very practical reasons:

- installation of optical splitters and operation of specialized measurement cards on 10 Gbit/s speeds could not be tested beforehand in a lab environment, which required some adjustments on site as reaction on early experiences.
- even tested commodity PCs used as measurement nodes required additional physical visits at the measurement location due to unexpected early hardware defects such as harddisk crashes and RAID controller problems, which are most common in early and very late stages of hardware lifecycles (following the bathtub-curve).

Even if located in the same city, physical access to the actual measurement location, situated in secure premises of an external network operator, was not entirely straight-forward to obtain and involved inconvenient administrative overhead and idle times. Limited access possibilities

[4] 85% of the outbound traffic is routed via the 40 Gbit/s link.

to operational network facilities and equipment is common for many network research projects and should be taken into account when negotiating rules and policies with the network operator enabling the measurements. To prevent some of the physical visits, it is useful to equip nodes with remote management hardware capable of hardware resets and access to the system console. Unfortunately, the remote management cards recommended by the supplier of MonNet's measurement nodes turned out to be unstable and unreliable (i.e., useless), which highlights the importance of using as much well-established and tested hardware as possible. As a consequence, SSH access, which was granted only to specified hosts inside Chalmers University, was the only way to remotely maintain and operate the measurement nodes.

Lessons learned:

1. Good contacts with the network's operators and well defined access procedures alleviate installation and configuration of measurement equipment.

2. Proven remote management possibilities should be exploited.

3. Measurement locations should in the first place be chosen in order to provide data supporting the specific research purpose. If possible, it is of advantage to choose geographically close locations, which is especially true in early project phases.

4. Unforeseen delays by external parties (ethics committee, operators, hardware suppliers) require sufficient slack times and parallel work, especially in early project phases.

5. In face of frequent changes to network topologies and technologies, measurement hardware supporting various link-layer technologies (e.g., PoS HDLC and GbE) and wavelengths is preferable since it can be reused without additional costs.

8.3 Technical solution

The measurement nodes applied have been designed to meet the anticipated requirements of packet-header measurements on PoS OC192 links. During the planning phase, related measurement projects such as NLANR PMA's OC48MON [77] and Sprint's IPMON [56] provided valuable inspiration. The described technical solution is based on state-of-the-art hardware available during the design phase in 2004.

8.3.1 Traffic access

Optical splitters on two OC-192 links, one for each direction, are used to capture PoS HDLC traffic. Since the signal strength was quite high, splitters with a 90/10 ratio turned out to be sufficient for the sensitivity of the measurement cards, while not requiring any additional signal amplifiers on the production links. Each optical splitter, tapping either the inbound or outbound OC912 link, is attached to an Endace DAG6.2SE card sitting in one of the measurement nodes via a PCI-X 133 MHz 64-bit PCI interface. The DAG cards have been configured with a buffer reserved from the node's main memory in order to deal with burst of high traffic load. For Linux

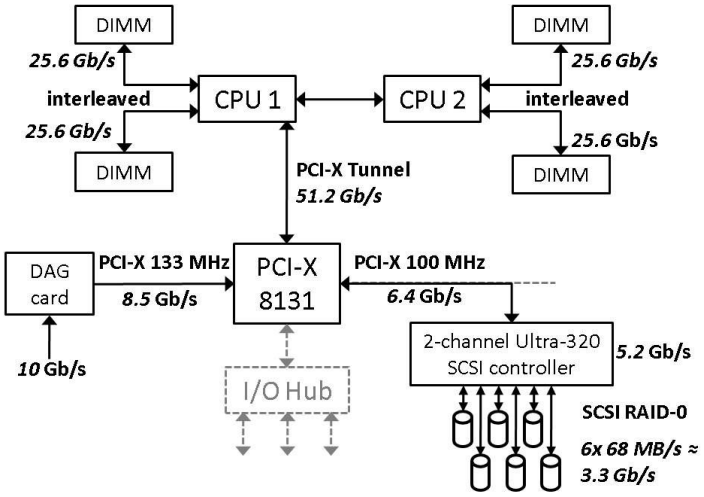


Figure 1: Block diagram of measurement nodes

systems, the buffer size needs to lie between 128 MB and 890 MB. Endace recommends a buffer size of at least 32MB for OC-12 links, thus we conservatively chose a large buffer of 512 MB for OC-192 links, which did not result in any packet loss due to insufficient buffer space in any of our measurements.[5] DAG6.2SE cards can capture 10 Gbit/s on links with optical wavelengths between 1300 and 1600 nm with STM-64c, 10GbE WAN and 10GbE LAN encapsulations. Packets are timestamped with a resolution of 15 ns.

8.3.2 Collection hardware

The two measurement nodes are designed and configured identically. A schematic block diagram of the relevant components is shown in Fig. 1. A measurement node consists of two AMD Opteron 64-bit processors with 2 GHz clock frequency and a total of 2 GB of main memory, 1 GB per CPU as two interleaved 512 MB DDR-400 SDRAM DIMMs. The Tyan K8SR motherboard is equipped with an AMD-8131 PCI-X Tunnel chipset connecting the processing units with I/O devices on PCI-X slots. The DAG6.2SE card is the only device attached to the 133 MHz 64-bit PCI-X slot. On slot 2, supporting 100 MHz, six SCSI disks are connected to a dual-channel Ultra-320 SCSI controller. The SCSI disks are configured to operate in RAID0 (striping), and thereby add up to about 411 GB of cumulated disk-space for preliminary storage of collected network traces. The 6 Maxtor Atlas SCSI disks reach a sustained data rate of between 40 and 72 MB/s, depending on the cylinder location. A series of tests with sequential writes on the RAID0 system resulted in an average data rate of about 410 MB/s (3.3 Gbit/s). Furthermore, a mirrored RAID-1 disk containing the operating system is connected to the IDE controller (not visualized in Fig. 1).

As evident in Fig. 1, the bottleneck of this configuration is the storage system, with about 3.3 Gbit/s throughput. But also the nominal throughput of the SCSI interface (5.2 Gbit/s) and the PCI-X buses (8.5 and 6.4 Gbit/s, respectively) are not sufficient to collect full packet traces in

[5]Dropped packets due to insufficient buffer space, PCI bus limitations or losses between the DAG card and the memory are reported by DAG cards.

full line speed on 10 Gbit/s networks. Also note that the buses above the PCI-X 8131 tunnel in the figure are traversed twice during measurements (from the DAG Card into Memory, and then back to the storage), which, however, does not pose a bottleneck in the present configuration.

8.3.3 Time synchronization

During measurements, the two DAG cards have been synchronized with each other using Endace's DUCK Time Synchronization [59, 60] with no external reference time. Before and after measurements, the DAG cards were synchronized to true time (UTC) using a pool of three stratum 1 NTP servers. NTP synchronization was disabled during the measurements, since forms of clocks skew, drift and jumps despite usage of NTP are problematic as described earlier. DUCK, however, can provide an accurate and consistent timestamping between the connected DAG cards ranging between $\pm$ 30ns according to Endace [60], even though their time might not be accurate with respect to true time. The tight synchronization between the measurements of opposing traffic directions allows simple merging of the unidirectional data into bidirectional traces.

8.3.4 Processing and archiving platform

After data collection and a pre-processing procedures on the measurement nodes, the resulting traces have been transferred via a Gigabit-Ethernet interface and a 2.5 Gbit/s Internet connection to the storage and processing server located in a secured server room at Chalmers University. The processing platform is attached to an external SCSI array box with a RAID5 configuration, providing 2 TB of storage. 2 TB can store around 35 hours of compressed, bidirectional packet header data collected on the current measurement location in OptoSUNET. Since this is not sufficient for longitudinal measurement campaigns, an archiving solution was required. Due to a tight economic situation, this was solved by acquisition of relatively cheap 1 TB SATA disks, which have been temporary attached via USB. After archiving the data, the disks have been placed offline in a safe when not in use. With the rapidly decreasing storage costs during recent years, it was possible to install an additional 3 TB NAS (network array storage system) with RAID5 configuration acting as online (though slow) archiving system.

Lessons learned:

1. Passive TAPs (optical splitters) in combination with specialized measurement cards is the only way to ensure lossless and precise traffic measurement on high-speed links, which is required in many research situations.

2. Since measurement cards are disproportionally expensive compared to commodity equipment, it is worth it to invest in one measurement node per link (instead of multiple measurement cards in one node) with high quality state-of-the-art hardware components.

3. Measurement nodes need to be designed carefully - performance of each component needs to be considered in order to identify possible bottlenecks.

4. Time synchronization by daisy chaining the DAG cards worked very well and was straight forward, avoiding a lot of timing-related problems (such as need for retrospective time corrections) when analyzing or merging the unidirectional traces.

5. Archiving of network traces should be considered from the beginning, since it inevitable needs to be solved and thus needs to be part of the technical and economical planning.

8.4 Trace pre-processing

Pre-processing of traffic traces including reduction, de-sensitization and sanitization, is carried out on the measurement nodes during and immediately after the collection.

8.4.1 Traffic reduction

The DAG cards have been configured to capture the first 120 bytes of each PoS frame to ensure that all link-, network-, and transport-headers are preserved. The remaining payload fractions have been removed later during the pre-processing of the traces. The average packet size on the links lies around 700 bytes, which means a maximal throughput of around 1.8 million frames per second on a 10 Gbit/s link. 44% of all frames are smaller than 120 bytes and thus not truncated by the DAG card. As a result, the average size of packets that need to be stored on disk after truncation is 88 bytes.This means that even at maximum link utilization of 10 Gbit/s, only about 160 MByte/s need to be transferred to disk with this setup and packet distribution. However, due to heavy over-provisioning of the links measured, in reality the nodes rarely needed to store more than 35 MByte/s (280 Mbit/s) on disk during the MonNet measurement campaigns. Occasional traffic spikes can of course reach much higher throughput values, but these short spikes could successfully be buffered in the reserved main memory (512 MB).

8.4.2 Trace de-sensitization

After storing truncated packets on disks, the traces have been de-sensitized in offline fashion on the measurement nodes, since online pre-processing in real time is unfeasible due to computational speed. De-sensitization has therefore been carried out by batch jobs immediately after collection in order to minimize the storage time of unprocessed and privacy-sensitive traces.

As a first step in the de-sensitization process, the remaining payload beyond transport layer was removed using CAIDA's *CoralReef* [12] *crl_to_dag* utility. During this processing step, CoralReef also anonymized IP addresses in the remaining headers using the prefix-preserving *Crypto-PAn* [30]. A single, unique encryption-key was used throughout all MonNet measurement campaigns in order to allow tracing of specific IP addresses during the whole time period and for all measurements. This encryption key is kept secure and used for anonymization on the measurement nodes only.

8.4.3 Trace sanitization

Trace sanitization refers to the process of checking and ensuring that the collected traces are free from logical inconsistencies and are suitable for further analysis. This was done by using available tools such as the *dagtools* provided by Endace, accompanied by own tools for additional

consistency checks. These checks have been applied before and after each de-sensitization process. Resulting statistical figures such as byte and record numbers have been compared between consecutive passes of the sanitization procedures. In the common cases, when no inconsistencies or errors have been detected, the original, unprocessed traces have been deleted upon completion of the pre-processing procedures, and only de-sensitized and sanitized versions of the traces have been kept. If major errors such as packet loss have been detected, the pre-processing procedure has been stopped, the particular traces on both measurement nodes (for both directions) have been deleted and an immediate new collection has been scheduled automatically on both nodes. Detection of minor errors, such as single checksum inconsistencies, has been documented in the meta-data. For errors of unknown severity further steps have been postponed, requesting manual inspection. The sanitization included the following checks:

Major errors (discarding of traces)

- Are timestamps strictly monotonically increasing?
- Are timestamps in a reasonable time window?
- Are consecutive timestamps yielding feasible inter-arrival times according to line-speed and packet sizes?
- Are frames received continuously? (Packet arrival rates of zero packets/s should not happen on productive backbone links.)
- Are there any occurrences of identical IP headers within consecutive frames?
- Are all recorded frames of known type (i.e., POS with HDLC framing)?
- Is there an unreasonably high number of non-IP (v4 and v6) packets (which indicates garbled data)?
- Has the DAG reported loss of records during transfer to main memory (I/O bus limits)?
- Has the DAG reported packet loss or truncation due to insufficient buffer space?
- Are record counts before and after de-sensitization matching (i.e. have any packets been discarded)?

Minor errors (report in meta-data)

- Are there any IP header checksum errors?
- Have there been any receiver errors (i.e., link errors, such as incorrect light levels on the fiber and HDLC FCS (CRC) errors)?

Errors with unknown severity (manual inspection)

- Did the system log show any error messages during the measurements (e.g., by the measurement card or storage system)?
- Have there been any other internal errors reported in-line by the DAG card?

The sanitization process revealed some traces that had to be discarded due to garbled data or packet arrival rates of zero after a certain time, particularly on one measurement node. We suspect that this particular DAG card sometimes looses framing due to a hardware failure. Furthermore, infrequently the DAG cards discard single frames due to receiver errors, typically

HDLC CRC errors. Some frames can be reported as corrupted by the sanitization process due to IP checksum errors. Since the HDLC CRC was shown to be correct, there are cases when the IP checksum and CRC disagree [78]. Another explanation could be checksum errors already introduced by the sender, coupled with routers on the path ignoring the IP checksum in their validation of incoming IP packets and only performing incremental updates [79]. Since such missing or corrupted packets occur very rarely, the traces have still been used for analysis, but missing packets and IP checksum errors have been documented in the attached meta-data file.

Lessons learned:

1. Over-provisioning and packet truncation (as often required for privacy reasons anyhow) reduce hardware requirements and alleviate possible bottlenecks.

2. Thorough trace sanitization after collection and de-sensitization is important in order to avoid waste of computational resources and storage space. Furthermore it is imperative to ensure sound and unbiased scientific results during traffic analysis.

3. Collection circumstances (hardware, software, link, time) and pre-processing steps should be documented in meta-data and attached to the traces throughout the trace lifetime (from collection to archiving of the data).

4. Even if traffic data is sanitized, syntactical problems with individual packets need to be anticipated. This means that pre-processing and analysis tools need to be robustly designed in order to be able to handle all sorts of unknown protocols and packet header anomalies [27].

8.5 Data sharing policy

During the start-up phase, when the MonNet project was planned, vetted and later granted, we missed to establish a clear data sharing policy. After collecting the first traces and publishing results in scientific conferences and journals, other researchers identified our project as a possible resource of recent Internet data and asked for access to MonNet traffic traces. In absence of policies agreed upon by the network provider and the vetting committee, a "move code to data" approach was chosen, in which MonNet project members act as proxy (one level of indirection) between external researchers and the traffic traces.

Lessons learned: Data sharing is an essential part of scientific work, which needs to be explicitly considered already in early project phases. A technological and policy framework that might help future projects to implement secure data sharing is currently being suggested by Kenneally and claffy [17].

8.6 Traffic analysis and scientific results

So far, only the measurement processes including data pre-processing have been discussed. In this Section, the analysis approaches used to extract scientific results are outlined briefly in order to indicate the applicability and value of Internet measurements [80].

Packet-level analysis: In one of our early studies [81], Internet backbone traffic has been analyzed in order to extract cumulated statistical data into a database. The main challenge in this analysis program was to provide sufficient robustness, i.e., being able to deal with any possible kind of header inconsistency and anomaly. The resulting database consists of tables for specifically interesting features, such as IP header length, IP packet length, TCP options and different kinds of anomalous behavior, which could be analyzed conveniently with the help of SQL queries. A later follow up study [27] provided a more systematic listing of packet header anomalies in order to discuss potential security problems within Internet packet headers.

Flow-level analysis: In order to be able to conduct a detailed connection level analysis, the tightly synchronized unidirectional traces have been merged according to their timestamps. In the resulting bidirectional traces directional information for each frame was preserved in a special bit of the ERF trace format. As a next step, an analysis program collected per-flow information of the packet-level traces. Packet streams have then been summarized in flows by using a hash-table structure in memory. The gathered per-flow information includes packet and data counts for both directions, start- and end times, TCP flags and counters for erroneous packet headers and multiple occurrences of special flags like RST or FIN. This information was inserted into one database table for each transport protocol, each row representing a summary of exactly one flow or connection. The resulting flow database was used to study directional differences [9], increasing portions of UDP traffic [82] and routing symmetry [83] in Internet backbone traffic.

Traffic classification: To get a better understanding of traffic composition, different traffic classification methods have been studied [84]. A first approach to classify traffic on application level was done based on a set of heuristics regarding connection patterns of individual endpoints in the Internet [10]. The resulting classified flow table then allowed us to analyze and compare flow and connection characteristics among traffic of different network applications [11]. Recently, a classification approach based on statistical protocol properties has been suggested [85] and is currently further investigated and evaluated.

Lessons learned: Analysis of packet pevel-data often produces extensive result-sets, even if processed and aggregated. While many researchers and available analysis tools handle and process results-sets on file-level, our experience shows that it is advisable to exploit database systems (e.g., MySQL), since databases are designed to handle large data amounts and facilitate data-mining.

9 Summary and conclusions

The development of the Internet has without doubt not yet come to an end. In the next years, we can expect a continuing growth in user numbers and traffic volumes. Traffic will exhibit an even higher diversity, with the Internet becoming an even more unified backbone for all forms of communication and content (e.g., VoIP, IPTV). As a consequence, network bandwidths will continue to increase with at least the same pace as computer processing and storage capacities. However, the ability to keep up with link speeds will not be the only challenge for Internet measurement. There are a number of technical and commercial applications which could directly

benefit from results of Internet measurement and analysis, including network design and provisioning, improvement of network protocols and infrastructure but also network performance and accounting. Analysis of actual Internet traffic is also crucial input for network modeling and further development of network services. The Internet community will therefore have an increasing need for methods and means to collect, analyze, interpret and model Internet traffic.

The success and popularity of the Internet has unfortunately also lead to a rapid increase in all forms of misuse and unsocial, malicious activities - a trend, which is very likely to exacerbate as the importance of the Internet continues to grow. Network security measures, such as intrusion detection and prevention, are depending on profound understanding of traffic properties and have to rely on fast and reliable analysis methods of network anomalies and detection of vulnerabilities. Therefore research on modern, real-life datasets is vital for network security research in order to remain proactive.

Research on technologies and methods to monitor and measure Internet traffic are also of increasing legal relevance. With the data retention directive of the European Union [15], providers in member states will soon be required to retain connection data for periods of up to two years. While this directive could be postponed until March 2009, governments and operators currently need to establish the possibilities to execute the directive. This type of regulation obviously requires adequate technical solutions and know-how - which can both be provided by past, but also upcoming achievements of the Internet measurement and analysis community.

Analysis of Internet traffic is for obvious reasons heavily depending on the quality of existing network traces. It is therefore crucial to provide continuous possibilities to monitor and measure Internet traffic on as many sites as possible while at the same time maintaining respect for moral and ethical constraints. Acquiring network traces on backbone links, however, is a non-trivial task. Our experience shows that many problems can be avoided by careful and anticipatory planning. To facilitate the setting-up of future measurement projects, this paper is intended to serve as a guide for practical issues of Internet measurement based on lessons learned during the MonNet project. The paper addresses the main challenges of passive, large-scale measurements, including legal, ethical, technical and operational aspects. Furthermore, a detailed overview of the research field is given by describing different design choices and state-of-the-art solutions. This paper should provide researchers and practitioners with useful guidelines to setting up future monitoring infrastructures - which will in turn help to improve results from traffic analysis and therefore contribute to a better and more detailed understanding of how the Internet functions.

Acknowledgements

This work was supported by SUNET, the Swedish University Computer Network.

References

[1] R. Nelson, D. Lawson, and P. Lorier, "Analysis of Long Duration Traces," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 1, pp. 45–52, 2005.

[2] A. Householder, K. Houle, and C. Dougherty, "Computer Attack Trends Challenge Internet Security," *Computer*, vol. 35, no. 4, pp. 5–7, 2002.

[3] RIPE NCC, "YouTube Hijacking: A RIPE NCC RIS case study," http://www.ripe.net/news/study-youtube-hijacking.html (accessed 2009-10-27).

[4] S. McCanne and V. Jacobson, "The BSD Packet Filter: A New Architecture for User-level Packet Capture," in *USENIX Winter*, 1993, pp. 259–270.

[5] S. Ubik and P. Zejdl, "Passive Monitoring of 10 Gb/s Lines with PC Hardware," in *TNC: Terena Networking Conference*, Bruges, BE, 2008.

[6] R. Braden, "Requirements for Internet Hosts - Communication Layers," RFC 1122 (Standard), 1989.

[7] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "Simple Network Management Protocol (SNMP)," RFC 1157 (Historic), 1990.

[8] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), 2004.

[9] W. John and S. Tafvelin, "Differences between in- and outbound internet backbone traffic," in *TNC: Terena Networking Conference*, 2007.

[10] ——, "Heuristics to Classify Internet Backbone Traffic based on Connection Patterns," in *ICOIN: International Conference on Information Networking*, 2008, pp. 1–5.

[11] W. John, S. Tafvelin, and T. Olovsson, "Trends and Differences in Connection-Behavior within Classes of Internet Backbone Traffic," in *PAM: Passive and Active Network Measurement Conference*, 2008, pp. 192–201.

[12] K. Keys, D. Moore, R. Koga, E. Lagache, M. Tesch, and kc claffy, "The Architecture of CoralReef: an Internet Traffic Monitoring Software Suite," in *A workshop on Passive and Active Measurements*, 2001.

[13] "Directive 95/46/EC of the European Parilament and of the Council," 1995, http://ec.europa.eu/justice_home/fsj/privacy/docs/95-46-ce/dir1995-46_part1_en.pdf (accessed 2009-07-03).

[14] "Directive 2002/58/EC of the European Parilament and of the Council," 2002, http://eur-lex.europa.eu/LexUriServ/site/en/oj/2002/l_201/l_20120020731en00370047.pdf (accessed 2009-07-03).

[15] "Directive 2006/24/EC of the European Parilament and of the Council," 2006, http://eur-lex.europa.eu/LexUriServ/site/en/oj/2006/l_105/l_10520060413en00540063.pdf (accessed 2009-07-03).

[16] AK-Vorrat, "Overview of national data retention policies," https://wiki.vorratsdatenspeicherung.de/Transposition (accessed 2009-10-27).

[17] E. E. Kenneally and kc claffy, "An Internet Data Sharing Framework For Balancing Privacy and Utility," in *Proceedings of Engaging Data: First International Forum on the Application and Management of Personal Electronic Information*, 2009.

[18] D. C. Sicker, P. Ohm, and D. Grunwald, "Legal Issues Surrounding Monitoring During Network Research," in *IMC: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 141–148.

[19] "18 United States Code §2511," http://www4.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002511----000-.html (accessed 2009-07-03).

[20] "18 United States Code §3127," http://www4.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00003127----000-.html (accessed 2009-07-03).

[21] "18 uniTed States Code §2701," http://www4.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002701----000-.html (accessed 2009-07-03).

[22] "18 United States Code §2702," http://www4.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002702----000-.html (accessed 2009-07-03).

[23] "18 United States Code §2703," http://www4.law.cornell.edu/uscode/html/uscode18/usc_sec_18_00002703----000-.html (accessed 2009-07-03).

[24] k. claffy, "Ten Things Lawyers should Know about Internet Research," CAIDA,SDSC,UCSD, Tech. Rep., http://www.caida.org/publications/papers/2008/lawyers_top_ten/lawyers_top_ten.pdf (accessed 2009-07-03).

[25] ——, "Internet as Emerging Critical Infrastructure: What Needs to be Measured?" CAIDA, SDSC, UCSD, Tech. Rep., http://www.caida.org/publications/presentations/2008/uchile/uchile.pdf (accessed 2009-07-03).

[26] T. Karagiannis, A. Broido, N. Brownlee, k. claffy, and M. Faloutsos, "Is P2P Dying or Just Hiding?" in *IEEE GLOBECOM: Global Telecommunications Conference*, vol. Vol.3, Dallas, TX, USA, 2004, pp. 1532 – 8.

[27] W. John and T. Olovsson, "Detection of Malicious Traffic on Backbone Links via Packet Header Analysis," *Campus Wide Information Systems*, vol. 25, no. 5, pp. 342 – 358, 2008.

[28] S. Coull, C. Wright, F. Monrose, M. Collins, and M. Reiter, "Playing Devil's Advocate: Inferring Sensitive Information from Anonymized Network Traces," in *Proceedings of the Network and Distributed Systems Security Symposium*, San Diego, CA, USA, 2007.

[29] R. Pang, M. Allman, V. Paxson, and J. Lee, "The Devil and Packet Trace Anonymization," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 29–38, 2006.

[30] J. Xu, J. Fan, M. H. Ammar, and S. B. Moon, "Prefix-Preserving IP Address Anonymization: Measurement-Based Security Evaluation and a New Cryptography-Based Scheme," in *ICNP: Proceedings of the 10th IEEE International Conference on Network Protocols*, Washington, DC, USA, 2002, pp. 280–289.

[31] T. Ylonen, "Thoughts on How to Mount an Attack on Tcpdpriv's -a50 Option," Web White Paper, http://ita.ee.lbl.gov/html/contrib/attack50/attack50.html (accessed 2009-07-03).

[32] T. Kohno, A. Broido, and kc claffy, "Remote Physical Device Fingerprinting," *IEEE Trans. Dependable Secur. Comput.*, vol. 2, no. 2, pp. 93–108, 2005.

[33] M. Allman and V. Paxson, "Issues and etiquette concerning use of shared measurement data," in *IMC: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 135–140.

[34] "ACM Workshop on Network Data Anonymization," http://www.ics.forth.gr/~antonat/nda08.html (accessed 2009-07-03).

[35] G. Minshall, "TCPDPRIV: Program for Eliminating Confidential Information from Traces," http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html (accessed 2009-07-03).

[36] A. Slagell, J. Wang, and W. Yurcik, "Network Log Anonymization: Application of Crypto-Pan to Cisco Netflows," in *SKM: Proceedings of Workshop on Secure Knowledge Management*, Buffalo, NY, USA, 2004.

[37] R. Ramaswamy, N. Weng, and T. Wolf, "An IXA-Based Network Measurement Node," in *Proceedings of Intel IXA University Summit*, Hudson, MA, USA, 2004.

[38] T. Brekne and A. Årnes, "Circumventing IP-address Pseudonymization," in *Proceedings of the Third IASTED International Conference on Communications and Computer Networks*, Marina del Rey, CA, USA, 2005.

[39] Endace, "DAG Network Monitoring Cards," http://www.endace.com/our-products/dag-network-monitoring-cards/ (accessed 2009-07-03).

[40] Napatech, "Napatech Protocol and Traffic Analysis Network Adapter," http://www.napatech.com (accessed 2009-07-03).

[41] Invea-Tech, "COMBO Accelerated NIC Cards," http://www.invea-tech.com/solutions/packet-capture (accessed 2009-07-03).

[42] Endace, "Ninjaprobe 40G1," http://www.endace.com/ninjaprobe-40g1.html (accessed 2009-10-2).

[43] J. Zhang and A. Moore, "Traffic Trace Artifacts due to Monitoring Via Port Mirroring," in *E2EMON: Workshop on End-to-End Monitoring Techniques and Services*, 2007.

[44] V. Paxson, "Growth Trends in Wide-area TCP Connections," *Network, IEEE*, vol. 8, no. 4, pp. 8–17, Jul/Aug 1994.

[45] P. Phaal, S. Panchen, and N. McKee, "InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks," RFC 3176 (Informational), 2001.

[46] B.-Y. Choi, J. Park, and Z.-L. Zhang, "Adaptive Packet Sampling for Accurate and Scalable Flow Measurement," *IEEE GLOBECOM: Global Telecommunications Conference*, vol. 3, pp. 1448–1452 Vol.3, 29 Nov.-3 Dec. 2004.

[47] T.Zseby, M. Molina, N.Duffield, S.Niccolini, and F.Raspall, "Sampling and Filtering Techniques for IP Packet Selection," 2009, RFC 5475.

[48] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954, 2004.

[49] B.Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," 2008, RFC 5101.

[50] C. Estan and G. Varghese, "New Directions in Traffic Measurement and Accounting," in *SIGCOMM: Proceedings of the 2002 conference on Applications, technologies, architectures, and protocols for computer communications*, 2002, pp. 323–336.

[51] N. Duffield, C. Lund, and M. Thorup, "Properties and Prediction of Flow Statistics from Sampled Packet Streams," in *IMW: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, 2002, pp. 159–171.

[52] E. Cohen, N. Duffield, H. Kaplan, C. Lund, and M. Thorup, "Algorithms and Estimators for Accurate Summarization of Internet Traffic," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 265–278.

[53] M. C. Caballer and L. Zhan, "Compression of Internet Header Traces," Master Thesis, Chalmers University of Technology, Department of Computer Science and Engineering, Tech. Rep., 2006.

[54] V. Paxson, "Strategies for Sound Internet Measurement," in *IMC: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, 2004, pp. 263–271.

[55] J. Cleary, S. Donnelly, I. Graham, A. McGregor, and M. Pearson., "Design Principles for Accurate Passive Measurement," in *PAM: Proceedings of the Passive and Active Measurement Workshop*, 2000.

[56] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-Level Traffic Measurements from the Sprint IP Backbone," *IEEE Network*, vol. 17, no. 6, pp. 6–16, 2003.

[57] D. Mills, "Network Time Protocol (Version 3) Specification, Implementation and Analysis," RFC 1305 (Draft Standard), 1992.

[58] V. Paxson, "On Calibrating Measurements of Packet Transit Times," in *SIGMETRICS '98/PERFORMANCE '98: Proceedings of the 1998 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, 1998, pp. 11–21.

[59] J. Micheel, S. Donnelly, and I. Graham, "Precision Timestamping of Network Packets," in *IMW: Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, 2001, pp. 273–277.

[60] S. Donnelly, "Endace DAG Timestamping Whitepaper," 2007, endace, http://www.endace.com/ (accessed 2009-07-03).

[61] A. Pásztor and D. Veitch, "PC Based Precision Timing Without GPS," in *SIGMETRICS: Proceedings of the 2002 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, 2002, pp. 1–10.

[62] E. Technologies, "CDMA Network Time Server," http://www.endruntechnologies.com/pdf/TempusLxCDMA.pdf (accessed 2009-07-03).

[63] P. O. Hedekvist, R. Emardson, S.-C. Ebenhag, and K. Jaldehag, "Utilizing an Active Fiber Optic Communication Network for Accurate Time Distribution," *ICTON: 9th International Conference on Transparent Optical Networks*, vol. 1, pp. 50–53, 1-5 July 2007.

[64] S. Moon, P. Skelly, and D. Towsley, "Estimation and Removal of Clock Skew from Network Delay Measurements," *INFOCOM '99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, pp. 227–234, 1999.

[65] L. Zhang, Z. Liu, and C. Honghui Xia, "Clock Synchronization Algorithms for Network Measurements," *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, pp. 160–169, 2002.

[66] J. Wang, M. Zhou, and H. Zhou, "Clock Synchronization for Internet Measurements: a Clustering Algorithm," *Comput. Networks*, vol. 45, no. 6, pp. 731–741, 2004.

[67] Y. Lin, G. Kuo, H. Wang, S. Cheng, and S. Zou, "A Fuzzy-based Algorithm to Remove Clock Skew and Reset from One-way Delay Measurement," *IEEE GLOBECOM: Global Telecommunications Conference*, vol. 3, pp. 1425–1430 Vol.3, 2004.

[68] D. DuBois and H. Prade, *Fuzzy Sets and Systems: Theory and Applications*. Academic Press, 1980.

[69] H. Khlifi and J.-C. Grégoire, "Low-complexity Offline and Online Clock Skew Estimation and Removal," *Comput. Networks*, vol. 50, no. 11, pp. 1872–1884, 2006.

[70] C. Shannon, D. Moore, K. Keys, M. Fomenkov, B. Huffaker, and k claffy, "The Internet Measurement Data Catalog," *SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 5, pp. 97–100, 2005.

[71] M. Allman, E. Blanron, and W. Eddy, "A Scalable System for Sharing Internet Measurement," in *PAM: Passive & Active Measurement Workshop*, 2002.

[72] CAIDA, "DatCat: Internet Measurement Data Catalog," http://imdc.datcat.org/ (accessed 2009-07-03).

[73] J. Mogul, "Trace Anonymization Misses the Point," 2002, WWW 2002 Panel on Web Measurements, http://www2002.org/presentations/mogul-n.pdf (accessed 2009-07-03).

[74] J. Mirkovic, "Privacy-safe Network Trace Sharing via Secure Queries," in *Proceedings of the 1st ACM workshop on Network data anonymization*, 2008.

[75] A. Parate and G. Miklau, "A Framework for Safely Publishing Communication Traces," in *CIKM: Conference on Information and Knowledge Management*, 2009.

[76] S. Kang, N. Aycirieix, H. Kwak, S. Kim, and S. Moon, "CASFI Data Sharing Platform," in *PAM: Passive and Active Network Measurement Conference, Student Workshop*, 2009.

[77] J. Apisdorf, kc claffy, K. Thompson, and R. Wilder, "OC3MON: Flexible, Affordable, High Performance Staistics Collection," in *LISA '96: Proceedings of the 10th USENIX conference on System administration*, Berkeley, CA, USA, 1996.

[78] J. Stone and C. Partridge, "When the CRC and TCP Checksum Disagree," in *SIGCOMM: Conference on Applications, Technologies, Architectures and Protocols for Computer Communication*, 2000.

[79] A. Rijsinghani, "Computation of the Internet Checksum via Incremental Update," RFC 1624, 1994.

[80] W. John, "On Measurement and Analysis of Internet Backbone Traffic," Licentiate Thesis, Department of Computer Science and Engineering, Chalmers University of Technology, Göteborg, SE, Tech. Rep., 2008, ISSN 1652-076X, Technical Report 50L.

[81] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic and Header Anomalies Observed," in *IMC: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 111–116.

[82] M. Zhang, M. Dusi, W. John, and C. Chen, "Analysis of UDP Traffic Usage on Internet Backbone Links," in *SAINT: 9th Annual International Symposium on Applications and the Internet*, 2009.

[83] M. Dusi and W. John, "Observing Routing Asymmetry in Internet Traffic," 2009, http://www.caida.org/research/traffic-analysis/asymmetry/, (accessed 2009-10-27).

[84] M. Zhang, W. John, kc claffy, and N. Brownlee, "State of the Art in Traffic Classification: A Research Overview," in *PAM: Passive and Active Network Measurement Conference, Student Workshop*, 2009.

[85] E. Hjelmvik and W. John, "Statistical Protocol IDentification with SPID: Preliminary Results," in *SNCNW: 6th Swedish National Computer Networking Workshop*, 2009.

PAPER II

Wolfgang John and Sven Tafvelin

Analysis of Internet Backbone Traffic and Anomalies Observed

IMC '07: Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement
San Diego, California, USA, 2007

Analysis of Internet Backbone Traffic and Header Anomalies Observed

Wolfgang John and Sven Tafvelin

Department of Computer Science and Engineering
Chalmers University of Technology, Göteborg, Sweden
{firstname.lastname}@chalmers.se

Abstract

The dominating Internet protocols, IP and TCP, allow some flexibility in implementation, including a variety of optional features. To support research and further development of these protocols, it is crucial to know about current deployment of protocol specific features and accompanying anomalies. This work is intended to reflect the current characteristics of Internet backbone traffic and point out misbehaviors and potential problems. On 20 consecutive days in April 2006 bidirectional traffic was collected on an OC-192 backbone link. The analysis of the data provides a comprehensive summary about current protocol usage including comparisons to prior studies. Furthermore, header misbehaviors and anomalies were found within almost every aspect analyzed and are discussed in detail. These observations are important information for designers of network protocols, network application and network attack detection systems.

1 Introduction

Today, the Internet has emerged as the key component for commercial and personal communication. One contributing factor to the still ongoing expansion of the Internet is its versatility and flexibility. Applications and protocols keep changing not only with time [1], but also within geographical locations. Unfortunately, this fast development has left little time or resources to integrate measurement and analysis possibilities into the Internet infrastructure. However, the Internet community needs to understand the nature of Internet traffic in order to support research and further development [2]. It is also important to know about current deployment of protocol specific features and possible misuse. This knowledge is especially relevant in order to improve the robustness of protocol implementations and network applications, since increasing bandwidth and growing numbers of Internet users also lead to increased misuse and anomalous behavior [3]. One way of acquiring better understanding is to measure and analyze real Internet traffic, preferably on highly aggregated links. The resulting comprehensive view is crucial for a better understanding of the applied technology and protocols and hence for the future development thereof. This is important for establishing simulation models [4] and will also bring up new insights for related research fields, such as network security or intrusion detection.

A number of studies on protocol specific features have been published earlier, based on a variety of datasets. Thompson et al. [5] presented wide-area Internet traffic characteristics on data recorded on OC-3 traffic monitors in 1997, including figures about packet size distribution and transport protocol decomposition. McCreary et al. [1] provided a longitudinal analysis of Internet traffic based on data collected on an OC3 link of the Ames Internet exchange in 1999 to 2000. Fractions of fragmented traffic were presented and the usage of Path MTU Discovery was inferred based on the packet size distribution. Shannon et al. [6] studied frequency and cause of IP fragmented traffic on data collected on different WANs (100Mbit Ethernet, OC3 and OC12) in March 2001. Fraleigh et al. [7] analyzed traffic measurements from the Sprint IP backbone, based on a number of traces taken on different OC12 and OC48 links in 2001-2002. Pentikousis et al. [8] indirectly quantified deployment of TCP options based on traces with incomplete header information. The data was collected between October 2003 and January 2004 on a number of OC3 and OC12 links by the NLANR/PMA. In that paper, recent figures about packet size distributions were presented as well. Already earlier, Allman [9] presented observations about usage of TCP options within traffic from a particular webserver in a one and a half year period from 1998-2000. Finally, in his investigations about the evolution of transport protocols, Medina et al. [10] presented usage of TCP features like ECN (RFC 3168) based on passive measurements on a local webserver during two weeks in February 2004.

Despite these existing studies, there is a need for further measurement studies [2, 11]. Continued analysis work needs to be done on updated real-world data in order to be able to follow trends and changes in network characteristics. Therefore, in this work we will consequently continue to analyze IP and TCP, as they are the most common protocols used in today's Internet, and compare the results to previous work. After describtion of the analyzed data in Section 2, we present our results for IP and TCP specific features in Section 3. Finally, Section 4 summarizes the main findings and draws conclusions.

2 Methodology

2.1 Collection of Traces

The traffic traces have been collected on the outermost part of an SDH ring running Packet over SONET (PoS). The traffic passing the ring to (outgoing) and from (incoming) the Internet is primarily routed via our tapped links. This expected behavior is confirmed by SNMP statistics showing a difference of almost an order of magnitude between the tapped link and the protection link. Simplified, we regard the measurements to be taken on links between the region of Göteborg, including exchange traffic with the regional access point, and the rest of the Internet.

On the two OC-192 links (two directions) we use optical splitters attached to two Endace DAG6.2SE cards. The DAG cards captured the first 120 bytes of each frame to ensure that the entire network and transport header information is preserved. The data collection was performed between the 7th of April 2006, 2AM and the 26th of April 2006, 10AM. During this period, we simultaneously for both directions collected four traces of 20 minutes each day at identical times. The times (2AM, 10AM, 2PM, 8PM) were chosen to cover business, non-business and nighttime hours. Due to measurement errors in one direction at four occasions we have excluded these traces and the corresponding traces in the opposite direction.

2.2 Processing and Analysis

After storing the data on disk, the payload beyond transport layer was removed and the traces were sanitized and desensitized. This was mainly done by using available tools like Endace's *dagtools* and CAIDA's *CoralReef*, accompanied by own tools for additional consistency checks, which have been applied after each preprocessing step to ensure sanity of the traces. Trace sanitization refers to the process of checking and ensuring that the collected traces are free from logical inconsistencies and are suitable for further analysis. During our capturing sessions, the DAG cards discarded a total of 20 frames within 12 different traces due to receiver errors or HDLC CRC errors. Another 71 frames within 30 different traces had to be discarded after the sanitization process due to IP checksum errors.

By desensitization the removing of all sensitive information to ensure privacy and confidentiality is meant. The payload of the packets was removed earlier, so we finally anonymized IP addresses using the prefix preserving CryptoPAN [12]. After desensitization, the traces were moved to a central storage. An analysis program was run on the data to extract cummulated statistical data into a database. For packets of special interest, corresponding TCP flows have been extracted.

3 Results

The 148 traces analyzed sum up to 10.77 billion PoS frames, containing a total of 7.6 TB of data. 99.97% of the frames contain IPv4 packets, summing up to 99.99% of the carried data. The remaining traffic consists of different routing protocols (BGP, CLNP, CDP). The results in the remainder of this paper are based on IPv4 traffic only.

3.1 General Traffic Properties

3.1.1 IP packet size distribution

In earlier measurements, cumulative distribution of IPv4 packet lengths was reported to be trimodal, showing major modes at small packet sizes just above 40 bytes (TCP acknowledgments), large packets around 1500 bytes (Ethernet MTU) and default datagram sizes of 576 bytes according to RFC 879. Data collected between 1997 and 2002 reported about fractions of default datagram sizes from 10% up to 40% [5, 1, 6, 7]. Pentikousis et al. [8] however showed in 2004, that packet size distribution was no longer trimodal, but rather bimodal, with default datagram sizes accounting for only 3.8% of all packets.

Fig. 1 illustrates the cumulative distribution of IPv4 packet lengths in our traces of 2006. The distribution is still bimodal, with the major portion of lengths between 40 and 100 bytes and between 1400 and 1500 bytes (44% and 37% of all IPv4 packets, resp.). The usage of the default datagram size of 576 byte was further decreased to a fraction of only 0.95%, now not even being among the first three most significant modes anymore. This is caused by the predominance of Path MTU Discovery in today's TCP implementations, which is confirmed later by the analysis of the IP flags and the TCP maximum segment size (MSS) option. On the other hand, two other notable modes appeared at 628 bytes and 1300 bytes, representing 1.76% and 1.1% of the IPv4 traffic, resp.

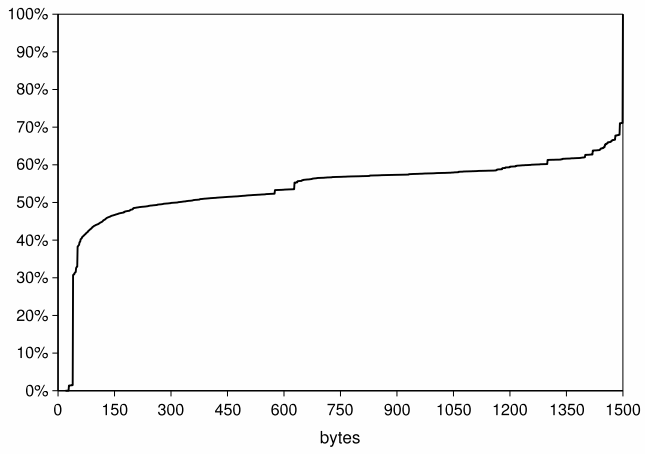


Figure 1: Cum. IPv4 Packet Size Distribution

An analysis of TCP flows including a lot of 628 byte packets showed that these packets typically appear after full sized packets (MSS of 1460), often with the PUSH flag set. We suspect that they are sent by applications doing 'TCP layer fragmentation' on 2KB blocks of data, indicating the end of data a data block by PUSH. This is confirmed by flows where smaller MSS values have been negotiated (e.g. 1452). In this cases, the following packets became larger (e.g. 636 bytes) to add up to 2048 bytes of payload again. Examples for applications using such 2KB blocks for data transfer can be found in [13], where different file-sharing protocols using fixed block sizes are presented. A look at the TCP destination ports revealed that large fractions of this traffic are indeed sent to ports known to be used for popular file-sharing protocols like Bittorrent and DirectConnect. The notable step at 1300 bytes on the other hand could be explained by the recommended IP MTU for IPsec VPN tunnels [14].

Packets larger than 1500 bytes (Ethernet MTU) aggregate a fraction of 0.15%. Traffic of packets sized up to 8192 bytes was observed, but the major part (99.7%) accounts for packet sizes of 4470 bytes. A minor part of the >1500 byte sized packets represents BGP updates between backbone- or access routers. The majority of the large packet traffic (mainly 4470) could after thorough investigation be identified as customized data-transfer from a space observatory to a data center using jumbo-packets over Ethernet.

3.1.2 Transport protocols

The protocol breakdown in Table 1(a) once more confirms the dominance of TCP traffic. Compared to earlier measurements reporting about TCP accounting for around 90 - 95% of the data volume and for around 85-90% of IP packets, [5, 1, 6, 7], both fractions seem to be slightly larger in the analyzed SUNET data. In Table 1(a), the fractions of cumulated packets and bytes carried in the respective protocol are given in percent of the total IPv4 traffic for the corresponding time.

An interesting observation can be made at the 2PM data. Here, the largest fraction of TCP and the lowest of UDP packets appear. A closer look at the differences between outgoing and incoming traffic revealed that three consecutive measurements on the outgoing link carried up to

(a) IPv4 Protocol Breakdown (values in %)

	2AM		10AM		2PM		8PM	
	Pkts	Data	Pkts	Data	Pkts	Data	Pkts	Data
TCP	91.3	97.6	91.5	96.8	93.2	97.1	91.4	97.2
UDP	8.5	2.3	7.6	2.8	6.1	2.7	8.3	2.7
ICMP	0.2	0.02	0.19	0.02	0.20	0.02	0.12	0.01
ESP	0.01	0.00	0.47	0.19	0.35	0.14	0.02	0.02
GRE	0.01	0.01	0.08	0.08	0.04	0.03	0.06	0.04

(b) UDP Burst (values in %)

OUTGOING UDP			
Date	Time	Packets	Data
2006-04-16	2PM	6.8	1.7
2006-04-16	8PM	40.6	5.1
2006-04-17	2AM	51.9	6.1
2006-04-17	10AM	58.1	7.1
2006-04-17	2PM	5.7	1.8

Table 1: Transport Protocols

58% UDP packets, not covering the 2PM traces, as shown in Table 1(b). These figures indicate a potential UDP burst of 14-24 hours of time. A detailed analysis showed that the packet length for the UDP packets causing the burst was just 29 bytes, leaving a single byte for UDP payload data. These packets were transmitted between a single sender and receiver address with varying port numbers. After reporting this network anomaly, the network support group of a University confirmed that the burst stemed from an UDP DoS script installed undetected on a webserver with a known vulnerability. Although TCP data was still predominant, a dominance of UDP packets over such a timespan could potentially lead to TCP starvation and raise serious concerns about Internet stability and fairness.

3.2 Analysis of IP Properties

3.2.1 IP type of service

The TOS field can optionally include codepoints for Explicit Congestion Notification (ECN) and Differentiated Services. 83.1% of the observed IPv4 packets store a value of zero in the TOS field, not applying the mechanisms above. Valid 'Pool 1' DiffServ Codepoints (RFC 2474) account for 16.8% of all TOS fields.

Medina et al. [10] reported about almost an doubling of ECN capable webservers from 1.1% in 2000 to 2.1% in 2004, but indicates that routers or middleboxes might erase ECT codepoints. In our data only 1.0 million IPv4 packets provide ECN capable transport (either one of the ECT bits set) and additionally 1.1 million packets actually show 'congestion experienced' (both bits set). This means that ECN is implemented in only around 0.02% of the IPv4 traffic. These numbers are consistent with the observations by Pentikousis et al. [8], suggesting that the number of ECN-aware routers is still very small.

3.2.2 IP Options

The analysis of IP options showed that they are virtually not used. Only 68 packets carrying IP options were observed. One 20-minute trace contained 45 packets with IP option 7 (Record Route) and 3 traces carried up to 12 packets with IP option 148 (Router Alert).

3.2.3 IP fragmentation

During the year 2000, McCreary et al. [1] observed an increase in the fraction IP packets carrying fragmented traffic from 0.03% to 0.15%. Indeed, one year later, Shannon et al. [6] reported fractions of fragmented traffic of up to 0.67%. Contrary to this trend, we found a much smaller fraction of 0.06% of fragmented traffic in the analyzed data. Even though these numbers are relatively small, there is still an order of magnitude difference between earlier and current results. 72% of the fragmented traffic in our data is transmitted during office hours, at 10AM and 2PM. We also observed that the amount of fragmented traffic on the incoming link is about 9 times higher than on the outgoing one.

While UDP and TCP are responsible for 97% and 3% respectively of all incoming fragmented segments, they just represent 19% and 18% of the outgoing. The remaining 63% of the outgoing fragmented traffic turned out to be IPsec ESP traffic (RFC 4303), observed between exactly one source and one receiver during working hours on weekdays. Each fragment series in this connection consists of one full length Ethernet MTU and one additional 72 byte fragment. This can easily be explained by an unsuitably configured host/VPN combination transmitting 1532 bytes (1572 - 40 bytes IP and TCP header) instead of the Ethernet MTU due to the additional ESP header. The dominance of UDP among fragmented traffic is not surprising, since Path MTU Discovery is a TCP feature only.

The first packets in all observed fragment series are in 96.7% sized larger or equal than 1300 bytes. This goes along with the assumption that fragments are sent in-order and the first segments should be full sized MTUs. It should be noted that 1.6% of first packets in fragment series are smaller than 576 bytes. This is not surprising, considering an earlier observation by Shannon et al. [6] that about 8% of fragment series are sent in reverse-order, sending the smallest segment first. This is accepted behavior, since the IP specification (RFC 791) does not prescribe any sizes of fragments. Another reason for small first segments are misconfigured networks or deliberate use of small MTUs, like serial links (RFC 1144) connected to the backbone. An example for such unusual small sized fragments of only 244 bytes will be given in the next subsection.

3.2.4 IP flags

The analysis of the IP flags (fragment bits) revealed that 91.3% of all observed IP packets have the don't fragment bit (DF) set, as proposed by Path MTU Discovery (RFC 1191). 8.65% use neither DF nor MF (more fragments) and 0.04% set solely the MF bit.

Following the IP specification (RFC 791) no other values are valid in the IP flag field. Nevertheless, we observed a total of 27,474 IPv4 packets from 70 distinct IP sources with DF

and MF set simultaneously. About 35 of those invalid bit values are evenly observed among both directions in all traces, with exception of one burst of 21,768 packets in a trace of the incoming link. This burst stems from a 10 minutes long TCP flow between a local server on port 49999 and a remote client on the gaming port 1737 (UltimaD). Surprisingly, all the incoming traffic is fragmented to series of 244 byte long IP packets. The data carried by these fragment series adds up to full Ethernet MTUs size. Because being fragmented, each but the last fragment in a series has the MF bit set. Disregarding its actual fragmentation, each fragment also has the DF bit set. A similar behavior could be observed on the outgoing link, where one source generates 85% of all outgoing DF+MF packets, evenly distributed over 70 out of 76 measured times. Again, each IP packet has the DF bit set by default, while MF is set additionally when fragmentation is needed. Looking at the traffic pattern and considering that UDP port 53 is used, it seems to be obvious that there is a DNS server using improper protocol stacks inside the Göteborg region.

Additionally, we observed a total of 233 cases of a reserved bit with value 1, appearing in small numbers in most of the collected traces and stemming from 126 distinct sources. According to the IP standard (RFC 791) the reserved bit must be zero, so this behavior has to be regarded as misbehavior.

3.3 Analysis of TCP Properties

3.3.1 TCP Options

In an early study, Allman [9] reported about portions of hosts applying the Window Scale (WS) and Timestamp (TS) options, both increasing from about 15% to 20% during a 15 month period from 1998 to 2000. The SACK permitted option was shown to increase even further from 7% to 40%. No numbers for hosts applying the MMS option were given. The more recent approach to quantify TCP option deployment by Pentikousis et al. in 2004 [8] was unfortunately carried out on traces with incomplete header information. Since TCP option data was not available in these traces, their deployment had consequently to be analyzed indirectly. Our results, based on traces including complete header information, show that this indirect approach yielded quite accurate results.

Table 2(a) shows the deployment of the most important TCP options as fractions of the SYN and SYN/ACK segments, divided into summaries of the four times each day. The results show that MSS and SACK permitted options are widely used during connection establishment (on average 99.2% and 89.9% resp.). The positive trend of the SACK option deployment, as indicated by Allman, was obviously continued and the inferred values of Pentikousis et al. are finally confirmed. The frequent usage of the MSS option again indicates the dominance of Path MTU Discovery in TCP connections, since an advertised MSS is the precondition for this technique. The WS and TS options on the other hand are still applied to the same extent as in 2000 (17.9% and 14.5% resp.). In Table 2(b) the occurrence of TCP options with respect to all TCP segments is summarized. Around 87% of the TCP segments do not carry any options at all. Only an average of 2.9% of all segments actually applies the SACK opportunity, which was permitted by around 90% of all connections. It is interesting, that although 15.5% of the connection establishments advertise usage of the TS option, it just reappears in 9.3% of all

(a) TCP Options in SYN segments

Kind	2AM	10AM	2PM	8PM
2(MSS)	99.0%	98.7%	99.7%	99.1%
3(WS)	21.4%	18.4%	16.6%	16.5%
4(SACK perm.)	91.0%	86.6%	88.9%	89.8%
8(TS)	18.2%	15.3%	13.3%	12.8%

(b) TCP Options in all segments

Kind	2AM	10AM	2PM	8PM
No Opt.	86.5%	85.2%	87.3%	88.6%
5(SACK)	3.1%	2.8%	2.9%	3.1%
8(TS)	9.7%	11.2%	9.0%	7.6%
19(MD5)	0.02%	0.02%	0.01%	0.01%

Table 2: TCP Option Deployment

segments. This might be caused by TCP servers not responding with the TS option set in their initial SYN/ACK. All other option kinds were observed with very low frequency.

3.3.2 TCP option values

Allman [9] reported about 90% of connections advertising an MSS of about 1460 bytes in the SYN segment, leaving 6% for larger MSS values, and another 5% for MSS values of about 500 bytes. An analysis of advertised values within the MSS option field in our data revealed that the major portion (93.7%) of the MSS values still lies between 1400-1460 bytes, thus close to the Ethernet maximum (1500-40 byte for IP and TCP headers). Values larger than 1460 bytes are carried by only 0.06% of the MSS options, with values up to the maximum of 65535. Values smaller than 536 bytes (the default IP datagram size minus 40) are carried by another tiny fraction (0.05%), including MSS values down to zero. The 53,280 packets carrying small MSS values are send by 2931 different IP addresses. The major fraction of the $<$536 MSS values carries a value of 512 (87.5%), followed by 64 (2.4%) and 260 (1.3%). Values down to 265 bytes can be explained by standard active fingerprinting attacks, like nmap [15], whereas smaller values are more likely to be DoS exploits.

In Allman's data from 2000, Window Scale (WS) factors as high as 12 appeared, with zero as the main factor, accounting for 84%, followed by a factor of one with about 15%. In our contemporary data, WS factor values appear in the range of 0 to 14. The most common scale factor with 58% is zero, which should not be interpreted as real factor, but as an offer to scale while scaling the own receive window by 1. The major real scale factor appears to be 2, with 30.8% deployment. Other scale factors in recognizable fractions are 3, 1, and 6, applied in respectively 4.2%, 4.1% and 1.0% of all segments carrying a WS option. As a general observation, the WS option is applied much more effective now, most probably due to bandwidth increases and larger data transfers. A detailed look at diurnal behavior of WS option values revealed that traces at nighttime (2AM) carry constantly about 10% more scale factor values of 2, compensated by around 10% less factors of zero.

3.3.3 TCP option misbehavior

Connected to the analysis of TCP options, a couple of anomalies were encountered (Table 3(a)). The table shows only counts of packets, since the relative fractions are too small compared to the amount of total TCP segments. It should be mentioned that the differences between outgoing and incoming traffic lie typically in the order of a magnitude. Also diurnal differences can be observed, with non-working hours (2AM and 8PM) responsible for 67% of all reported anomalies.

(a) TCP options and header lengths

Anomaly	2AM	10AM	2PM	8PM
Undef.Kind	1062	507	413	388
Invalid OL	1200	399	915	3020
Invalid HL	71	528	130	119

(b) TCP flags

Anomaly	2AM	10AM	2PM	8PM
RST+SYN+FIN	8	35	11	15
RST+SYN	25	70	43	27
SYN+FIN	4	22	8	9
Zero Flags	32	78	86	90
RST+FIN	10200	10988	14320	16334

Table 3: Anomalies in TCP Headers

The first misbehavior experienced was the occurrence of undefined option types. Out of the 8bit range for TCP option kinds, only 26 are defined. From the remaining types almost all (228) have been observed. 522 distinct sources sent the 2370 undefined options observed, with 85% appearing on the incoming link. One single source sent 42% of these packets during the 20 minutes duration of one measurement at 2AM. Usage of a single destination port and 8200 different destination hosts within a one network prefix clearly indicate a scanning attack, even though only a minor fraction (6%) of the scanning traffic actually showed undefined options. The malformed packets carried instead of {MSS, NOP, NOP, SACK perm.} the option sequence of {MSS, random byte, random byte, 0, 0}. It seems likely that it is indeed the scanning software which is buggy and generates occasional malformed packets.

Another inconsistency encountered are option headers appearing to be valid while carrying option lengths that do not correspond to the total header length in the regular TCP header. 98.2% of the 5534 cases happened on the incoming link, with two sources responsible for 45% and 22% of such anomalous headers. The first source adds a SACK option with constant pattern to the TCP header, declaring an option header length of 180 bytes. This source was observed at 4 different days. The second source applies valid TCP options including an MSS value of 1460 during connection setup in SYN/ACK packets. However, also in the proceeding data packets an option of type 2 (MSS) appears, but this time followed by zeros, and thereby consequently advertising an option length of zero. According to the traffic pattern this source was a webserver. In total, 259 unique sources of this anomaly have been identified.

Finally, 848 TCP segments advertising header length values of less than 20 bytes were generated by 184 distinct sources, probably being DOS exploits. Again, the major fraction (91.3%) was observed in incoming traffic. 81.5% of the invalid values advertised a TCP header of zero length. The remaining 18.5% are evenly distributed between the remaining possible length values (in multiples of 4). The main source of zero byte TCP headers sends 351 such packets during a period of at at least 20 minutes. 351 unique destinations for 351 packets indicate a scanning campaign, this time to some well-known source port numbers (21, 23, 110, 80, 8080).

3.3.4 TCP Flags

Analyzing the TCP flag field, 10,972 ECN-setup SYN packets and just 800 ECN-setup SYN/ACK segments (RFC 3168) have been observed. The small numbers are consistent with earlier observations by Medina et al. [10], where only 0.2% of tested web clients advertise ECN capabilities. In section 3.2.1 we identified around 2.1 million ECN capable IP packets. This indicates that the few ECN enabled TCP connections represent large flows.

The urgent flag (URG) was set in only 663 segments. The acknowledgment flag (ACK) on the other hand was set in 98.6% of all segments, which is expected, since theoretically only the initial SYN packets should not carry an ACK flag. The push bit (PSH) was enabled in 22.4% of all segments.

In Table 3(b) we present packet counts for unexpected combinations of connection flags. The four first-listed anomalies have been seen in packets sent by 56 distinct sources. Such inconsistencies can easily be generated by port scanning tools like nmap [15]. We can rule out T/TCP as reason for SYN+FIN packets, since none of the 43 packets carried CC options (RFC 1644). The most frequent anomaly is connection termination with both, FIN and RST flags set. This was seen in 51,842 segments, send by 7576 unique source IP addresses. All connection flag anomalies are spread quite evenly over all measurements, with no particular sources to stand out.

4 Summary and Conclusions

In order to be able to present contemporary characteristics of Internet traffic, 148 traces of 20 minutes duration have been collected on a pair of backbone links in April 2006. The analysis revealed that IP options are virtually not applied and IP fragmentation is done to a minor extent (0.06%), with UDP accounting for most IP fragments. The latter observation stems from an increased employment of TCP Path MTU Discovery, which was shown to be even more dominating than reported earlier. Regarding packet size distribution, two findings should be noted. First, IP packet lengths of 628 bytes have become even more common than the default datagram size, with P2P traffic identified as likely source. Second, except for router traffic, jumbo packets are used for a single custom application only and are not seen otherwise. Even though these observations are limited to our measurements from a single point in the Internet, this summary about current behavior of network protocols helps to understand the influence of additional protocol features on Internet traffic and can contribute to an improvement of future simulation models.

Additionally, a number of anomalies and inconsistencies have been observed, serving as pointers to keep in mind for protocol and application developers. As one cause for the otherwise rare occurrence of IP fragmentation additional headers introduced by VPN have been identified, advising application developers to use smaller MSS values. Furthermore, one single long-duration UDP burst was observed while gathering protocol statistics. This was found to be an UDP DoS attack, undetected by the network management tools in operation. This indicates the need for continuous refinement of network monitoring policies. The magnitude of the burst also raises stability and fairness concerns, calling for addition of some kind of congestion control to UDP. Finally, several types of misbehaviors within IP and TCP headers have been discussed. The anomalies observed could be explained by three different causes:

- Buggy or misbehaving applications or protocol stacks
- Active OS fingerprinting [13]
- Network attacks exploiting protocol vulnerabilities

Even though all header anomalies observed are rare compared to the total number packets, their existence shows that developers need to carefully design implementations. Almost any possible inconsistency in protocol headers will appear eventually, thus network protocols and applications have to be designed and implemented as robust as possible, leaving no vulnerabilities.

Since access to traffic on highly aggregated links is still uncommon for researchers working on network security, our results form valuable input to related research on topics like large scale intrusion detection or traffic filtering. Besides quantifying the occurrence of different header anomalies 'in the wild', the results yield explanations for the origins of these commonly observed inconsistencies. Not every malformed packet header is necessarily part of attacking traffic, as proven by the example of the DNS server setting invalid fragmentation bits, but can also be introduced by improper protocol stacks. This information can be relevant when refining rule-sets for traffic filters, as applied in firewalls or network intrusion detection systems. Furthermore, knowledge about the nature of header anomalies can be interesting for researchers or developers creating stress tests for routers and other network components.

5 Acknowledgements

The authors want to thank Pierre Kleberger for his kind technical support and Tomas Olovsson for his valuable and constructive comments throughout the MonNet project.

References

[1] S. McCreary and kc claffy, "Trends in Wide Area IP Traffic Patterns - A View from Ames Internet Exchange," CAIDA, San Diego Supercomputer Center, Tech. Rep., 2000.

[2] N. Brownlee and kc claffy, "Internet Measurement," *IEEE Internet Computing*, vol. 08, no. 5, pp. 30–33, 2004.

[3] A. Householder, K. Houle, and C. Dougherty, "Computer Attack Trends Challenge Internet Security," *Computer*, vol. 35, no. 4, pp. 5–7, 2002.

[4] S. Floyd and E. Kohler, "Internet Research Needs Better Models," ser. Comput. Commun. Rev. (USA), vol. 33, 2003, pp. 29–34.

[5] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet Traffic Patterns and Characteristics," *IEEE Network*, vol. 11, no. 6, pp. 10–23, 1997.

[6] C. Shannon, D. Moore, and kc claffy, "Beyond Folklore: Observations on Fragmented Traffic," *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 709–20, 2002.

[7] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and C. Diot, "Packet-level Traffic Measurements from the Sprint IP Backbone," *Network, IEEE*, vol. 17, no. 6, pp. 6–16, 2003.

[8] K. Pentikousis and H. Badr, "Quantifying the Deployment of TCP Options - a Comparative Study," *IEEE Communications Letters*, vol. 8, no. 10, pp. 647–9, 2004.

[9] M. Allman, "A Web Server's View of the Transport Layer," *SIGCOMM Comput. Commun. Rev.*, vol. 30, no. 5, 2000.

[10] A. Medina, M. Allman, and S. Floyd, "Measuring the Evolution of Transport Protocols in the Internet," *Computer Communication Review*, vol. 35, no. 2, pp. 37–51, 2005.

[11] A. Hussain, G. Bartlett, Y. Pryadkin, J. Heidemann, C. Papadopoulos, and J. Bannister, "Experiences with a Continuous Network Tracing Infrastructure," in *MineNet: ACM SIGCOMM workshop on Mining network data*. New York, NY, USA: ACM Press, 2005.

[12] J. Xu, J. Fan, M. Ammar, and S. Moon, "On the Design and Performance of Prefix-preserving IP Traffic Trace Anonymization," in *ACM Workshop on Internet Measurement*, 2001.

[13] T. Karagiannis, A. Broido, N. Brownlee, kc claffy, and M. Faloutsos, "File-sharing in the Internet: A Characterization of P2P Traffic in the Backbone," UCR, Tech. Rep., 2003.

[14] CiscoSystems, "IPsec VPN WAN Design Overview," Cisco Doc., 2006, accessed 2006-12-05. [Online]. Available: http://www.cisco.com/univercd/cc/td/doc/solution/ipsecov.pdf

[15] Fyodor, "Nmap Security Scanner," 1998, accessed 2006-12-05. [Online]. Available: http://insecure.org/nmap/index.html

PAPER III

Wolfgang John and Tomas Olovsson

Detection of Malicous Traffic on Backbone Links via Packet Header Analysis

Campus-Wide Information Systems
Vol. 25(5), Emerald, 2008

Detection of Malicious Traffic on Backbone Links via Packet Header Analysis

Wolfgang John and Tomas Olovsson

Department of Computer Science and Engineering,
Chalmers University of Technology, Göteborg, SE
{firstname.lastname}@chalmers.se

Abstract

Purpose: In this study, modern Internet backbone traffic has been investigated in order to study occurrences of malicious activities and potential security problems within Internet packet headers.

Design/Methodology/Approach: Contemporary and highly aggregated backbone data has been analyzed regarding consistency of network and transport layer headers (i.e., IP, TCP, UDP and ICMP). Possible security implications of each anomaly observed are discussed.

Findings: A systematic listing of packet header anomalies together with their frequencies as seen "in the wild" is provided. Inconsistencies in protocol headers have been found within almost every aspect analyzed, including incorrect or incomplete series of IP fragments, IP address anomalies and other kinds of header fields not following Internet standards. Internet traffic was shown to contain many erroneous packets; some are the result of software and hardware errors, other the result of intentional and malicious activities.

Practical Implications: This study not only presents occurrences of header anomalies as observed in today's Internet traffic, but it also provides detailed discussions about possible causes for the inconsistencies and their security implications for networked devices.

Originality/Value: The results of this study are relevant for researchers as well as practitioners, and form valuable input for intrusion detection systems, firewalls and the design of all kinds of networked applications exposed to network attacks.

Keywords: Internet Measurement; Security; Header Anomaly; Vulnerability Classification; Malicious Traffic; Backbone Traces;

Paper Type: Research Paper

1. Introduction

In this study, Internet backbone traffic has been investigated with respect to potential security problems and many security-related anomalies in packet headers have been found. Internet traffic contains many erroneous packets; some are the result of software and hardware errors,

other the result of intentional and malicious activities. We have searched for anomalies in contemporary, highly aggregated Internet backbone traffic. The results show that header problems can be found within almost every aspect being analyzed. In this study, 27.9 billion frames have been collected and protocol headers on network and transport layers have been analyzed in order to point out all behaviour that could potentially result in a security problem for connected hosts. As a result, a systematic listing of all possible packet header anomalies is provided, including their frequencies as seen "in the wild" on an Internet backbone link. In addition, the possible security implications of each anomaly observed are discussed.

The study of backbone traffic gives a complementary view to studies of traffic with low level of aggregation, such as traffic in local networks. Backbone data provides the opportunity to gain a broader picture of different types of malicious traffic present on the modern Internet. Besides detection of various types of malicious traffic, specific attack patterns that never show up when studying traffic reaching a smaller network can be observed (e.g., distributed denial of service (DDoS) attacks). Furthermore, rare attacks are more likely to be detected within large amounts of diverse and aggregated traffic and might therefore also reveal previously unreported attack types. Some traffic may seem legitimate when studying only one host but may turn out to be malicious when studying a larger portion of the Internet.

This study sets out to update and extend older studies (Bykova et al., 2001; John and Tafvelin, 2007; Mahoney and Chan, 2001) which have reported about packets not following modern Internet standards (IP, TCP, UDP, ICMP). Additionally, detailed figures about invalid use of fragmentation are included, which is an important security issue previously not covered in the same extent, with the exception of a study about general observations of fragmented traffic (Shannon et al., 2002). While the previous studies only focus on some specific aspects, the present study presents a wider, systematic overview of the most commonly found misbehaviours on the modern Internet and how, and in what extent, common protocols are misused by attackers in their search for vulnerabilities in Internet-connected systems.

As a result, frequencies of occurrence for different kinds of malicious traffic are presented, such as invalid IP packets headers, incorrect use of IP fragmentation, IP address problems, ICMP, UDP and TCP misuse. In the search for anomalies, possible vulnerabilities are listed in table form with references to the packet headers, an approach which should make it easy to find potential problems and make it possible to evaluate the completeness of the study with respect to what header fields are being analyzed. Besides identification of header anomalies deviating from accepted Internet standards, particular well-known attacks and their common names (e.g. *Land*, *Jolt*, *sPing*, *Teardrop, Boink*) are pointed out in the tables and the analysis.

Many attacks like the ones described above are old and well-known and would therefore be expected to be very rare on the Internet and seemingly unlikely to be found in our data. However, the results of the study show that many of these attacks are still present on the Internet. One reason may be that the recent arrival of new operating systems and mobile

devices with small and newly written IP stacks (e.g., mobile phones and PDAs) may have made these attacks meaningful again. A recent example of this was the introduction of Windows Vista, where it turned out that the beta versions were vulnerable to a large number of such old well-known vulnerabilities (Newsham and Hoagland, 2006).

This study shows that it is possible to detect many commonly known attacks from an analysis of network and transport layer packet headers. We therefore believe that our approach to identify potential problems and the scale of our datasets allow us to classify, detect and report a substantial portion of malicious, incorrect and unwanted traffic present on the Internet today. The results of this study should not only be interesting for practitioners and researchers but should also be valuable input for work with intrusion detection systems, firewalls and support the design of all kinds of networked applications that must withstand network attacks.

1.1. Limitations

In our study, it has not always been possible to correlate all packets or to find all series of packets belonging to a malicious activity. There may also be other limitations. The *Smurf* and *Fraggle* attacks, for example, are attacks where ICMP and UDP packets are sent to a network's broadcast address. However, on the Internet backbone there exists no information about what addresses are used for local broadcast messages on smaller networks. Another limitation is that application-level data was removed immediately after data collection making it impossible to inspect the contents of the packets in order to validate, for example, DNS queries or to find problems in application level protocols. Finally, another group of attacks are link-level attacks such as ARP attacks which, of course, are not visible on the Internet and cannot be detected with this type of study. However, we strongly believe that these limitations do not affect the usefulness of this study in any significant way.

1.2. Outline of the paper

Section 2 describes possible anomalies in IP, TCP, UDP and ICMP headers. Different types of anomalies are divided into classes and possible security implications of each type of anomaly are highlighted. In Section 3, the real-life dataset used for the analysis is presented together with the methodology applied. Section 4 provides figures about occurrences of anomalies in the dataset according to the classification scheme presented in Section 2. Besides the figures, each anomaly is discussed and interpreted regarding its security implications. Section 5 concludes the study, highlights the main findings and gives recommendations for how to use the results.

2. Classification of anomalies

In order to provide a systematic overview of possible header misbehaviours and to ease the reading of this paper, the headers of the protocols being analyzed are shown in fig. 1-4 (IP, TCP, UDP and ICMP headers). Header fields highlighted in grey are considered to contain

potentially unusual or harmful values and are therefore included in this study. Roman numbers in the figures indicate different types of anomalies within the header fields and the tables below provide descriptions of each anomaly. This type of classification makes it possible to identify all packets violating Internet standards in any way. The motivations for investigating many of the anomalies can be found in the right column of the tables.

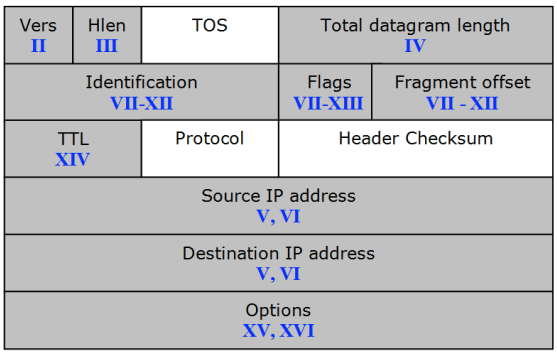


Figure 1: IP header structure

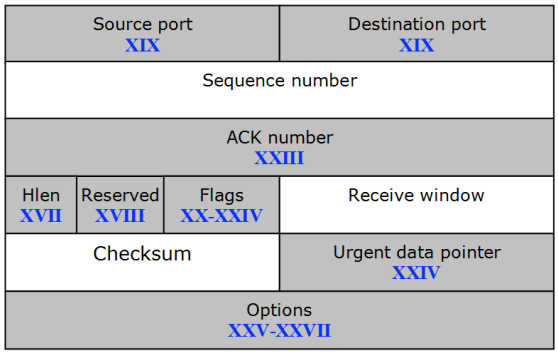


Figure 2: TCP header structure

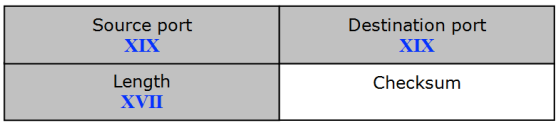


Figure 3: UDP header structure

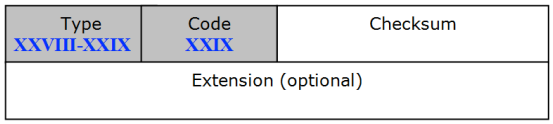


Figure 4: ICMP header structure

General IP header errors:

I	Actual IP packet length is not large enough to host a complete IP and transport header	Truncated packets might be used to confuse firewalls or remote hosts
II	Packet according to HDLC header Ethertype should be IPv4, but IP version is not 4	This is a general protocol error and these packets should have been removed by Internet routers
III	Header length field less than minimum IP header length of 20 bytes	Same type of error as in II above.
IV	Total datagram length value is not sufficient to host IP and transport header (TCP,UDP,ICMP)	Such inconsistencies might be used to confuse firewalls or remote hosts

IP address anomalies:

V	Source address equal to destination address	Can confuse a host to start sending responses to itself (*Land* and other DoS attacks)
VI	Traffic to or from private addresses (RFC 1918) and reserved addresses like loopback, class E, link local or "this network" addresses	These addresses should not be seen. If delivered to hosts, packet may create confusion or may cause unwanted or illegal traffic on local networks.

IP fragmentation anomalies:

VII	First fragment too small to contain full transport header (only for TCP, UDP and ICMP)	No reason such fragments should occur except when trying to confuse firewalls
VIII	Single packets with MF flag or Fragment offset	Either the result of lost fragments or attacker may try to use up buffer space at receiving hosts (DoS)
IX	Gaps in datagram when assembled (including missing first or last fragments)	Attempts to trigger bugs in the reassembly code or to exhaust buffer space at the receiving host (DoS), (*Boink, Opentear, Frag*)
X	Overlapping fragments	Attempts to trigger bugs during datagram reassembly or to traverse traffic filters with malicious code inside the datagram (*Teardrop, Newtear, Jolt, Nestea*)
XI	Duplicate fragments (with or without different contents)	Fragments overwriting its own contents, especially the first fragment, may be used to confuse firewalls that believe they have already inspected the TCP header
XII	Fragment makes assembled datagram exceed max. IP packet length of 64 Kbytes	Attacks where fragment offset plus datagram size exceeds 64 Kbytes IP datagram limit. A possible buffer overflow problem. (*Ping-of-death, sPing*, *IceNewk*)
XIII	Invalid IP flag combinations	May confuse receiving hosts or firewalls

Potential IP header problems:

XIV	Small TTL values (values smaller 10)	Could be result of topology mapping scan (or legitimately used by e.g., *traceroute*)
XV	IP option(s) used	IP options can be used to circumvent normal routing or to cause other problems (e.g., strict source routing). Not necessarily erroneous, but suspicious.
XVI	IP option length not matching announced IP header length	May confuse receiving hosts or firewalls

General transport header errors:

XVII	TCP header length or UDP length fields less than minimum header length of 20 bytes/8 bytes resp.	May confuse receiving hosts or firewalls
XVIII	TCP reserved bits set	Must be zero according to RFC 793
XIX	Source or destination port of zero (UDP, TCP)	Should not occur in ordinary communication if the host expects a reply. Such packets could confuse hosts when receiving them or replying.

Invalid or unusual use of TCP flags:

XX	Invalid combination of TCP flags (multiple signalling flags, zero flags)	Examples are *Xmas packets* which are results of setting random flags in hope to create confusion at endpoints or to fingerprint operating systems.
XXI	TCP SYN segment fragmented	SYN segments should never be fragmented
XXII	TCP SYN segment contains data (except T/TCP)	Data in SYN segments serve no practical use
XXIII	ACK number of zero and ACK bit set	Could be result of ACK or FIN scan attacks.
XXIV	Urgent data pointer value when URG flag set	Has been used for DoS attacks (e.g., *WinNuke*)

TCP option errors:

XXV	TCP option type invalid	May confuse receiving hosts or firewalls
XXVI	TCP option length not matching header length	May confuse receiving hosts or firewalls
XXVII	TCP option length equal to zero	Applications (e.g., Symantec Personal Firewall) might loop endlessly when parsing such options

ICMP anomalies and general statistics:

XXVIII	ICMP length anomalies	ICMP messages not following standards (too small or too large for specific type/code)
XXIX	ICMP types and codes	Source Quench may slow down senders (DoS), Redirect may place attacker as man in-the-middle

3. Data description and methodology

The dataset used in this study (John and Tafvelin, 2006) was collected from September to November 2006 on an OC192 backbone link of the Swedish University Network (SUNET). The packet header traces have been collected on a highly aggregated backbone link at 277 randomly selected times during 80 days, in order to provide a good statistical representation of all Internet traffic during the time-period at this location. At each randomly selected time, two traces of 10 minutes duration were stored. When recording the packet level traces on the 2x10GB links, payload beyond transport layer was removed and IP addresses were anonymized due to privacy concerns using the prefix preserving CryptoPAN (Xu et al., 2001). After further pre-processing of the traces as described in (John and Tafvelin, 2006) and (John and Tafvelin, 2007), the traces were moved to a central storage. An analysis program was run on the raw traces to extract malformed packet headers and invalid series of fragments. The reduced data was then stored in a database together with statistical summaries for each particular observation as listed in Section 2. For packets of special interest, corresponding flows have been extracted from the raw traces and analyzed in detail using available packet visualization software.

The complete dataset consists of 554 traces including 27.9 billion frames. 99.98% of the traffic was IPv4 carrying 19.5 TB of data in 636 million flows. During the single 10 minute intervals, depending on time of day, between 13,000 and 37,000 unique IP addresses were observed belonging to the region of western Sweden connecting to 300,000-1,000,000 unique addresses on the main Internet. A breakdown of transport protocols in the IP traffic is

summarized in Table 1. Not only numbers and fractions of packets are shown but also IP fragments and fragment series are listed (first three lines).

	IPv4	TCP	UDP	ICMP	GRE	ESP
Packets total	27,873,847,645	89.7%	9.8%	0.3%	0.1%	0.1%
Fragments total	255,470,635	0.2%	99.3%	0.0%	0.0%	0.4%
Frag. Series total	20,752,539	1.5%	95.7%	0.0%	0.1%	2.7%
Fragments w/o bulk transfer	42,755,210	1.5%	95.7%	0.0%	0.1%	2.6%
Frag. Series w/o bulk transfer	11,337,769	2.7%	92.1%	0.0%	0.2%	5.0%

Table 1: Transport protocol breakdown

The analysis of fragmented traffic requires correlation of fragmented IP packets to create fragment series. Following Shannon (Shannon et al., 2002), a fragment series has been defined as a list of fragments observed on the network derived from a single original IP packet. Consequently, fragments have been grouped into fragment series based on the IP ID, protocol and the source and destination IP fields. Furthermore a timeout value of one second was chosen to further separate fragment series. As opposed to the study by Shannon et al. (2002) which was carried out on OC12 links, the timeout had to be chosen smaller in order to compensate for the higher throughput of the links measured (OC192). In rare cases, wraparounds of the IP ID space have been observed within a few seconds, which made such a small timeout necessary. Considering the transmission rates of modern computers, a timeout of one second seems to be sufficient to capture all fragments belonging to a certain series, which was proven to be true by a number of empirical tests on the dataset. Furthermore, fragment series observed in the first or the last second of a measurement interval have not been tested for completeness, in order to avoid bias due to border effects.

Earlier studies have shown that only 0.06% of the traffic was fragmented on the measured network (John and Tafvelin, 2007). The increased fraction of IP fragments in the dataset used for this study (0.9%) is explained by a special bulk data transfer event from a space observatory to a data centre in Europe. During 7 time intervals, 213 million fragments in 9.4 million fragment series have been transferred on the outgoing link using a customized fast bulk transfer protocol based on UDP. Figures for the remaining fragmented traffic without the mentioned bulk transfer are summarized in the final two rows of Table 1. Disregarding this special event, only about 0.15% of the IPv4 traffic was fragmented.

4. Observed misbehaviours and anomalies

In the following subsections, occurrences of the anomalies classified in Section 2 as seen in our dataset, are presented in tables. The index columns use the same Roman numbers as introduced earlier. For IP level anomalies a transport protocol breakdown is provided as well.

Each table is followed by remarks and discussions about the anomalies being observed, including an interpretation and discussion of their probable causes.

4.1. IP header anomalies

I – IV: Packets with an insufficient actual packet length to carry the minimal IP and transport headers (I) have been seen very rarely, originating from different IP addresses at different times. 105 of these packets also announced insufficient sizes in the IP total length field (IV). IP version numbers not agreeing with the HDLC Ethertype (II) or IPv4 header length fields smaller than 20 (III) have not been seen at all. Since these errors rarely happen and no well-known attacks exploit such anomalies, we believe that these packets are caused by rare IP stack errors.

V: Packets with a source IP address equal to the destination address, as used in the *Land* attack, have been seen 321 times. The original land attack was based on TCP SYN packets which has been observed 9 times in the dataset. Most packets with this anomaly are UDP segments, which means that they are modified versions of *Land*. These packets have been observed at 158 different times, sent between a number of different IP addresses.

VI: IP packets to or from reserved address spaces have been observed in relatively large numbers. A couple of hundred such packets are observed at each of the 277 measurement times, with exception of two 10 minute intervals with peak numbers of around one million each. The majority (95%) of these packets use a source IP addresses belonging to the private class C address room 192.168 /16 even if private class A (10 /8) and class B (172.16 /16 – 172.31 /16) have also been observed as source addresses (5%). Traffic from loopback, link-local, class E or this-network addresses have been recorded, but in very low numbers. Most of the packets in this category are ICMP echo replies (type 0) with length of 228 bytes to four destination hosts during two measurement intervals. We believe that this was an ICMP DoS attack, where Echo replies were chosen to evade stateless firewalls. In order to disguise the real origin, spoofed private addresses were chosen. The remaining 300,000 packets, which appear in a more random and spread out fashion, could also be attacks but might as well be caused by misbehaving or misconfigured NAT gateways.

Index	# packets	TCP	UDP	ICMP	Description
I	123	104	11	8	Insufficient actual packet length
II,III	0	0	0	0	IP version and IP header length fields
IV	105	102	0	3	IP total datagram length field
V	321	9	309	3	source IP addr. = destination IP addr.
VI	2,663,891	185,863	33,780	2,444,232	Reserved address space
XIII	265,324	42,632	222,667	4	Invalid IP flags
XIV	8,067,930	896,790	1,915,931	5,199,576	Small TTL values (<10)
XV, XVI	21,991	0	18,721	2,318	IP options

Table 2: Packet counts observed in 27.8 billion IP

XIII: The only defined values for the three IP flags are don't fragment (DF), more fragments (MF) or no bits set. However, 265,000 packets with other, undefined bit values have been observed, which is an increase compared to previous studies (John and Tafvelin, 2007). All possible bit combinations have been seen, with MF+DF responsible for 99% of the invalid combinations. Most IP packets with invalid flag values carry UDP traffic, but no source or destination hosts or port numbers stand out. Furthermore, such packets are seen within each of the 277 traces, with a few traces carrying relatively large numbers of up to 10% of the packets. We believe that these packets are mainly forged packets by hacker tools like *nmap* in order to test robustness of implementations. Furthermore, a previous study (John and Tafvelin, 2007) also found indications that erroneous IP stack implementations contribute to this behaviour.

XIV: While the usage of small TTL values is no unusual behaviour per se, it might still indicate topology map scanning as preparations for specific attacks towards a network. Modern operating systems use default TTL values of 60 or more. Network paths with hop-counts of more than 50 are very rare, which means that IP packets with small TTL values can be explained by:

- old Windows systems with TTL values of 32 (hop-counts between 20 and 30 are plausible)
- packets from *traceroute* applications (commonly only ICMP and UDP)
- topology mapping scans using TCP or UDP on common ports in order to avoid ICMP filters

Indeed, 99.6% of the ICMP packets with a small TTL are of type 8 (echo), which is used for *traceroute* in Windows systems and some Unix versions. The remaining ICMP packets with small TTL values are of type 3 (destination unreachable). Their packet length of 28 bytes indicates that they are replies to either UDP or ICMP packets. Hosts receiving these ICMP type 3 messages typically show heavy activity on UDP ports known to be commonly used for P2P signalling traffic, which leads to the conclusion that the messages are artefacts of the unreliable nature of P2P overlay networks and thus are most likely not a security issue.

Many Unix systems use UDP packets with varying size of around 40 bytes for *traceroute* with a destination port within the otherwise uncommon port range 33434 to (around) 33534. 92% of the UDP packets with short TTL fall into this port range and since they are small in packet size, they are most likely legitimate *traceroute* packets. The remaining 8% UDP packets are not only directed to other random port numbers, they also have larger packet sizes between 100 and 1500 bytes, which indicates that they must be treated as suspicious.

Most of the TCP packets with small TTL values are the downstream part of regular TCP connections. They also show TTL values of 8 or 9 which is larger than the *traceroute* traffic observed for ICMP and UDP (TTL values of 3, 5 or 6 for this specific topology), which consequently is too large for topology mapping scans. These remote hosts are therefore most likely running a system with a small default TTL value, such as Windows NT or 95. This

leaves about 5300 packets to be suspected as topology mapping scans via TCP (suitable TTL values with small packets).

Additional 55,500 packets of protocol type 103 (PIM–protocol independent multicast) have been observed with TTL values of exactly one – which are valid PIM bootstrapping messages following RFC 2362.

XV, XVI: As already observed in a previous study (John and Tafvelin, 2007), IP options are rarely used. Source routing is the main security concern regarding IP options, but has not been observed at all (neither IP option value 131 nor 137). A large part of the packets with IP options is sent by 10 sources to one destination inside Gothenburg via UDP. Strangely, instead of using real options, a sequence of four EOOL bytes is sent (0,0,0,0) which is most likely due to inappropriate configuration or buggy software. The options used by ICMP traffic are of valid option type 7 (record route), and the remaining 948 IP options observed are option type 148 (router alert) being sent within RSVP packets. IP option header length inconsistencies (XVI) have not been observed.

4.2.IP fragmentation anomalies

The figures of IP fragmentation anomalies in the dataset used are skewed due to one exceptional event, where exactly one host inside a University was sending UDP segments fragmented into 6-7 fragments to five different hosts at five different measurement times during a few days, in very high frequency. As destination port number, the entire 16 bit port space was used in iterative fashion. About 50% of the IP series included different types of inconsistencies, most with missing last fragments, but also other gaps and a number of "single packet series". Most likely this is a hijacked host used for directed DoS attacks (such as a *Frag attack*). The 1.6 million fragment series from this host have been summarized in the first row of Table 3 and are excluded from the remaining analysis.

Index	#series	TCP	UDP	ICMP	Description
VII-IX	1,651,324	0	1,651,324	0	Exceptional fragmentation event
VII, XXI	71	71	0	0	Series with short first fragment
VIII	80,981	18,117	61,001	1,723	Single packet "series"
IX	29,939	685	29,217	37	Incomplete series ("gaps")
X	37	5	32	0	Series with overlapping fragments
XI	1,864	1,285	579	0	Series with duplicated fragments
XII	0	0	0	0	Series exceeding 64K IP length

Table 3: Fragment series counts observed in 20.8 million

VII, XXI: Fragment series where the first fragments are too small to contain all headers (VII) are observed in 71 series sent by a single host. The fragments had furthermore the TCP SYN flag set (XX) and the IP total length field was smaller than 40 bytes (IV). This is probably a DoS attack trying to confuse firewalls or receivers.

VIII: About 81,000 packets appear to be part of a fragment series (either MF bit or fragment offset set), but no other fragments are observed for the same series. This could potentially be used to confuse hosts or firewalls. Another plausible explanation is that further fragments have been dropped or routed asymmetrically. Around 10% of the single-packet series used IP IDs of zero, which means that IDs of zero are around 1,500 times more common than any other possible number in the 16-bit space. This obvious over-representation of one IP ID is suspicious and indicates either malicious intentions or usage of protocol implementations not following Internet standards. 58% of the single fragment series have the MF bit set and are full sized packets (~1500 bytes), looking like typical first packets in fragment series. The remaining 42% have characteristic properties of last packets in fragment series, with IP offset values set and small data portions. These observations suggest that dropped packets could be an explanation for many of these packets.

IX: Incomplete fragment series are very undesirable because they consume resources at the receiving host which needs to store the arriving fragments until the series is complete and the entire packet can be handed over to the next protocol layer. Known attack types in this category are *Opentear* and *Frag*. Around 50% of incomplete fragment series are missing the last fragment, a missing first fragment accounts for 25% and the remaining 25% are gaps in between. In the dataset, 42 hosts receive about 80% of all the incomplete series. The incomplete series are sent by different hosts and are targeted to random UDP ports. The sizes of the gaps range from 8(!) bytes to full packet size. Besides these incomplete series, valid series of fragments are also sent to the hosts in question and only around 1/5 of all fragment series are actually incomplete. Even though this behaviour could be explained by a high number of packet losses along the path, a suspiciously high density of IP IDs of zero together with the unusual gap size of 8 bytes makes it more likely that these hosts are the target of a DDoS attack by a number of bots, similar to the hijacked host causing the exceptional event described above.

X: Overlapping fragments are also known to be a common DoS attack type (such as *Teardrop, Jolt* and *Nestea*). Overlaps are very rare in the dataset and only 37 occurrences have been observed at 35 times between different hosts, mainly UDP. Almost all series with overlaps (35) also include missing sections in the complete IP datagram. The small overlapping fragments (8 to 48 bytes) have exactly the same size as the gaps in the specific series, but they fill the gaps at the wrong offset. Depending on the length of the series, such overlapping fragments appear up to 3 times per series, with the last byte of the overlapping fragment always at the datagram offset of 912, 1832 and 5352 bytes. We believe that this consistent behaviour is either the result of a soft- or hardware error or an attack tool repeating the same behaviour.

XI: In contrast to overlaps, duplicate fragments mean that two fragments cover the exact same portion (offset and fragment length) within the fragmented datagram. Potentially, this could cause similar problems as overlaps in *Teardrop* attacks, namely overwriting previous

benign portions with new malicious data. Since packet payload in our dataset has been removed, we need to rely on transport header checksum information (note that transport header checksums in fragment series are only available in the first fragment, so changed payload in following fragments cannot be detected in the present study). According to a checksum comparison of duplicate first fragments, many duplicate fragments observed are in fact sheer retransmissions. There are also sequences of duplicated single fragment series (VIII) which are therefore categorized as duplicates as well. Note that an unproportional large number of these single series duplicates use IP IDs of zero, which again appears to be a good criteria for identification of malicious fragmented traffic. Duplicated fragments with different payload (and consequently different transport header checksums) within otherwise complete and valid series have only been observed 104 times by 21 hosts.

XII: Attacks, with fragment series exceeding the maximum IP packet length of 64 Kbytes (*Ping-of-death, sPing, IceNewk*), are not observed at all. This attack type was popular in the late 90's, but since then most applications and operating systems have been patched. Even though it is good news that this attack is not observed anymore, application developers and firewall administrators should keep this attack in mind.

4.3.TCP header anomalies

In the first row of Table 4 TCP segments with multiple invalid header fields have been summarized. Garbled TCP headers have been defined as combinations of two or more independent[1] field anomalies within one TCP header. Garbled headers have been observed during all measurement intervals with no specific host standing out. Such packets can easily be forged by network exploration tools using raw sockets, such as *nmap*. Note that the segment counts in the following categories do not include the 9,757 garbled TCP headers.

Index	# segments	Description
XVII - XXVII	9,757	Garbled TCP header
XVII	72	TCP length short
XVIII	114,876	Reserved bits set
XIX	6,180	TCP port zero
XX a	178,993	Invalid signaling flags
XX b	81,982	pure FIN (no ACK)
XXII	29,369	SYN with data
XXIII	389,060	ACK number of zero
XXIV	440	Urgent pointer set
XXV - XXVII	9,038	TCP option errors

Table 4: TCP segment counts observed in 25 billion TCP

[1] i.e., anomalies in different fields. XVIII / XX and XXV-XXVII are considered dependent

XVII: TCP headers with length values smaller than the minimum TCP header length of 20 bytes have been observed 72 times. The announced header length values are mainly zero and eight bytes.

XVIII: The reserved bits in the TCP header are the four bits following the TCP header length field. The previously six reserved bits have been reduced to four since the introduction of ECN (RFC 3168). In this study, the ECN bits have not been considered. According to the TCP specification (RFC 793) the reserved bits must be zero. However, almost 115,000 packets with non-zero reserved bits have been observed in the dataset. Interestingly, TCP reserved bits are only set together with TCP flag combinations of either RST/ACK or SYN/ACK. 21% of the invalid reserved bits have all 4 bits set and appear consistently with RST/ACK packets. These RST/ACK packets are mainly replies from HTTP servers on port 80. Valid TCP conversations are closed by the servers with sequences of 3-4 RST/ACK packets, where each but the initial RST/ACK has all reserved bits set. This appears to be an incorrect TCP implementation rather than a security issue.

The remaining 79% of TCP headers with invalid reserved bits appear within SYN/ACK packets only, but this time with different bit combinations. These SYN/ACKs are sent in high frequency by different sources, usually from port 80 or 7000. Interestingly, no SYN packets triggering these SYN/ACK responses and no further packets originating from these sources have been seen. This behaviour can either be explained by an asymmetrically routed (and therefore un-captured) SYN attack, but more likely are SYN/ACK attacks. In this case the often used source port of 80 can be explained by attempts to pass certain stateless firewalls.

XIX: TCP port number zero is reserved and should not be used for data transfer. Nevertheless, 6,180 TCP segments with a port number of zero have been observed equally shared between source and destination port numbers. These segments have been sent by approximately 700 different sources within almost all measurement intervals. Most of these packets are SYN packets being part of host scanning campaigns. Very few of them are actually replied to with RST/ACK packets.

XX: Anomalies within the 6-bit TCP flags field have been divided into invalid combinations of the signalling flags SYN, FIN and RST (XX-a) and another, less critical, but still unexpected flag value (XX-b).

XX-a: Combinations of invalid signalling flags appear in frequencies comparable to observations in an earlier study (John and Tafvelin, 2007). 826 segments had no signalling flags set at all (zero flags). The combination RST+FIN in the same header has been observed 939 times, SYN+FIN 435 times and 377 segments had SYN+FIN+RST set. The most common flag anomaly however is RST+FIN with more than 176,000 occurrences. Packets with invalid signalling flag combinations have been seen evenly distributed within all traces, sent by more than 45,000 hosts to different destinations where approximately 50% are directed to port 80.

The most likely explanation for such segments is crafted packets (such as *X-mas*) by network exploration and testing tools like *nmap*.

XX-b: Beside combinations of signalling flags, another type of unexpected flag values has been observed. According to the TCP specification, every segment in an established TCP connection (except the initial SYN) is required to carry an ACK, so there is no reason for pure FIN packets to exist. Mahoney (Mahoney and Chan, 2001) showed that identification of FIN packets without an ACK can reveal port-sweeps and OS fingerprinting campaigns. In our dataset, 82,000 segments with only FIN flags set have been observed, sent by 10,000 different hosts to 27,000 destinations. Interestingly, more than 50% of these packets are sent to different well known P2P ports. Most pure FIN packets are sent after a sequence of SYN connection attempts just before the socket is finally closed on the sending host after the TCP timeout. Even if this behaviour is not defined in the standards, we do not consider it to be security relevant.

XXII: According to RFC 793, it is not prohibited to append payload data to SYN packets. However, this behaviour is de facto non standard and therefore somewhat suspicious. The only well defined usage of SYN packets with data is T/TCP (RFC 1644) which can be identified by TCP options. However, this has not been observed in this dataset. Around 29,000 SYN packets with data had a data portion of exactly 24 bytes and have been sent to TCP port 53 (DNS). These packets are seen quite evenly distributed among all measurement times and are exchanged between about 113 different IP addresses outside the region of Gothenburg to 66 hosts inside. The connections are initialized by this SYN data segment, replied by a SYN/ACK and then immediately closed by the initiator with a RST. According to SANS Intrusion Detection FAQ (SANS, 2008) this behaviour has been observed in other networks and is probably caused by a common but buggy DNS system. Other segments with SYN flags and data have only been observed in packets with garbled TCP headers.

XXIII: TCP sequence and acknowledgement numbers use 32-bit integers. Even if the selection of initial sequence numbers (ISN) is known not to be completely random for many systems (Zalewski, 2002), 390,000 out of 25 billion segments having an ACK number of zero is a clear overrepresentation. It turned out that 96% of these packets are RST/ACK segments. A large portion of these segments is sent by hosts closing valid connections with series of RST/ACK packets. In these connections, all but the initial RST/ACK packets carried an ACK number of zero, which appears to be an implementation problem rather than malicious activity. In addition, some RST/ACK storms have been observed with no SYN packets that could have triggered these replies. This behaviour could be explained by an asymmetrically routed (and therefore un-captured) SYN attack, but more likely as RST/ACK attacks where the ACK number field was left empty (zeros). The remaining ACK numbers of zero (4%) are pure ACK packets sent between a large number of hosts, thus there is no indication of obvious malicious intentions.

XXIV: Urgent pointers are basically a valid way to transmit "out of band" data within the regular TCP stream of a connection, used e.g., for quick delivery of control strings in applications like *telnet* or *ftp*. However, in the past urgent pointers have turned out to be an effective way for DoS attacks due to buggy operating systems (*WinNuke*) - regardless of the actual value of the urgent pointer. This means that many firewalls today drop all packets with urgent pointer flags. In the dataset, 3,389 TCP segments carried an URG flag, though none of these packets were directed to port 139, the target of the infamous original *WinNuke* attack. Most of the URG segments had generally garbled TCP headers and only 440 "pure" URG flags have been observed. 71 of these packets have been sent to port 21 (*ftp*) with plausible urgent pointer values of 2 – 4 (e.g., for control characters like "ctrl-c"). The remaining pure URG segments have been sent to different P2P port numbers with urgent pointer values of one or zero. Especially urgent pointers pointing to a data offset of zero are suspicious since it indicates that there is in fact no data to deliver urgently.

XXV - XXVII: TCP option anomalies in this dataset have been observed in frequencies comparable to results of a previous study (John and Tafvelin, 2007). Three different anomalies (XXV-XXVII) have been observed in different combinations within 9,000 TCP segments. Such packets are most likely crafted by tools like *nmap*. The most common inconsistencies are either usage of undefined option types (TCP options are only defined for type numbers up to 26) and length announcements in the length field of specific options which do not agree with the header length of the general TCP header. Additionally, 967 options carried an option length value of zero which has been shown to cause endless loops when processing them in receivers and traffic filters (e.g., Symantec Personal Firewall).

4.4.UDP header anomalies

Index	# packets	Description
XVII	67	UDP length field
XIX	17,242	UDP port zero

Table 5: UDP segments observed in 2.7 billion UDP

XVII: 67 packets with too small values in the UDP length field have been observed in 56 different measurement intervals sent by 59 different hosts. The most common invalid header length value is zero. Only two packets announced header length of one and two, which is too small to carry the minimum UDP header length of 8 bytes. The low frequency of this anomaly does not allow us to reliably classify this as a malicious action.

XIX: As for TCP, UDP port numbers of zero are also reserved. In the dataset, around 3,000 UDP segments have been sent to UDP port zero, which is definitely not permitted and can lead to crashes of hosts or firewalls. According to the UDP specification (RFC 768) the source port number field is optional and may be set to zero if not used, i.e., no reply is

expected. A large portion of about 14,000 packets has been seen with source port values of zero. Even if this behaviour per se is permissible, it turned out that all the segments coming from UDP port zero are sent in short scanning campaigns, scanning over ranges of /24 networks (254 IP addresses) on port numbers 1025 and 1026 (win-rpc). Such campaigns have been launched by 30 different hosts at 30 different times. The payload length of all these packets was consistently 319 bytes. UDP Source port numbers of zero therefore seem to be good indication of windows messenger spam, where spammers sweep over IP ranges and try to deliver pop-up messages to windows systems with windows messenger active.

4.5.ICMP anomalies and observations

XXVIII, XXIX: A breakdown of observed ICMP packet types is presented in Table 6. ICMP messages with undefined type or code are summarized in the second last row of the table. Furthermore, messages with impossible length values according to their ICMP types are summarized in the last row. This means that the counts and fractions presented per ICMP type are counts of packets with valid types and codes and plausible length values. In the following paragraphs, this table will be analyzed regarding possible ICMP attacks.

Ping-of-death type attacks, where a fragmented ICMP packets exceed 64 Kbytes when assembled, have not been observed (in fact no such fragments attacks were detected, see XII). There where also no ICMP *p-Smash* attacks (floods of **ICMP router advertisements** (ICMP type 9)).

Spoofed **ICMP destination unreachable** messages (type 3), as used in a *Smack* attacks, could be present in the dataset, but are difficult to pinpoint in this study due to anonymized IP addresses and missing payload of the ICMP messages. However, neither source nor destination hosts appeared to be involved in unusually dense sequences of ICMP type 3 messages during the 277 measurement times.

Also **ICMP source quench messages** (type 4) have been reported to be exploited in order to slow down networks. In the dataset, almost 38,000 such messages have been seen and even if such DoS attacks cannot be ruled out, no obvious attack patterns were identified.

The large number of **ICMP redirects** is caused by two hosts sending about 46 million ICMP packets with type 5, code 1 (host redirect) to 300,000 destinations during the measurement intervals within a period of 12 days. The general behaviour of these hosts clearly shows that they are not routers or gateways but rather normal workstations establishing only connections to HTTP and P2P hosts. Most likely, these packets are part of a DoS attack like *Winfreez*, which can cause windows machines to change their routing tables. Unfortunately, missing packet payload makes it impossible to analyze the announced gateway addresses in the redirect messages observed.

ICMP type	# packets	Percent	Description
0	5,927,990	6.10%	Echo Reply
3	11,964,456	12.31%	Destination unreachable
4	37,899	0.04%	Source Quench
5	46,437,420	47.77%	Redirect
6	1	0.00%	Alternate Host Address
8	16,287,609	16.76%	Echo
11	10,160,608	10.45%	Time Exceeded
12	60	0.00%	Parameter Problem
13	63	0.00%	Timestamp
14	60	0.00%	Timestamp Reply
15	2	0.00%	Information Request
17	10	0.00%	Address Mask Request
Undefined	33,517	0.03%	Undefined ICMP type or code
Invalid length	9,467,433	9.74%	Valid type and code, but invalid length

Table 6: Breakdown of 97.2 million ICMP packets

ICMP timestamp attacks, like *Moyari13*, cannot be identified since the timestamp information has not been preserved in the dataset. However, all timestamp messages (type 13 and 14) have valid packet lengths. Furthermore, all except three timestamp messages (type 13) have immediately been replied to (type 14), which does not indicate malicious behaviour.

Undefined ICMP types and codes could potentially be part of *Twinge* or *Trash* attacks, which cycle through all types and codes, thereby trying to create confusion or crash certain operating systems. The 33,517 packets with random types and codes in the dataset have been sent between a couple of thousand hosts quite evenly distributed among all trace intervals, with no host or time interval standing out, which means that at least large scale campaigns of these attacks have not been observed.

Finally, a quite large number of packets with valid types but **invalid packet lengths** have been observed. According to RFC 792 (ICMP), most messages except echo have well defined packet sizes or are at least bound to a maximum (often 56 bytes including 20 bytes IP header, 8 bytes ICMP header, 20 bytes original IP header and up to 8 bytes of original payload). Almost all ICMP packets with invalid lengths are of type 3 (destination unreachable) and the remaining 2% are of type 11 (time exceeded), having packet sizes exceeding 56 bytes. Since a large number of hosts were sending these packets in small frequencies, it is likely that most of these are the result of implementations with wrong interpretation of the ICMP standard. Only three ICMP messages have been observed with insufficient IP packet lengths to host an ICMP header.

5. Summary and Conclusions

In this paper, we first provide a systematic classification of header fields not following Internet specifications with a potential to cause security problems. This systematic

classification serves as a starting point for identification of such header inconsistencies within our large dataset consisting of packet header data collected on a contemporary, highly aggregated Internet backbone link with diverse traffic composition. Occurrences of each header anomaly as observed "in the wild" are then presented followed by detailed discussions about possible causes and an interpretation of the observations with respect to the relevance for security.

As a general observation, it is surprising to see that that many old, well known attacks can still be found. On the upside, some former popular attacks, such as *Ping-of-death* and the *IP source route exploit* have not been observed at all. Generally, a constant "noise" of malformed or inconsistent packet headers was observed, similar and consistent with the observations of constant scanning activities in another recent connection-level study (John et al., 2008). This type of background noise is in some cases likely to be caused by rare hardware and software errors, but most must be attributed to the possibilities even inexperienced hackers have today to generate more or less random packet headers with existing networking tools.

Also a number of exceptional events of malicious activity have been observed. An ICMP DoS attack with otherwise unsuspicious echo reply messages has been identified due to IP address analysis regarding reserved IP spaces. A sequence of fragmented datagrams has been sent in high intensity from a single host during short time intervals. The detailed analysis of the fragment series revealed a directed *Frag* attack, using incomplete fragment series with the intention to exhaust resources at the receivers. Furthermore, an analysis of IP ID values of zero appeared to be a successful approach to detect different fragmentation anomalies, and observations in the reserved bits field of the TCP header revealed a series of SYN/ACK attacks. Port number values of zero proved to be effective in detecting port scanning campaigns, both for TCP and for UDP. Finally, a DoS attack applying ICMP redirect messages has been observed.

There are many interesting future research possibilities to improve the results and insights of this study, such as a complementary flow-level investigation of scanning traffic or a similar packet inspection including at least some application-level data, to name but a few. However, the results of this study show that it is possible to detect a substantial part of malicious activities just from inspection of header data. The observations also show that inspection of IP addresses spaces, IP ID values, port number values, the entire flags section in the TCP header (reserved bits and signalling flags) and ICMP messages are the most effective mechanisms to find malicious traffic and therefore form a basic set of rules which should be included into all modern firewall and IDS systems.

The results presented here are based on a rare dataset of aggregated backbone traffic and are intended to guide and support network administrators and application developers in their constant task of tuning their systems in order to mitigate the wide range of incoming

malicious attack traffic. Furthermore, we believe that this study helps researchers and practitioners to gain a better understanding of the characteristics of today's Internet traffic in order to remain proactive.

Acknowledgements

This work was supported by SUNET, the Swedish University Computer Network. The authors furthermore want to thank Sven Tafvelin for valuable discussions and comments.

References

Bykova M. and Ostermann S. and Tjaden B. (2001), "Detecting Network Intrusions via a statistical Analysis of Network Packet Characteristics", Proceedings of the 33rd Southeastern Symposium on System Theory, pp. 309-314

John, W. and Tafvelin, S. (2006), "SUNET OC 192 Traces, fall 2006", DatCat, available at: http://imdc.datcat.org/collection/1-04HQ-3=SUNET+OC+192+Traces\%2C+fall+2006 (accessed 22 July 2008)

John, W. and Tafvelin, S. (2007), "Analysis of Internet Backbone Traffic and Header Anomalies observed", Proceedings of ACM SIGCOMM Conference on Internet Measurement, pp. 111-116

John W. and Tafvelin S. and Olovsson T. (2008), "Trends and Differences in Connection Behavior within Classes of Internet Backbone Traffic", in Claypool M. and Uhlig S. (Eds.), Proceedings of the 9th Passive and Active Measurement Conference, Springer, Berlin, pp. 192-201

Mahoney M. and Chan P. (2001), "PHAD: Packet Header Anomaly Detection for identifying hostile Network Traffic", Florida Tech, Technical Report CS-2001-4

Newsham T. and Hoagland J. (2006), "Windows Vista Network Attack Surface Analysis: A Broad Overview", Symantec Corporation (accessed 22 July 2008), available at: http://www.symantec.com/avcenter/reference/Vista_Network_Attack_Surface_RTM.pdf

SANS Institute (2008), "SANS Intrusion Detection FAQ", available at: www.sans.org/resources/idfaq/dns.php (accessed 22 July 2008)

Shannon C. and Moore D. and claffy k.c. (2002), "Beyond Folklore: Observations on fragmented Traffic", IEEE/ACM Transactions on Networking, Vol. 10, No. 6,, pp. 709-720

Xu J. and Fan J. and Ammar M. and Moon S.B. (2001), "On the Design and Performance of Prefix-preserving IP Traffic Trace Anonymization", Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement, ACM, New York, NY, pp. 263-266

Zalewski M. (2002), "Strange Attractors and TCP/IP Sequence Number Analysis - One Year Later", available at: http://lcamtuf.coredump.cx/newtcp/ (accessed 22 July 2008)

PAPER IV

Wolfgang John, Maurizio Dusi and kc claffy

Estimating Routing Symmetry on Single Links by Passive Flow Measurements

Submitted to Conference

Chalmers University, 2009

Estimating Routing Symmetry on Single Links by Passive Flow Measurements

Wolfgang John[1], Maurizio Dusi[2] and kc claffy[3]

[1]Chalmers University, Sweden. email: wolfgang.john@chalmers.se

[2]Università degli Studi di Brescia, Italy. email: maurizio.dusi@ing.unibs.it

[3]CAIDA, UCSD. email: kc@caida.org

Abstract

The assumption of routing symmetry is often embedded into traffic analysis and classification tools. This paper uses passively captured network data to estimate the amount of traffic actually routed symmetrically on a specific link. We propose a Flow-Based Symmetry Estimator (FSE) – a set of metrics to assess symmetry in terms of flows, packets and bytes, which disregards inherently asymmetrical traffic such as UDP, ICMP and TCP background radiation. This normalized metric allows fair comparison of symmetry across different links. We evaluate our method on a large heterogeneous dataset, and confirm anecdotal reports that routing symmetry typically does not hold for non-edge Internet links, and decreases as one moves toward core backbone links, due to routing policy complexity. Our proposed metric for traffic asymmetry induced by routing policies will help the community improve traffic characterization techniques and formats, but also support quantitative formalization of routing policy effects on links in the wild.

1 Introduction

In today's Internet, path stability is not guaranteed, i.e. many nodes along a path offer alternative routes to the same destination. If packet streams between two endpoints follow the same physical links[1] between intermediate nodes for both forward and reverse direction, they are *symmetrically routed*. Otherwise, routing between this pair is asymmetric. A common cause of routing asymmetry is "hot-potato routing", the business practice of configuring traffic crossing one's own network to exit as soon as possible. Another cause is link redundancy within networks or multipath routing. Since routing decisions occur independently for each flow[2], load-balancing may cause different flows destined for the same endpoint to follow different physical links, even if all the intermediate nodes are the same.

[1]Optical links, generally composed of a pair of unidirectional fibers or wavelengths, are here considered as one physical link.

[2]To our best knowledge, most routing is done on a flow- or IP-Pair level in order to minimize jitter and out-of-order packets within sessions.

Literature on routing asymmetry has mainly considered an end-to-end perspective, inferred by active measurements of delay or path differences between endpoints [1, 2, 3, 4]. To our knowledge, using passive measurement to quantify routing asymmetry observed on a specific link has only received tangential reference [5]. We propose a technique that uses passive measurements to quantify the amount of traffic routed (a)symmetrically on specific network links, in terms of flows, packets and bytes. Using passively captured network data, the Flow-Based Symmetry Estimator (FSE) method provides an effective way to exclude traffic that is canonically asymmetric, such as ICMP traffic or *nonproductive TCP background radiation* [6], allowing a fair comparison of routing symmetry across different links with substantially different traffic decomposition.

Knowledge of the fraction of symmetric flows on specific links is especially important to traffic analysis and characterization tasks, which are often performed on data collected on single measurement points. Researchers and developers often embed an assumption of traffic symmetry in tools and analyses [7, 8, 9], an assumption only safe for stub access links, otherwise quite harmful [10].

We wanted to provide the community with a technique and accompanying open source tool for measuring flow symmetry, as well as raise awareness about macroscopic symmetry characteristics by providing statistics from running such tools over a variety of data. We evaluated our technique on traffic traces from four varied locations (Tier-2 to Tier-1 backbone) in two countries (USA and Sweden) over a period of four years (from 2006 till 2009), to provide a baseline global data set on routing symmetry. Such data sets will allow tracking of macroscopic Internet trends. Our main contributions are: (i) a simple method to assess and fairly compare routing symmetry on specific links; (ii) an open source tool for analyzing flow symmetry based on our method; and (iii) symmetry statistics for a large heterogeneous set of network traces.

Section 2 explains our choice and implementation of FSE to analyze flow symmetry. Section 3 and 4 describe the data and the results of applying FSE to the data, resp. Section 5 validates the method and Section 6 concludes the paper.

2 Flow-based Symmetry Estimator

In this section we present the *Flow-based Symmetry Estimator (FSE)*, a simple method (depicted in Figure 1) and associated tool[3] to estimate the level of routing symmetry from passively measured flow data that takes unidirectional 5-tuple flow data as input. We could have computed symmetry based on IP pairs (2-tuples), but most traffic classification and engineering methods deal with flows [7, 8, 9], so we chose the flow granularity. We used CoralFlow (part of CoralReef [11]) to extract interval-based 5-tuples of source and destination IP, port numbers and protocol. Due to its simplicity, most traffic analysis tools [12] prefer this method to tracking TCP connection state, although we use TCP connection information extracted from packet level-data [5] to validate our technique in Section 5.

[3] Available at http://www.cse.chalmers.se/∼johnwolf/FSE/

```
1:  given a time-interval of traffic trace:
2:        consider TCP data traffic (TCP packets carrying data)
3:        T_f (T_b) = set of tuples going forward (backward)
4:        T_f ∩ T_b = set of symmetric tuples T_S
5:        pkts (bytes) in T_S=set of symmetric pkts (bytes)
```

Figure 1: The FSE method. After collecting a unique list of unidirectional flows for each direction of a link, FSE classifies 5-tuples as symmetric if they appear on both lists. Packet (byte)-level symmetry is the fraction of packets (bytes) sent between tuples classified as symmetric, so that the degree of symmetry can be quantified in three dimensions: 5-tuple flows, packets, bytes.

2.1 Removing inherently asymmetric traffic from data

Our first step is to remove from the traces any traffic that is inherently asymmetric, such as UDP and ICMP flows that do not always expect packet recipients to reply[4], and which would mislead symmetry comparisons if they appear in different magnitudes across networks. TCP background radiation, such as network scanning and probing, can also be a substantial fraction of total inherently asymmetric flows on some links, although it is usually a much lower proportion of bits [6, 14]. FSE discards ICMP, UDP, and TCP signaling packets with no data. As a heuristic for the TCP category, we keep only TCP packets without signaling flags (SYN/FIN/RST) but with the ACK bit set, thereby removing unreplied single-packet probes, scans, or attacks using SYN, FIN, or RST flags. We call the post-filtered data *TCP data traffic*, reflecting the dominant transport activity on the Internet [15, 16], at least so far.

2.2 Observation time interval

We use CoralFlow to create flow 5-tuples for a given observation interval. CoralFlow defines flows by timeout interval, i.e., two packets sharing the same tuple belong to the same flow if their timestamps are within a given time interval. CoralFlow splits traces into chunks according to the specified time interval and collates unique lists of 5-tuples for each direction. The results might be affected by border effects, i.e. long flows spanning many intervals, or short symmetric flows that seem asymmetric because packet exchange occurs at the edge of an interval. We will evaluate these effects by varying the time interval, described in Section 4.2.

3 Datasets

Table 1 lists the packet-level datasets we considered. The data from GigaSUNET was collected on a backbone close to the edge of the Internet, on an OC192 link which was the primary link from the region of Gothenburg to the main Internet outside Sweden. The link mainly carried traffic from major universities and large student residential networks, but also from a regional

[4]While many application protocols communicate in bidirectional request/respond fashion over UDP (e.g. DNS), related work has shown that UDP flows on some links are dominated by single-packet flows with no observed response, such as P2P signaling and unsolicited traffic (scanning, DoS) [13].

		Time interval	#flows	pkt/s	bytes/s	Network loc.
GigaSUNET	2006-04	6x10min	8.9M	142Kp/s	790Mbit/s	Tier2 backbone
	2006-11	6x10min	15.6M	176Kp/s	1008Mbit/s	(Sweden)
OptoSUNET	2009-01	6x10min	57M	358Kp/s	1700Mbit/s	Tier2-Tier1 connection
	2009-02	6x10min	62M	442Kp/s	2000Mbit/s	(Sweden)
Eq-Chicago	2008-04	1x1hour	119M	717Kp/s	3970Mbit/s	Tier1 backbone
	2008-05	1x1hour	134M	936Kp/s	6100Mbit/s	(Illinois-Washington)
Eq-SanJose	2008-07	1x1hour	145M	680Kp/s	3000Mbit/s	Tier1 backbone
	2008-08	1x1hour	139M	664Kp/s	3040Mbit/s	(California)

Table 1: Dataset description. Two datasets are from OC192 links in Swedish networks: GigaSUNET, operative until 2007, and OptoSUNET's current connection to NorduNet. The latter two are from OC192 backbone links of a Tier1 ISP in the U.S.

access point exchanging SUNET traffic with local ISPs. TCP was responsible for 42% of flows, which corresponded to 93% of packets and 97% of bytes. UDP carried 55% of flows (6% of packets and 3% of bytes). Other transport protocols, such ICMP, GRE and ESP, represented minor traffic amounts.

In the current OptoSUNET, customers are redundantly connected to a central Internet access point. Besides some local exchange traffic, the traffic routed to the international commodity Internet is carried on two links (40Gb/s and 10Gb/s) between SUNET and NorduNet. The data used in this study was collected on the 10Gb/s link, which according to SNMP statistics carried 50% of all inbound but only 15% of the outbound traffic volume. Around 20% of flows on the link during the measurement interval have been exchanged via TCP, corresponding to 82% of packets and 89% of bytes, while 79% of connections (16% of packets, 9% of bytes) have been UDP flows.

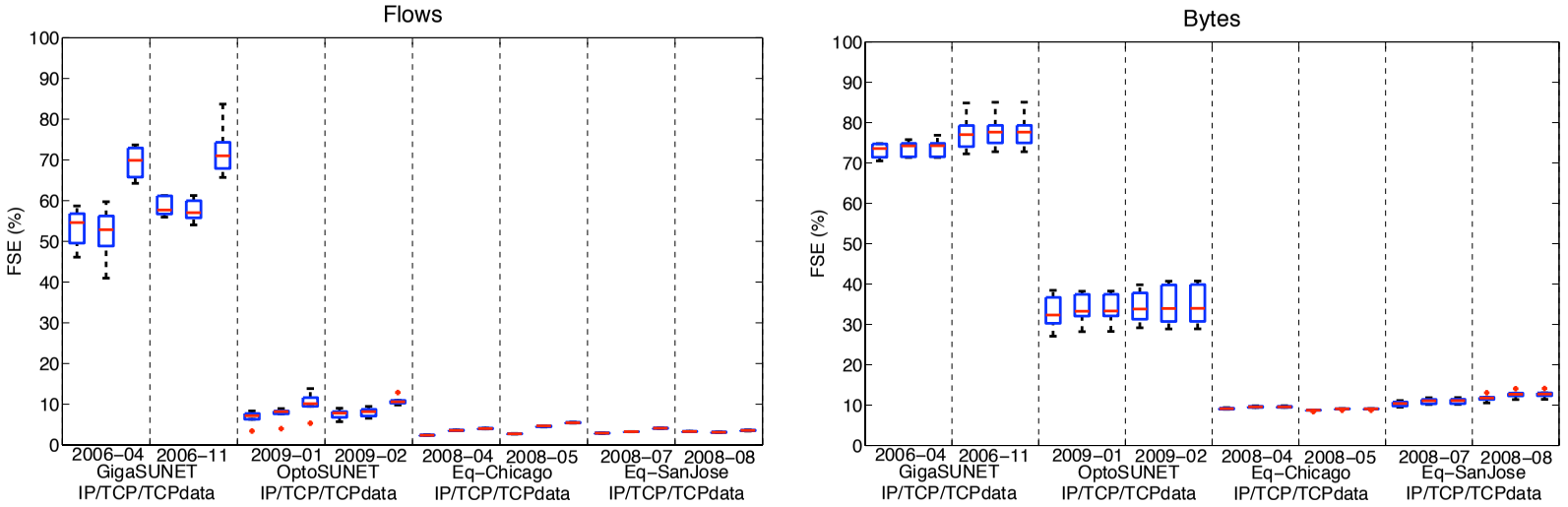
The two core links are part of an OC192 Tier1 backbone operated by a commercial ISP in the U.S. The first link connects Chicago and Seattle, monitored at an Equinix datacenter in Chicago. The other one connects San Jose and Los Angeles, monitored at a datacenter in San Jose. On those links, TCP is responsible for about 50% of flows, which was 85% of packets and 93% of bytes on average. UDP carried about 45% of flows (13% of packets and 6% of bytes).

4 Experimental results

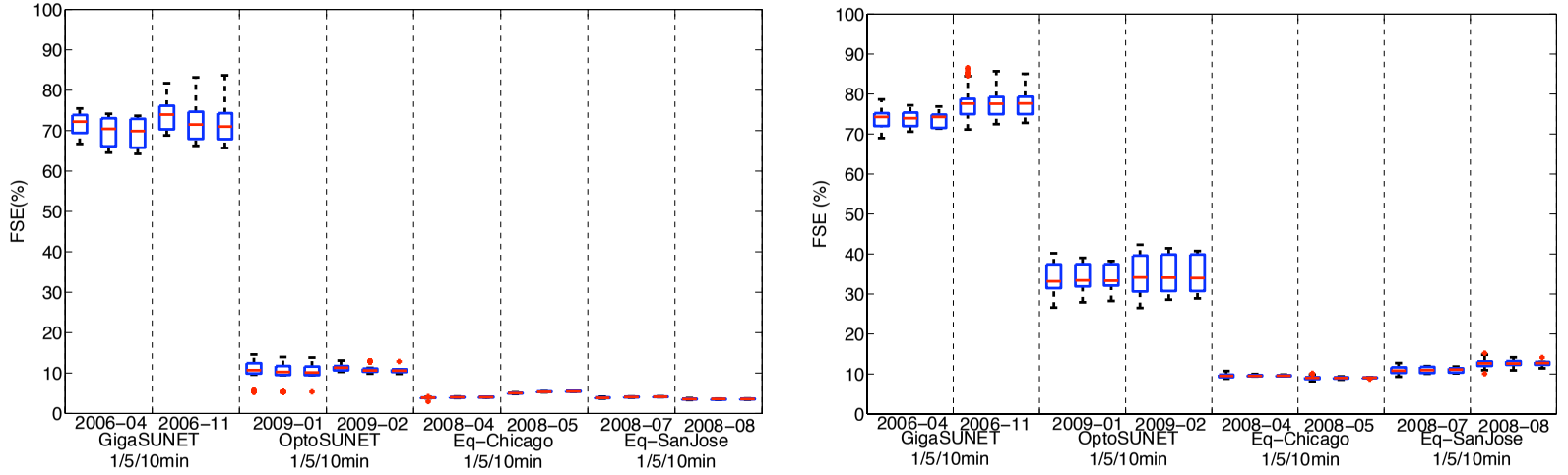
We apply FSE to the datasets of Table 1 and discuss the impact of traffic composition, observation interval and flow granularity on routing symmetry estimation.

4.1 The impact of inherently asymmetric traffic

To evaluate the impact of flows that are inherently asymmetric on traffic symmetry estimates, we first apply the method to all IP traffic, then on TCP traffic (i.e. disregarding UDP, ICMP and other traffic) and finally on the proposed category: TCP data traffic. The last category excludes nonproductive, inherently asymmetric TCP background radiation. Table 2 provides the excluded TCP-signaling fractions, a reasonable estimate for the amount of (asymmetric) TCP background radiation on our links, consistent with other studies [6, 14].



(a) Comparison of FSEs for IP, TCP and TCP-data flows (left) and bytes (right): statistics for 10-minute samples. FSEs increase when excluding inherently asymmetric traffic. FSEs in terms of bytes and packets (not shown here) are only slightly affected. Routing symmetry decreases towards the core of the Internet (from GigaSUNET close to the edge to Tier-1 backbone links Eq-Chicago and Eq-SanJose).



(b) Comparison of FSEs for TCP-data flows (left) and bytes (right): statistics for 1, 5 and 10-minute samples. Observation intervals have little effect on FSEs. FSEs are relatively stable both over short periods (low interquartile-ranges) and over long periods (between different months on the measured links).

Figure 2

Dataset		% flows	% packets	% bytes	Dataset		%flows	% packets	% bytes
GigaSUNET	2006-04	32.36	4.85	0.15	Eq-Chicago	2008-04	19.19	5.60	0.51
	2006-11	27.86	1.95	0.15		2008-05	23.62	4.31	0.34
OptoSUNET	2009-01	34.81	2.05	0.08	Eq-SanJose	2008-07	25.27	8.04	0.83
	2009-02	34.74	2.05	0.09		2008-08	19.41	7.75	0.78

Table 2: TCP traffic carrying only signaling packets, as removed by the TCP data filter. The numbers are good estimates for the amount of nonproductive TCP background radiation on the links.

Figure 2a provides box-plots[5] of flow-based symmetry estimates (FSEs) for 10-minute samples of traffic filtered in three ways. Due to space constraints we only show symmetry in terms of flows and bytes. As expected, the fractions of symmetric tuples increase when excluding inherently asymmetric traffic (e.g. from a median of 53% to 69% for GigaSUNET 2006-04 and from 2.7% to 5.5% for Eq-Chicago 2008-05). But the filtering operation only slightly affects symmetry in terms of bytes (e.g., from 8.7% to 9.0%) and packets (e.g. from 73% to 74%, not shown here), since packets carrying TCP signaling flags are a minor fraction of the total TCP packets and typically carry no data (see Table 2).

Figure 2a also suggests that the degree of routing symmetry radically decreases as we move toward the core of the Internet. On GigaSUNET, inside a Tier2 network close to the edge of the Internet, most traffic we observed was routed symmetrically (around 70%). The asymmetric traffic fraction here is caused by hot-potato routing due to local peering and the underlying ring architecture which does not guarantee shortest-path transport. One step closer to the core, on the OptoSUNET link connecting a Tier2 to a Tier1 network, only about 10% of the observed flows were symmetrical. On this link asymmetry can be explained by the load-balancing policy applied on the redundant route between SUNET and NorduNet (see Section 3) as well a regional exchange point introducing some hot-potato routing. On the two Tier1 ISP backbone links, hot-potato routing and other peering artifacts in aggregation induce high asymmetry: only 4-5% of tuples generate traffic routed symmetrically.

4.2 The impact of observation intervals

The observation interval used for the analysis impacts flow, and thus symmetry, assessments. Short intervals introduce border effects, such as causing short symmetric flows to seem asymmetric if packet exchange occurs at the edge of an interval. Large intervals increase the probability of incorrectly aggregating multiple sessions with identical 5-tuples into one flow within the interval.

To evaluate the impact of these effects, we split each traffic trace into feasible chunks[6] of 1, 5, and 10-minutes, and apply FSE to filtered TCP data traffic within each observation interval. Figure 2b shows box-plots of the FSEs, reflecting symmetrically routed traffic in terms of tuples and bytes for each time interval (we omit packets again). Observation intervals shorter than 10 minutes have little effect on routing symmetry estimates, which are stable (low interquartile-range) over the entire dataset samples (six 10-minute samples across one month

[5]Boxes represent median, lower and upper quartile, plus whiskers and outliers.

[6]intervals >10min on large backbone traces may exhaust memory (e.g. 10min of SanJose0807: 2.7GB for 23M flows).

TCP-data traffic 10-min samples		% of tuples flow	% of tuples IPpair	% of packets flow	% of packets IPpair	% of bytes flow	% of bytes IPpair
GigaSUNET	06-04	69.4	79.4	73.6	73.7	73.9	73.9
	06-11	72.3	77.9	78.1	78.1	77.9	77.9
OptoSUNET	09-01	10.1	10.7	25.3	25.4	33.8	33.9
	09-02	10.9	11.7	24.5	24.6	34.7	34.8
Eq-Chicago	08-04	4.0	3.3	9.0	10.3	9.6	11.6
	08-05	5.5	5.2	9.9	11.8	9.0	11.7
Eq-SanJose	08-07	4.1	3.5	9.3	11.8	11.0	13.8
	08-08	3.6	4.2	10.7	14.0	12.7	16.3

Figure 3: Mean FSEs computed by considering TCP data traffic exchanged between 2-tuples (IPpairs) and 5-tuples (TCP flows), and how this aggregation granularity affects FSEs. Higher symmetry values in the IPpairs follow from the fact that the method counts all traffic generated by two 5-tuples with the same source and destination IP as symmetric even if only one 5-tuple is actually observed as symmetric. In fact, the total number of packets (bytes) remains unchanged regardless of granularity. In terms of tuples, traffic granularity affects the degree of symmetry, depending on the fraction of flows that share the IP pairs.

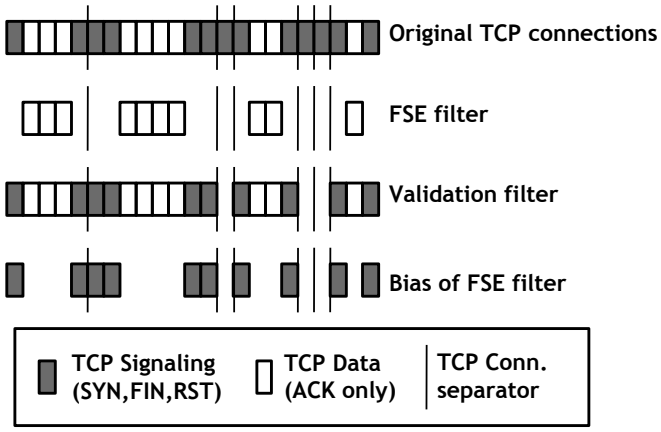


Figure 4: FSE removes purely signaling and scanning packets prior to flow creation. The validation method filters out TCP background radiation by retaining only connections with at least one non-signaling packet.

for SUNET data, and within one continuous hour for Equinix data). Moreover, we observe that the symmetry estimate computed on TCP data traffic remains stable on each location over time (comparing FSEs of data samples separated by seven months for GigaSUNET, two months for the other locations), and this observation also holds for all IP traffic as well as for all TCP traffic (not shown).

In recent work [17], Lee and Brownlee studied traces measured during 24 hours on the network boundary of the University of Auckland in 2006, and showed that around 98% flows last less than 10 minutes. In the rest of this paper we will consider 10-minute samples, which minimize border effects but represents a meaningful statistical data sample.

4.3 The impact of traffic granularity

In this subsection we compare routing symmetry between two levels of traffic granularity: IP pairs, more relevant to routing questions [2]; and flows, more relevant to traffic analysis and classification techniques [7, 8, 9].

Figure 3 lists the mean values of the FSE metric calculated for 10-minute observation intervals of TCP data traffic. In terms of packets and bytes, IP pairs (which have higher levels of aggregation) often exhibit higher symmetry, indicating that flows between the same IP pairs may follow different paths.

5 Validation

This section validates our FSE metric against an approach using explicit TCP flags to distinguish bidirectional sessions, as described in John *et al.* [5]. The validation method considers TCP traffic in both directions, inspecting TCP-signaling flags (SYN, FIN, RST) to distinguish TCP flows. We define the percentage of symmetric 5-tuples as the fraction of connections with at least one packet for each direction. The amount of packets (bytes) carried by symmetric tuples yields packet- (byte-)symmetry. This flow definition classifies scanning behavior that re-uses 5-tuples as a series of 1-SYN-packet flows, while many common timeout-based flow definitions [11, 12] (often used as input for traffic classification tools [8, 9]), will label it as a single flow with multiple SYN packets. Figure 4 outlines the difference between FSE and the validation method applied to original TCP connections. The validation method filters out TCP background radiation by retaining only connections with at least one non-signaling packet. FSE filters out all signaling packets prior to flow creation. The filter discards scanning traffic, reducing the size of legitimate TCP sessions by its signaling packets and the respective header data.

Validation performed on a smaller validation dataset of one 10-minute interval from each dataset in Table 1 revealed that the interval-based flow definition as applied in FSE led to significant underestimation (between 14% and 31%) of the number of TCP connections. This underestimation derives from our aggregation of TCP connections into one flow if the exact five-tuple is re-used within the timeout interval. However, when considering (filtered) TCP data traffic, the underestimation is much slighter, 0.15%-0.45%. Table 3 shows the small impact of the FSE filter on symmetry assessments. These results indicate that legitimate TCP traffic (i.e. connections including SYN packets, data packets and RST/FIN termination), in contrast to TCP background radiation (often consisting of one signaling packet like SYN only), rarely reuses the same five-tuple for connections within 10 minutes, which demonstrates the utility of the proposed traffic filter. This fact further suggests that FSE is robust against varying flow definitions (i.e., timeout-based vs. signaling-based), at least for intervals less than 10 minutes.

In terms of packets and bytes, the validation shows that their absolute numbers are slightly higher than FSE estimated, since FSE aggressively discards signaling packets (see Figure 4). Table 4 shows this discrepancy during a ten-minute interval. On the complete validation dataset, FSE removed 1-7% of TCP packets, corresponding to 0.1-0.6% of bytes, before computing its symmetry estimates. However, this bias in absolute numbers has negligible effect on corresponding symmetry estimates, which shows the validity of the estimation.

Using the validation method to characterize background radiation in the datasets (quantified in Table 2), we can confirm that in our data background radiation is indeed mostly asymmetric: it is mainly composed of 1-pkt flows. Between 85% and 95% of the discarded connections are 1-SYN-pkt flows. Verification of the number of ICMP destination unreachable packets shows

10-min sample		TCP all		TCP data	
		F (%)	V (%)	F (%)	V (%)
GigaSUNET	2006-04	48.9	41.9	65.8	64.7
	2006-11	55.8	42.0	74.1	73.9
OptoSUNET	2009-01	8.0	6.8	9.7	9.7
	2009-02	9.5	7.8	12.9	12.8
Eq-Chicago	2008-04	3.5	3.0	3.9	3.9
	2008-05	4.7	4.0	5.6	5.5
Eq-SanJose	2008-07	3.3	3.0	4.2	4.2
	2008-08	3.2	3.0	3.7	3.7

Table 3: Flow level symmetry by FSE (F) vs. validation method (V) for all TCP [left] and TCP data [right] traffic. Flow symmetry differs greatly for all TCP traffic, but negligibly for TCP data traffic. Thus, TCP data traffic is robust against the different flow definitions (timeout vs flags).

Eq-Chicago 2008-05

Packets

	Sym.	Tot.	Sym.%
V	47.2M	469.8M	10.05%
F	45.7M	455.5M	10.04%
Diff.	1.5M	14.3M	

Bytes

	Sym.	Tot.	Sym.%
V	39.4G	433.6G	9.09%
F	39.3G	432.5	9.09%
Diff.	0.1GB	1.1G	

Table 4: FSE (F) vs. validation method (V). A small bias is introduced by the FSE TCP-data filter when discarding signaling packets. However, symmetry estimates are hardly affected.

that no more than 15% of the 1-SYN-pkt TCP flows receive ICMP packets in response. If we did not remove these sources of strong bias from the symmetry estimate, even exclusive access links (100% symmetric) could be erroneously perceived as having substantial routing asymmetry.

To further validate our estimation method, we collected two samples of 10min traffic on the 100Mb/s single access link which connects the edge router of the University of Brescia to the Internet [18]. Traffic that flows on this link is 100% symmetric, i.e. all outgoing and incoming packets follow this link, so this data can serve as ground truth to assess the effectiveness of the FSE mechanism. Estimating flow symmetry based on all IP traffic on the link resulted in an FSE of only 79%. Considering TCP traffic resulted in an FSE of 84%, which is closer to ground truth (100%) but still a significant underestimation. However, when assessing routing symmetry on our proposed category of TCP data traffic, FSE for flows resulted in >98%, and almost 100% of bytes and packets (>99.99%). The remaining underestimation of <2% of flows, which the FSE erroneously classified as asymmetric, can be attributed to border effects due to the observation interval: connections established/terminated just before/after the interval, which happen to send only one data packet within the interval, appear as asymmetric flows. Since this link carries relatively little P2P traffic (around 10%) [18], thus also little P2P signaling traffic (1-pkt UDP flows) [13], we believe that the positive effect of the TCP data filter could be even stronger for other links with more inherently asymmetric traffic.

6 Summary and Conclusions

In order to shed light on the assumption of routing symmetry often embedded into traffic analysis and classification methods, we provided insight into symmetric routing on a flow granularity using observations from a variety of Internet links. We do so by proposing a simple flow-based symmetry estimation method, FSE, providing a normalized metric allowing to assess and compare routing symmetry of links on flow level. We provide an open source tool implementing the proposed method, and apply it to a heterogeneous dataset, resulting in valuable reference data points on routing symmetry.

We designed FSE to leverage available tools providing traditional, timeout-based 5-tuple flows (e.g. CoralFlow). Since TCP is an inherently bidirectional protocol and still the dominant protocol carrying traffic on today's observable Internet, we established a TCP-based metric. We filtered out the inherently asymmetric TCP traffic (TCP background radiation), leaving only TCP packets without signaling flags. This process allows for fair comparison of symmetry across links with substantially different traffic decomposition.

We did use TCP signaling flags to validate our simplified metric against ground truth measurements, allowing us to demonstrate that our flow-based symmetry estimate (FSE) is robust against multiple flow definitions. We quantified the small bias of the filter and confirmed that most of the filtered nonproductive flows are asymmetric, carrying one packet only.

We also found that in the data we examined, spanning over four years, four measurement locations on two continents, 5-tuples carrying legitimate TCP data traffic are rarely reused within ten-minutes observation intervals. Shorter observation intervals do not significantly alter symmetry estimates. Aggregating traffic by IP pairs instead of flows often results in greater symmetry. Unsurprisingly, routing-based symmetry seems to be stable over hours and even months, and decreases as one moves from edge links to highly aggregated backbone, which also hinders examination of complete, bidirectional flows on a single link. This result implies that traffic analysis tools and methods should assume little routing symmetry unless intended only for stub access links with no path diversity.

Acknowledgements

The authors would like to thank Emile Aben for valuable discussions and Luca Salgarelli and Tomas Olovsson for useful feedback and comments. This research was supported in part by SUNET, the Swedish University network.

References

[1] Z. M. Mao, L. Qiu, J. Wang, and Y. Zhang, "On AS-level Path Inference," in *ACM SIGMETRICS*, Banff, Alberta, Canada, 2005.

[2] Y. He, M. Faloutsos, and S. Krishnamurthy, "Quantifying Routing Asymmetry in the Internet at the AS Level," in *IEEE GLOBECOM*, 2004.

[3] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The End-to-end Effects of Internet Path Selection," *SIGCOMM Computer Communication Review*, vol. 29, no. 4, 1999.

[4] V. Paxson, "End-to-end Routing Behavior in the Internet," *IEEE/ACM Transactions on Networking*, vol. 5, no. 5, 1997.

[5] W. John and S. Tafvelin, "Differences between In- and Outbound Internet Backbone Traffic," in *TERENA Networking Conference*, Copenhagen, DK, 2007.

[6] R. Pang, V. Yegneswaran, P. Barford, V. Paxson, and L. Peterson, "Characteristics of Internet Background Radiation," in *ACM Internet Measurement Conference*, Taormina, Sicily, Italy, 2004.

[7] L. Bernaille, R. Teixeira, and K. Salamatian, "Early Application Identification," in *ADETTI/ISCTE CoNEXT*, Lisboa, Portugal, 2006.

[8] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow Clustering Using Machine Learning Techniques," in *Passive and Active Measurement Conference (PAM)*, Antibes Juan-les-Pins, France, 2004.

[9] S. Zander, T. Nguyen, and G. Armitage, "Automated Traffic Classification and Application Identification using Machine Learning," in *IEEE Conf. on Local Computer Networks (LCN)*, Sydney, Australia, 2005.

[10] M. Crotti, F. Gringoli, and L. Salgarelli, "Impact of Asymmetric Routing on Statistical Traffic Classification," in *Proceedings of the GLOBECOM 2009 Conference*, Honololou, Hawaii, USA, 2009.

[11] K. Keys, D. Moore, R. Koga, E. Lagache, M. Tesch, and k claffy, "The Architecture of CoralReef: An Internet Traffic Monitoring Software Suite," in *Passive and Active Measurement Workshop*, Amst.,NL, 2001.

[12] B. Claise, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Internet Engineering Task Force, Oct. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3954.txt

[13] W. John, S. Tafvelin, and T. Olovsson, "Trends and Differences in Connection-behavior within Classes of Internet Backbone Traffic," in *Passive and Active Measurement Conference*, Ohio, USA, 2008.

[14] M. Allman, V. Paxson, and J. Terrell, "A Brief History of Scanning," in *ACM Internet Measurement Conference*, San Diego, USA, 2007.

[15] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic and Header Anomalies observed," in *ACM Internet Measurement Conference*, San Diego, CA, USA, 2007.

[16] "Real Time CoralReef Report Generator," http://www.caida.org/data/realtime (accessed 2010-01-14).

[17] D. Lee and N. Brownlee, "Passive Measurement of One-way and Two-way Flow Lifetimes," *SIGCOMM Comp. Comm. Rev. 37*, no. 3, 2007.

[18] A. Este, F. Gringoli, and L. Salgarelli, "Support Vector Machines for TCP Traffic Classification," *Computer Networks*, vol. 53, no. 14, 2009.

PAPER V

Wolfgang John, Min Zhang and Maurizio Dusi

Analysis of UDP Traffic Usage on Internet Backbone Links

Extended Report based on a Short Paper Published as:
Min Zhang, Maurizio Dusi, Wolfgang John and Changjia Chen
SAINT '09: Proceedings of the 9th Annual International Symposium on Applications and the Internet
Seattle, USA, 2009

Analysis of UDP Traffic Usage on Internet Backbone Links

Wolfgang John[1], Min Zhang[2], Maurizio Dusi[3]

[1]Chalmers University, Sweden. email: wolfgang.john@chalmers.se

[2]Beijing Jiaotong University, China. email: mia.minzhang@gmail.com

[3]Università degli Studi di Brescia, Italy. email: maurizio.dusi@ing.unibs.it

1 Introduction

It is still an accepted assumption that Internet traffic is dominated by TCP, as reported by e.g. Fomenkov et al. in 2004 [1], John and Tafvelin in 2007 [2] and continuously by Internet2 [3]. However, the rise of new streaming applications [4] such as IPTV (PPStream, PPLive) and new P2P protocols (e.g. uTP [5]) that try to avoid traffic shaping techniques (such as RST packet injection) will increase the use of UDP as a transport protocol. Since UDP lacks any functionality to adapt to network traffic congestion, a substantial increase in UDP usage might raise serious concerns about fairness and stability in the Internet.

The goal of this paper is to track the usage of UDP and shed light on the assumption that TCP is still the dominant transport protocol on the Internet. We evaluate the fraction of UDP traffic, in terms of flows, packets and bytes, on traces collected in the period 2002-2009 on several backbone links located in the US and Sweden. According to our data, the use of UDP as a transport protocol has gained popularity recently, especially in terms of number of flows. Our preliminary analysis suggests that most UDP flows use random high ports and carry few packets and little content (payload), consistent with its use as a signaling protocol for increasingly popular P2P applications [6]. Many such applications build overlay networks to exchange information about how to share specific (and typically large) files. UDP allows efficient establishment and maintenance of such an overlay network, while use of random ports evades detection by port-based traffic engineering or filtering techniques.

2 Datasets

For this study we analyzed packet-header traces from backbone links in the United States and in Sweden. The data from Sweden was collected on an OC192 link inside the GigaSUNET network during 2006, and on an OC192 connection link of the current OptoSUNET network. Traffic data from GigaSUNET includes 40min (2x20min) collected in April and another 2x20min in November 2006, summing up to 9M flows carrying 422M IP packets and 294GB of data. Two samples of 20-minute each were also collected from OptoSUNET in January and February

2009, and include 41M flows, 1100M packets and 657GB of data [7]. Note that the Opto-SUNET data include a substantial portion of traffic on UDP port 53, due to the presence of a RIPE DNS server located inside SUNET, serving over 400 zones. Traffic coming from and going to port 53 on this server cannot be considered native SUNET traffic and we filtered it out for this study.

The data from the US was collected on an OC48 peering link for a large ISP and on an OC192 backbone link. Two 60-minute traces were collected on the OC48 link, one in August 2002 and one in January 2003. The OC48 traces include 105M flows, 1834M packets and 1105GB of data. Traces from the OC192 link are also 60-minute long samples, one collected in April 2008 and another one in February 2009, and consist of 379M flows, carrying 8434M packets and 4446GB of data in total [8].

3 Methodology

We used CoralReef [9] to extract unidirectional UDP *flows* from our traces. Each flow record, defined by the five-tuple (source and destination IP, port numbers and protocol), includes the counts of packets and bytes exchanged. The flows are further discriminated by a 5-minute time-out interval, i.e., two packets sharing the same tuple belong to the same flow if their timestamps are within the given time interval. Since unidirectional UDP flow information does not allow us to infer if the specific flow is client-to-server (i.e., request) or server-to-client (i.e., response) traffic, we use a simple heuristic to choose the port number for the port-based analysis. We take advantage of the fact that well-known and common server applications are assigned to low port numbers ($<$1024), and many operating systems use the high end of the port range as ephemeral ports for dynamically assigned source port numbers[1]. We therefore consider the lower of the two port numbers within a flow 5-tuple for our port-based analysis, which is most likely to be the destination (i.e., server) port number.

4 Analysis of UDP traffic

In Table 1 we report the ratio between UDP and TCP traffic, in terms of packet numbers, traffic volume (bytes) and flow numbers. For each of the backbone measurement locations, we can observe that the use of UDP as a transport protocol has increased through the years. While TCP sessions are still responsible for most packets and bytes transferred, UDP dominates after 2003 in terms of flows: on OptoSUNET (2009) we observe 3x as many UDP flows as TCP flows.

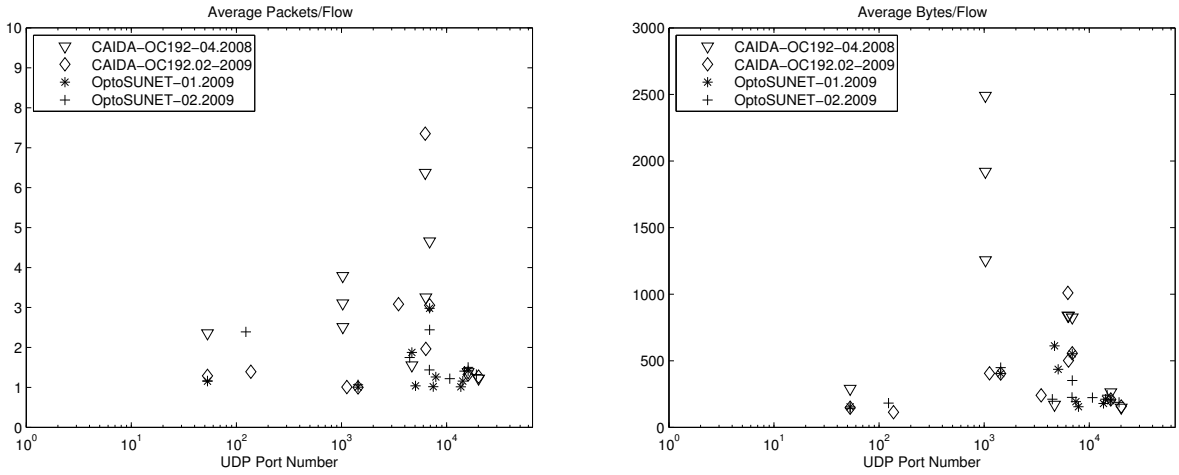
4.1 Port number analysis

To investigate the nature of the UDP traffic, we first took flows from the top ten port numbers in terms of flows for the most recent data (2008 and 2009) and plotted their average flow sizes

[1]IANA suggests ports 49152-65535 as dynamic ports, which is followed by newer Windows based Systems. Linux uses an ephemeral port range of 32768-61000 by default.

Trace	Sample	UDP/TCP Ratio		
		packets	bytes	flows
CAIDA-OC48	08-2002	0.11	0.03	0.11
	01-2003	0.12	0.05	0.27
CAIDA-OC192	04-2008	0.14	0.05	1.43
	02-2009	0.19	0.07	2.34
GigaSUNET	04-2006	0.06	0.02	1.06
	11-2006	0.08	0.03	1.45
OptoSUNET	01-2009	0.21	0.11	3.09
	02-2009	0.20	0.11	2.63

Table 1: UDP/TCP ratio in terms of packets, traffic volume (bytes) and flow numbers on our datasets.



(a) Average packets per flow

(b) Average bytes per flow

Figure 1: Top ten UDP ports in terms of flows numbers. Port numbers (x-axis) are plotted in log-scale.

in terms of packets and bytes (Figure 1). On the backbone links in Sweden and the US, flows of the ten top-ranked ports generally carry fewer than 8 packets which sum up to less than 1KB on average. This suggests usage of UDP mainly for small (signaling) flows. According to a port-based classification, the top flows include the traditional UDP applications DNS (53) and NTP (123). Other top port numbers are ports used for applications with known vulnerabilities, such as port 137 (Netbios) and port 1434 (MSSQL) [10]. Flows to these port numbers are therefore likely to be anomalous and unsolicited traffic. The majority of the top-used ports in terms of UDP flows, however, are the ones normally used by P2P applications, such as 4672 and 4665 (eDonkey), 6881 and 32459 (BitTorrent), 6346 (Gnutella), 4672 (eMule) and 6257 (WinMX).

On the CAIDA traces from 2008, we can observe three outliers in Figure 1b. These outliers are caused by traffic on the UDP port numbers 1026, 1027 and 1028, which we believe can be

attributed to anomalous activities directed to the Windows Messenger Service[2] [11]. The UDP traffic on these ports is exchanged between large numbers of IP addresses, but a major part of the data is destined to few IPs, which differ for each specific port. We also noticed that the packet lengths in these flows are consistently 1106 bytes. Based on these observations, together with the knowledge of previously observed anomalies of this kind [12], we speculate that a burst of Messenger Spam (probably with spoofed source IP addresses) during trace collection is responsible for the outliers.

To give a complete overview of UDP port number usage, we plotted the Cumulative Distribution Functions (CDF) of UDP traffic in terms of flow numbers and traffic volume (in bytes) over the complete port range for each measurement location (Figures 2a and 2b). For the 2002-2003 traces, more than 50% of the UDP flows use ports below 1024, i.e., mainly traditional UDP services such as DNS (port 53), NTP (port 123) and NetBios (port 137). In the more recent traces from our measurement locations, usage of ephemeral ports (>1024) has become prevalent. Virtually all UDP ports are used, and in modern backbone traffic around 80-90% of the UDP flows use ports >1024 (Figure 2a). Observable steps in the CDFs are at port 53 (DNS) and 6881 (BitTorrent).

In term of traffic volumes, more than 95% of the UDP bytes are carried on higher ports numbers for all recent traces (2008, 2009), as illustrated in Figure 2b. Observable steps in the CDFs are at port 53 (DNS) and at port 6970 (Real Time Protocol, RTP). Furthermore, substantial numbers of traffic in the CAIDA data of 2003 trough 2009 are carried on UDP port 1457. The traffic on UDP port 1457 was in the traces from these three years generated by a small number of IP addresses, exchanging large data amounts with between two and ten servers on port 1457. We cannot infer the exact application, however, we do not consider this traffic representative due to the low number of hosts involved.

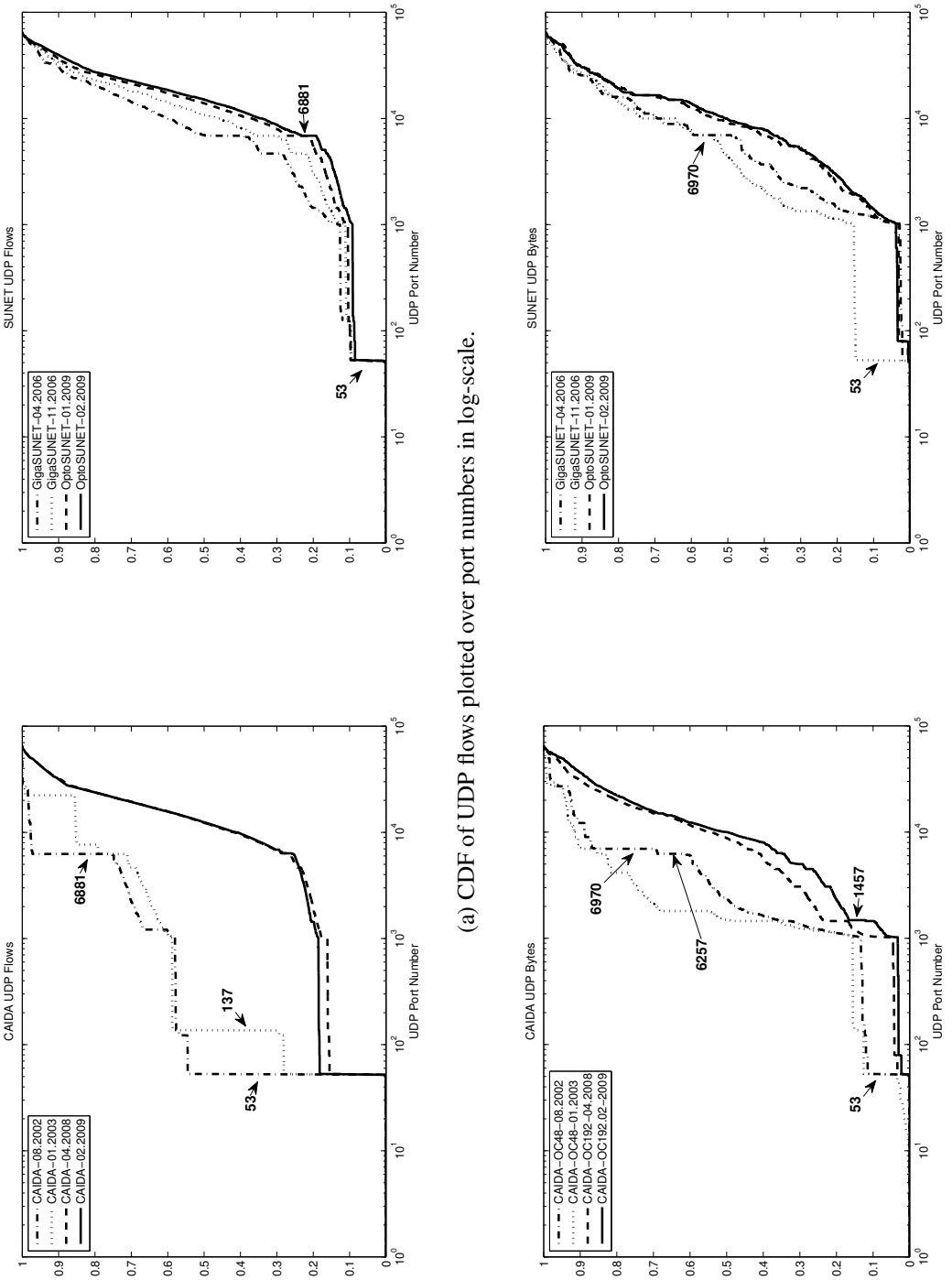
4.2 Flow size analysis

The results of the port-based analysis give us reason to speculate that the flows running on ephemeral ports can to a large degree be attributed to P2P overlay signaling traffic rather than to actual data transfers. To get more insight, we plotted CDFs of the flow sizes up to 1K for the 2008 and 2009 data traces (Figure 3). Our backbone traces show basically similar major steps: between 120 and 130 bytes, and between 300 and 310 bytes, most of these flows (> 90%) carrying one packet only. We looked at dominating port numbers in these flow ranges, and found many known P2P port numbers, esp. for BitTorrent (6881 and 32459).

To verify common UDP flow sizes of popular P2P applications, we complemented our backbone data with measurements of P2P traffic in a controlled lab environment[3]. The BitTorrent and eDonkey sessions collected on our lab computer confirmed that most UDP traffic generated by a popular P2P clients (uTorrent, eMule) falls in the observed byte ranges. The BitTorrent

[2] A vulnerability in the Windows Messenger Service on old, unpatched Windows Systems, typically on UDP ports 1026 et seq., is e.g., used for unsolicited advertisement of so called Windows Messenger Spam

[3] We installed uTorrent (a BitTorrent client) and eMule (an eDonkey client) on a WindowsXP PC. With the help of *Proxocket* (a dll proxy which allows to capture packets sent/received by a specific application) we collected UDP traffic generated by BitTorrent and eDonkey while downloading a Linux distribution from several peers.



(a) CDF of UDP flows plotted over port numbers in log-scale.

(b) CDF of UDP traffic volume in bytes plotted over port numbers in log-scale.

Figure 2: UDP port number usage in terms of flow numbers and traffic volumes

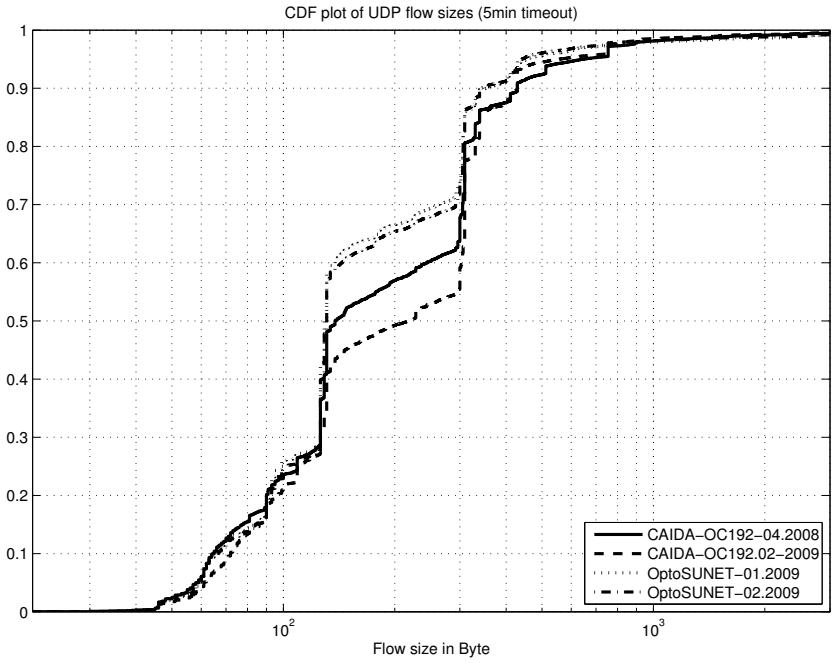


Figure 3: CDF of UDP flow sizes in byte.

ping and *find_node* Distributed Hash Table (DHT) queries with IP packet lengths of around 95 bytes (ping probes and responses), 125 bytes (find_node queries) and 300 bytes (find_node responses) [13]; and eDonkey UDP queries with comparable lengths.

We therefore believe that the main steps for backbone traces observable in Figure 3 are an artifact of P2P DHT signaling traffic (request and responses), which is a further indication that P2P signaling traffic is indeed responsible for a majority of the UDP flows on the observed backbone links. In Figure 3 we can also observe that about 80% of the flows in the traces from 2008/2009 carry less than 300 bytes. For the backbone data, only 2% of the UDP flows are larger than 1KB, and 0.01% are larger than 10KB, which explains the discrepancy in UDP/TCP ration between flow numbers and traffic volumes, as presented in Table 1.

5 Summary and Conclusions

We investigated UDP traffic in several traffic traces collected from different networks and geographical locations, at different times. We have found that TCP still dominates in terms of packets and bytes, but UDP is responsible for the largest fraction of flows in our most recent traces. A port-based analysis suggests that the increase in UDP flows in our data stems mainly from P2P applications using UDP for their overlay signaling traffic (e.g., DHT queries/responses).

This development may again change with the advent of IPTV and UDP based P2P applications, which not only signal, but also transport large datagrams via UDP [4, 5]. We will continue to monitor available data to track trends in UDP usage, and specifically seek data from China where UDP-based IPTV traffic is already common according to anecdotic reports [14].

The effect of increasing fractions of UDP traffic on network stability depends on the congestion control abilities of the applications utilizing UDP as main transport protocol (e.g. uTP [5]). The increasing trend of UDP therefore needs to be monitored closely in order to keep track of these effects.

Finally, we note that precise traffic classification requires methods beyond simple port classification. Most current traffic classification techniques focus on TCP [15, 16], with only preliminary examination of techniques for UDP traffic [17] (other than signature-based deep packet inspection). Given the growing evidence for the use of UDP for increasingly popular applications, we conclude that traffic analysis methods must evolve to be able to accurately classify UDP traffic.

Acknowledgements

This study was performed while the authors visited CAIDA[4] at UCSD under supervision of kc claffy. The authors furthermore want to thank Tomas Olovsson for valuable discussions.

References

[1] M. Fomenkov, K. Keys, D. Moore, and k. claffy, "Longitudinal Study of Internet Traffic in 1998-2003," in *WISICT: Winter International Symposium on Information and Communication Technologies*, 2004.

[2] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic and Header Anomalies Observed," in *IMC: ACM SIGCOMM Internet Measurement Conference*, 2007.

[3] "Internet2 NetFlow: Weekly Reports," http://netflow.internet2.edu/weekly/ (2010.01.23).

[4] P. Pan, Y. Cui, and B. Liu, "A Measurement Study on Video Acceleration Service," in *IEEE CCNC: Consumer Communications & Networking Conference*, 2009.

[5] "Micro Transport Protocol," http://en.wikipedia.org/wiki/Micro_Transport_Protocol (2009.11.23).

[6] W. John, S. Tafvelin, and T. Olovsson, "Trends and Differences in Connection-Behavior within Classes of Internet Backbone Traffic," in *PAM: Passive and Active Measrueemtn Conference*, 2008.

[7] W. John and S. Tafvelin, "SUNET OC192 traces," http://imdc.datcat.org/collection/1-04HN-W=SUNET+OC+192+Traces (2009.11.23).

[8] "CAIDA Passive Internet Backbone Traces," http://www.caida.org/data/passive/ (2009.11.23).

[9] D. Moore, K. Keys, R. Koga, E. Lagache, and k. claffy, "The CoralReef Software Suite as a Tool for System and Network Administrators," in *USENIX Conference on System Administration*, 2001, pp. 133–144.

[4] http://www.caida.org

[10] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, and N. Weaver, "Inside the Slammer Worm," *IEEE Security & Privacy*, vol. 1, no. 4, pp. 33–39, 2003.

[11] "Windows Messenger Popup Spam on UDP Port 1026," http://www.secureworks.com/research/threats/popup-spam/ (2010.01.24).

[12] R. Nelson, D. Lawson, and P. Lorier, "Analysis of Long Duration Traces," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 1, p. 52, 2005.

[13] "BitTorrent DHT Protocol," http://www.bittorrent.org/beps/bep_0005.html (2010.01.23).

[14] M. Zhang, Personal Communication.

[15] T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, 2008.

[16] M. Zhang, W. John, kc claffy, and N. Brownlee, "State of the Art in Traffic Classification: A Research Review," *PAM: Passive and Active Measurement Conference - Student Workshop*, 2009.

[17] T. Z. Fu, Y. Hu, D. M. Chiu, and J. C. Lui, "PBS: Periodic Behavioral Spectrum of P2P Applications," in *PAM: Passive and Active Measurement Conference*, 2009.

PAPER VI

Wolfgang John and Sven Tafvelin

Differences between In- and Outbound Internet Backbone Traffic

TNC '07: TERENA Networking Conference

Copenhagen, Denmark, 2007

Differences between In- and Outbound Internet Backbone Traffic

Wolfgang John and Sven Tafvelin

Department of Computer Science and Engineering
Chalmers University of Technology, Göteborg, Sweden
{firstname.lastname}@chalmers.se

Abstract

Contemporary backbone-traffic is analyzed with respect to behaviour differences between inbound and outbound Internet traffic. For the analysis, 146 traffic traces of 20 minutes duration have been collected in April 2006, carrying 10.7 billion frames and 7.5 TB of data. Significant directional differences, among others found in IP fragmentation, TCP termination behaviour and TCP options usage, are pointed out and discussed on different protocol levels (IP, TCP and UDP). The analysis includes a focus on TCP connection properties, yielding P2P and malicious traffic as main reasons for the differences. The results are relevant for a better understanding of how applied network protocols are used in an operative environment. Furthermore, a quantification of malicious traffic provides related research fields, such as network security or intrusion detection, with important insights.

Keywords

Internet Measurement; Directional Traffic Differences; TCP Connection Analysis; Network Anomalies;

1. Introduction

The Internet, as emerging key component for commercial and personal communication, has in the recent years undergone a fast development and is still expanding. Unfortunately, this rapid development has left little time or resources to integrate measurement and analysis possibilities into Internet infrastructure, applications and protocols. However, the Internet community needs to understand the nature of Internet traffic in order to support research and further development [3]. One way to acquire better understanding is to measure real Internet traffic. In the MonNet project [10][24], the technical and legal complications of the measurement task were overcome and resulted in packet-level data traces of contemporary Internet traffic.

The MonNet traffic traces analyzed in this article have been taken from the OC192 backbone of the Swedish University Network (SUNET) during 20 days in April 2006. The links tapped provide not only a backbone for two major Universities, but also for a substantial number of student dormitories and research institutes. Additionally, the links carry exchange traffic with commercial providers due to a local exchange point inside Göteborg. Because of the high

· This work was supported by SUNET, the Swedish University Network

aggregation of the measured links, we believe that this recent data provides a valid footprint of Internet traffic characteristics in Sweden at the current time.
The chosen measurement point on the outermost part of a ring architecture makes the traces specifically suitable for highlighting directional differences. Put simply, the measurements were taken on links between the region of Göteborg and the rest of the Internet. This work therefore analyzes the contemporary data with respect to behaviour differences between in- and outbound backbone traffic. The presented traffic constitutes a medium level of aggregation, between campus-wide traffic and tier-1 backbone traffic. We believe that this type of network, with smaller local exchange points, represents an upcoming class of networks.

1.1. Related work

There are numerous articles about general Internet measurements [7][16][25], with only a few of them partly dealing with directional differences. Thompson [25] e.g. presented wide-area Internet traffic characteristics on nowadays rather outdated data in 1997. The data was recorded on a core-backbone and a transatlantic link, including figures about directional differences in packet sizes and byte volumes.
In recent years, a few studies included discussions about directional differences, but typically only regarding specific properties. These articles are often based on unidirectional flow data and analyze a variety of datasets. The analyzed datasets are either collected at Tier-1 backbone level or on small campus or institute Internet gateways, so with either a low or very high level of aggregation. In his article about rapid model parameterization, Lan [14] showed differences between inbound and outbound traffic in terms of protocol mix and flow statistics, like flow size and duration. The data was recorded on the 100 Mbit/s Internet gateway of the USC Information Science Institute in 2001. Saroiu [22] analyzed different types of HTTP flows, recorded on two border routers of the University of Washington on 9 days in 2002. In this paper, WWW and P2P traffic carried over HTTP are contrasted, including a comparison of inbound and outbound flows. In his study about P2P properties in 2003, Gerber [8] was able to show that the IN/OUT traffic balance for P2P traffic on the border of a Tier-1 backbone is close to one. Kim [12] compared inbound and outbound flow statistics for different transport protocols, including flow, packet and byte ratios. The analysis was based on flow data collected in 2004 on the Internet routers of the POSTECH campus, a 2x100 Mbits/s Ethernet.
An interesting study based on packet-level traces was presented by Mellia [17]. Mellia analyzed traces collected on the Internet access link of the Politecnico di Torino campus LAN in 2000-2002. Besides presenting an automatic tool for statistical analysis of network traces, interesting results for IP and TCP characteristics are given, including a connection-level analysis of TCP.

1.2. Contribution of this work

Updated measurement results are crucial for a better understanding of how the applied technologies and protocols are used in an operative environment. In the present study, significant directional differences are pointed out and discussed on different protocol levels (IP, TCP and UDP). For TCP, the bi-direction packet level traces are reassembled to connections, in order to be able to conduct a detailed connection-level analysis. The presented results are destined for network engineers, network application developers and protocol designers in order to be able to optimize bandwidth efficiency and stability of future networks. The paper furthermore highlights network anomalies and inconsistencies, like attacking or scanning traffic. This is important knowledge, since improving the robustness of network applications and protocol implementations is gaining special importance. In fact, increasing bandwidth and growing numbers of Internet users have also lead to increased misuse and anomalous behaviour [9][13]. Knowledge of real-life traffic properties is also important for establishing more realistic simulation models [6]. Finally, some of the insights might as well bring up new research issues in related research fields, such as network security and intrusion detection. The contributions of this work are relevant, because:

- the analysis is based on updated, contemporary data
- the data was collected on links representing medium traffic aggregation, a class of networks not previously studied in the same extent
- packet-level traces allow a more detailed analysis than sampled flow-level data (e.g. TCP options)
- the presented bi-directional TCP connection analysis reflects real connections more closely than traditional flow level analysis
- the results provide a complete view of directional differences on different levels (IP, TCP, UDP)
- the special focus on network anomalies is especially important in the light of increasing amounts of network attacks

The paper is outlined as follows. Section 2 describes the methodology of collecting, pre-processing and analyzing the traces. Then some general traffic properties are presented in section 3. Next, sections 4, 5 and 6 quantify directional differences observed on different protocols levels (IP, TCP and UDP). Finally, in section 7, different traffic anomalies and inconsistencies found on the protocol levels are summarized, followed by concluding remarks about the main findings.

2. Methodology

2.1. Collection of traces

We collected our traces on the outermost part of an SDH ring running Packet over SONET (PoS). The traffic passing the ring to (outbound) and from (inbound) the main Internet is primarily routed via our tapped link, as confirmed by SNMP statistics. Simplified, we regard

the measurements to be taken on links between the region of Göteborg, including exchange traffic with the regional access point, and the rest of the Internet as discussed earlier in section 1.

We use optical splitters on two OC-192 links, one for each direction. The splitters are attached to two Endace DAG6.2SE cards sitting in identical Dual-Opteron servers. The servers use a 6 disk SCSI Raid0 to keep up with the speed of the 10 Gbit/s links. The DAG cards are configured to capture the first 120 bytes of each frame to ensure that the entire network- and transport header information is preserved. The two DAG cards are chained together with help of the DAG Universal Clock Kit (DUCK), with one card serving as synchronisation input for the second card, resulting in time synchronisation typically between ± 30ns [5].

The collection of the data was performed between the 7th of April, 10:00 and the 26th of April 2006, 10:20. During this period, we simultaneously for both directions collected four traces of 20 minutes each day at identical times. The times (02:00, 10:00, 14:00 and 20:00) were chosen to cover business, non-business as well as night time hours. Due to measurement errors in one direction at four occasions we have excluded these traces and the corresponding traces in the opposite direction.

2.2. Processing and analysis

After storing the data on disk, the payload beyond transport layer was removed and the traces were sanitized and desensitized. This was mainly done by using available tools like Endace's dagtools and CAIDA's CoralReef, accompanied by own tools for additional consistency checks, which have been applied after each pre-processing step to ensure sanity of the traces. Trace sanitization refers to the process of checking and ensuring that the collected traces are free from logical inconsistencies and are suitable for further analysis. During our capturing sessions, the DAG cards discarded a total of 20 frames within 12 different traces due to receiver errors, which includes HDLC CRC errors. Surprisingly, another 71 frames within 30 different traces had to be discarded after the sanitization process due to IP checksum errors.

By desensitization we mean the removing of all sensitive information to ensure privacy and confidentiality. The payload of the packets was removed earlier, so we finally anonymized IP addresses using the prefix preserving CryptoPAN [27]. After desensitization, the traces were moved to a central storage server. First, an analysis program was run on each trace to extract cumulated statistical data. As a second step, per-connection TCP analysis was conducted on merged, then bidirectional traces. More details on the connection analysis are described in beginning of section 5.

3. General traffic characteristics

As summarized in table 1, the 146 analyzed traces sum up to 10.68 billion PoS frames, containing a total of 7.53 TB of data. In his study on campus wide traffic, Kim [12] reported about a 1:1 ratio between outbound and inbound traffic for packets numbers, but an 1:1.38

inequality for traffic volume due to the "net provider" status of University networks. In the our data, no significant difference between neither, packet counts nor volumes, can be observed. This even distribution of traffic proves the higher level of aggregation and underlines the relevance of the presented data, representing Internet backbone traffic.

The frames contain in 99.97% of the cases IPv4 packets, which sum up to 99.99% of the carried data. The remaining traffic consists constantly of around 1200 IPv6 BGP Multicast messages, 8 CLNP routing updates (IS-IS) and 1 Cisco Discovery Protocol (CDP) message per minute. The results in the remainder of this paper are based on the IPv4 traffic only.

	Packets		Data	
	Count	%	Volume	%
Total	10.68E+9	100.00%	7.53 TB	100.00%
Outbound		48.74%		49.16%
Inbound		51.26%		50.84%

Table 2: Traffic amount of data captured

	Total		Outbound		Inbound	
	Inside	Outside	Source	Dest.	Dest.	Source
Total	634E+3	22.0E+6	275E+3	19.2E+6	490E+3	19.8E+6
TCP	408E+3	05.0E+6	176E+3	04.3E+6	310E+3	04.5E+6
UDP	484E+3	19.2E+6	175E+3	16.4E+6	384E+3	16.9E+6
Rest	155E+3	01.9E+6	024E+3	01.1E+6	146E+3	01.0E+6

Table 1: Distinct IP addresses seen

4. IP level

In this section out- and inbound traffic on the network layer level of the Internet Protocol (IP) is compared. This comparison includes the transport protocol mix, IP packet size distribution and IP fragmentation.

To start with, table 2 gives some scale to the aggregation level of the links. The numbers of distinct IP hosts seen within (inside) and outside the region of Göteborg are summarized, where outbound sources and inbound destinations are regarded as inside, and the opposite way around as outside. Note that the sum of the numbers exceeds the total numbers, since one host can obviously be both source and destination for packets of several transport protocols. As expected, the amount of hosts inside the region is outnumbered by hosts seen outside "in the Internet". Nevertheless, there is a surprisingly high number of hosts inside, considering that the numbers of hosts at the three main customers of the links (2 major universities and the regional network for student dormitories) do not exceed 7,000 each. Indeed, these main customers sum up to about 21,000 sources of outbound TCP connections. The remaining 150,000 outbound sources belong to different providers connected to the exchange point. The amount of inbound destinations is much larger due to incoming scanning traffic. As an example, the 16 bit address ranges of the two Universities are scanned in their entirety (2x65,534). The vast amount of UDP hosts outside was found to be due to short UDP sessions caused by P2P overlay networks, which will be discussed in more detailed in section 6.

It has to be noted that even though the hosts of the three main customers represent a minor part (13%) of the observed IP addresses inside the region of Göteborg, a majority of the traffic (around 85%) consists of packets to or from these hosts.

4.1. Transport protocol breakdown

The protocol breakdown, summarized in table 3, once more confirms the dominance of TCP traffic. Compared to earlier measurements [7][16][23][25], the fractions of both TCP data volume and packet counts have even increased slightly. In the table, fractions of packets and data carried in the respective protocol are in % of total IPv4 traffic for the corresponding direction. Ratios between out- and inbound traffic are shown in parentheses, summing up to 100 for each protocol.

TCP packets and data show an equal ratio, as it was the case for the total traffic. In Kim's report [12], outbound traffic carried 1.44 times more data than inbound traffic. We believe that this behaviour is not observed in our data since the traffic of diverse network types aggregating on the links measured cancel out the typical client-server imbalance (small requests, large data replies). UDP data on the other hand shows almost the same ratio (38:62) in favour of inbound data volumes in our data as previously reported by Kim. This is caused by multimedia traffic (mainly RTP) over UDP, which is more common to be served on hosts on the Internet. An interesting observation can be made for UDP packets, with an unexpected large amount of outgoing packets. A closer look reveals that three consecutive measurements carried up to 58% UDP packets, as shown in table 4. This indicates a potential single UDP burst of 14-24 hours of time. A detailed analysis shows that the packet length for the UDP packets causing the burst was just 29 bytes, leaving a single byte for UDP payload data. These packets were transmitted between a single sender and receiver address with varying port numbers. After reporting this network anomaly, the network support group of a University could identify the culprit host. This was a web server that had been exploited through a known vulnerability in a PHP script. Consequently, a UDP DoS script was installed and could run undetected, since the network management tool was monitoring amount of per-flow data only, but not the number of packets. Although TCP data was still predominant, we believe that a dominance of UDP packets over such a time span could potentially lead to TCP starvation and raise serious concerns about Internet stability and fairness. When removing the three traces with this outstanding network event from our data, UDP packets showed the same ratio as the TCP and the overall data. Due to the small packet sizes, the ration of UDP data kept almost unchanged (36:64).

ESP traffic seems to experience a typical client-server pattern with even packet ratio, but uneven data proportions. The hosts mainly responsible for this type of traffic will be discussed again in section 4.3. An explanation for the dominance of outbound traffic for ICMP could be the large amount of incoming network attacks as shown later, triggering ICMP responses from routers and firewalls.

	Fraction of Packets in %				Fraction of Data in %			
	outbound		inbound		outbound		inbound	
TCP	90.62	(48.1)	93.14	(51.9)	97.76	(49.5)	96.57	(50.5)
UDP	8.87	(56.8)	6.40	(43.2)	2.03	(37.8)	3.23	(62.2)
ESP	0.23	(52.5)	0.20	(47.5)	0.12	(66.5)	0.06	(33.5)
ICMP	0.22	(61.9)	0.13	(38.1)	0.02	(60.7)	0.02	(39.3)
GRE	0.05	(51.8)	0.05	(48.2)	0.07	(73.0)	0.02	(27.0)

Table 3: Protocol mix (ratios per protocol in parenthesis)

		outbound	
Date	Time	Packets	Data
2006-04-16	14:00	6.8%	1.7%
2006-04-16	20:00	40.6%	5.1%
2006-04-17	02:00	51.9%	6.1%
2006-04-17	10:00	58.1%	7.1%
2006-04-17	14:00	5.7%	1.8%

Table 4: UDP burst

Packet size	total	outbound	inbound
20-39	1.50%	2.96%	0.11%
40-60	38.72%	37.26%	40.12%
576	0.96%	0.60%	1.29%
628	1.76%	2.05%	1.49%
1300	1.11%	1.20%	1.01%
1400-1500	38.01%	37.66%	38.34%

Packet size	total	outbound	inbound
20-39	0.14%	0.18%	0.11%
40-60	39.25%	38.41%	40.02%
576	0.98%	0.63%	1.30%
628	1.79%	2.12%	1.49%
1300	1.13%	1.25%	1.01%
1400-1500	38.53%	38.62%	38.45%

Table 5: Major modes of IPv4 packet size distribution for all data (left) and without

4.2. Packet size distribution

While cumulative distribution of IPv4 packet sizes was reported to be trimodal in earlier measurements [7][16][23][25], more recent studies showed that it has changed to be rather bimodal [21]. The two major modes are small packet sizes just above 40 bytes (TCP acknowledgements) and large packets around 1500 (Ethernet MTU). The previous third mode of 576 bytes (default size according to RFC 879) has in our data decreased to less than 1%. Furthermore, we found that two other notable modes appeared at 628 bytes and 1300 bytes. In table 5 the major modes are summarized, with an extra table excluding the above mentioned UDP burst. As discussed in a prior study on the SUNET datasets [10], the mode at 628 bytes is an artefact of 'TCP layer fragmentation' applied by file sharing protocols like Bittorrent or DirectConnect, where 628 byte large packets typically appear after full sized packets in order to add up to 2KB blocks of data. The mode at 1300 bytes could be explained by the recommended IP MTU for IPsec VPN tunnels [4].

The studies of Thompson, Kim and Mellia [12][17][25] report about directional differences in packets sizes on two different levels of link aggregation, both caused by the classical client-server pattern. In contrast, in the SUNET data the two main modes for small and large packets show no significant directional differences. This might be caused by two reasons:

- since Thompson's report of 1997, network applications have undergone some fundamental developments
- compared to the campus-wide data of Kim and Mellia, our backbone data contains a higher aggregated traffic mix

Directional differences however can be observed for two other packet sizes. The differences between fractions of 628 byte sized packets are likely to be caused by popular P2P servers inside Göteborg's student network. It is well known that DirectConnect, but also Bittorrent are especially popular in Sweden, and consequently also in the region of Göteborg. The cause

for the difference in the default datagram size of 576 bytes is not obvious, but we think it might be caused by a better utilization of the Path MTU discovery feature in the comparable well configured hosts inside University and student networks.

4.3. IP fragmentation

Earlier studies of McCreary and Shannon [16][23] indicated an increase in the fraction of IP packets carrying fragmented traffic of to up to 0.67%. We found a much smaller fraction of only 0.065% of fragmented traffic in the analyzed data, as shown in table 6. It can be noted that 72% of the fragmented traffic in our data is transmitted during office hours, at 10AM and 2PM. While Shannon, analyzing data of three different locations in 2001, found that fragmented data was equally distributed between out-and inbound data, the amount of fragmented traffic on the SUNET inbound link is about 9 times higher than on the outbound one. Where UDP and TCP is responsible for 97% and 3% respectively of all incoming fragmented segments, they just represent 19% and 18% of the outgoing. The remaining 63% outgoing fragmented traffic turned out to be IPsec ESP traffic (RFC 4303) between exactly one source and one receiver at working hours on weekdays. Each fragment series in this connection consists of one full length Ethernet MTU and one additional 72 bytes fragment. This could easily be explained by an unsuitably configured host/VPN combination transmitting 1532 byte (1572-40 byte additional IP and TCP header) instead of the Ethernet MTU due to the additional ESP header. The dominance of UDP among fragmented traffic is not surprising since Path MTU Discovery is a TCP feature only.

	Total	outbound	inbound
Total	0.065% (100.0%)	0.014% (100.0%)	0.113% (100.0%)
TCP	(4.5%)	(18.0%)	(2.9%)
UDP	(88.6%)	(18.8%)	(97.1%)
ESP	(6.8%)	(63.1%)	(0.0%)

Table 6: Fractions of IPv4 fragments

The first approach to explain the differences is based on the fact that the probability for a packet to be fragmented is increasing with each hop. According to a TTL analysis of the fragmented traffic, the average hop count for outbound traffic was 6.77, whereas the average hop count for inbound traffic was 9.43. This alone does not seem to be significant enough to explain the imbalance between inbound and outbound fragments. We believe that another possible explanation could again be the fact that SUNET and its connection networks are very well configured and administered compared to Internet standards.

5. TCP level

In order to conduct a detailed connection level analysis on TCP, we merged the tightly synchronized unidirectional traces. From the resulting bidirectional traces an analysis program collected per-connection data, including packet and data counts for both directions,

start- and end times, TCP flags and counters for erroneous packet headers and multiple occurrences of special flags like RST or FIN. We define a connection by the classical tuple of IP addresses and ports for source and destination. A TCP connection starts with the observation of the first SYN segment and is closed by either one FIN segment for each direction or one RST segment. Additional SYN segments for one tuple can sometimes be seen in the same direction, most commonly within scanning campaigns. In this case, further "connections" are opened within the analysis program in order to record the pure SYN packets separately. The following non-pure-SYN packets are always recorded within the most recently opened connection. We decided not to use a timeout threshold for unclosed connections, since our traces are limited to 20 min duration anyhow.

A significant part of the traffic is routed asymmetrically, due to hot-potato routing. 8% of the TCP data was sent via the outgoing link, without any corresponding TCP packets seen on the incoming. Asymmetrical traffic on the incoming link was even more common, accounting for 20% of the observed TCP data. Knowing the prefixes of the SUNET network segments in the area of Göteborg, it was possible to show that around 14% of the TCP data is actual transit traffic with neither source nor destination being SUNET customers inside Göteborg, entering the links via the local exchange point. Of the transit traffic, 67% was asymmetrical traffic, which means that 1/3 of all asymmetrically routed traffic is transit traffic as well.

In the following subsections, first, TCP connections are classified according to their connection setup and termination behaviour. Then, connection properties like packet count, byte size and lifetime are analyzed with respect to connection direction. Finally, TCP options are discussed in the rather novel approach of per-connection information for SYN requests and replies.

5.1. Connection breakdown

The following tables summarize the connection breakdown for TCP in all 146 traces. The analysis database recorded a total of 72.6 Million connections according to our definition. Additional 8.9 million bidirectional flows do not include an initial SYN segment, which means that they either start before the measurement times or have asymmetrical properties. One million of these flows include no SYN, FIN or RST segments but show packets in both directions, which means that about 3.4% of the established connections last longer than 20 minutes. However, this small number of long lasting connections carries about 34% of the total TCP data. This is not unexpected, given the observations of Brownlee [2], saying that flows longer than 15 minutes carry more than 50% of the traffic on a link. According to their destination port numbers, the long lasting connections typically carry traffic of different P2P protocols and popular messenger services. The following analysis is performed on TCP connections with initial SYN segments.

	total		outbound		inbound	
	Count	%	Count	%	Count	%
TCP connections	72.6E+6	100.00%	28.0E+6	38.56% (100.00%)	44.6E+6	61.44% (100.00%)
rejected	44.3E+6	60.99%	12.3E+6	(44.04%)	32.0E+6	(71.63%)
established	28.3E+6	39.01%	15.7E+6	(55.96%)	12.7E+6	(28.37%)

Table 7: TCP connection attempt breakdown

rejected connections	44.3E+6	100.00%	12.3E+6	27.84% (100.00%)	32.0E+6	72.16% (100.00%)
scanning - no reply	34.8E+6	78.66%	08.2E+6	(66.74%)	26.6E+6	(83.26%)
asymetric traffic	04.8E+6	10.84%	02.2E+6	(17.94%)	02.6E+6	(8.10%)
scanning - RST reply	04.3E+6	9.81%	01.7E+6	(13.83%)	02.6E+6	(8.25%)

Table 8: Rejected connection breakdown (no 3-way handshake)

Table 7 presents the total of all TCP connections with initial SYN segments. We define established and rejected connections as connections experiencing a proper 3-way handshake or not, respectively. Outbound in this context means that the initial SYN packet was sent on the outbound link. Inbound consequently means that the connection establishment was initiated outside the region of Göteborg. The tables 8 and 9 summarize the termination properties for rejected and established connections. In the tables, the first line represents the vertically summed values for each respective column of absolute packet counts or relative fractions. The fractions of out- and inbound connections in relation to the total amount of connections are additionally given in the first line, summing up to 100% horizontally.

Arlitt [1] quantified different TCP connection states based on the campus wide traffic recorded at the University of Calgary between 2003 and 2004. He quantified rejected connections with about 25-30% of all TCP connections. Our contemporary data includes much more unsuccessful connection attempts, as shown in table 7. A major difference between the numbers of rejected outbound and inbound initiated connections is evident in table 8. The large amount of unreplied SYN packets on the incoming link was already indicated earlier, when discussing the numbers of distinct IP addresses appearing on the incoming link. These are mainly attacks trying to exploit well known vulnerabilities on ports commonly used by Trojans. The scans often cover the entire IP ranges of the connected networks inside Göteborg and are likely to be destined for non existing endpoints. Entrance routers to the specific network typically drop this kind of packets, which explains the absence of response packets. In some cases an ICMP response might be triggered, which would explain the larger number of outgoing ICMP packets according to table 3. Regardless of the much higher number of incoming scans, there is also a substantial number of outgoing unreplied connection attempts. More than 70% of the 8.2 Million attempts are sent by hosts within the student network. Note that not all of these attempts are necessary network scans. There is a large fraction of non-malicious outbound connection attempts to non existing hosts, resulting in unsuccessful connection attempts. This is often observed for P2P traffic, where unreliable file-sharing peers are common.

In cases where scanning attempts reach existing hosts on arbitrary port numbers, host-based firewalls should preferable drop the packets, but might in some cases reply immediately with an RST packet. This behaviour is more than twice as common for hosts in the student network as compared to hosts in University networks, which indicate that private Internet hosts are less carefully configured.
Asymmetric traffic was included in the summary for rejected connections (table 8) for reasons of completeness. Naturally, asymmetric traffic can not experience a bidirectional 3-way handshake, which means that we cannot consider this traffic as being established.

	total		outbound		inbound	
	Count	%	Count	%	Count	%
established connections	28.3E+6	100.00%	15.7E+6	55.21% (100.00%)	12.7E+6	44.68% (100.00%)
proper closing (2xFIN)	19.0E+6	66.99%	11.4E+6	(72.87%)	07.6E+6	(59.71%)
FIN and RST outbound	03.2E+6	11.21%	542E+3	(3.46%)	02.6E+6	(20.81%)
FIN and RST inbound	01.7E+6	6.06%	711E+3	(4.54%)	01.0E+6	(7.93%)
single RST	02.2E+6	7.71%	01.6E+6	(9.98%)	620E+3	(4.89%)
FIN, RST in counter dir.	01.2E+6	4.11%	889E+3	(5.67%)	276E+3	(2.18%)
unclosed	01.0E+6	3.63%	487E+3	(3.11%)	540E+3	(4.27%)

Table 9: Established connection termination breakdown

In table 9 finally the 28.3 Million connections with proper 3-way handshake observed are split up into different termination behaviours per direction. Considering the quite even distribution of TCP traffic volumes (table 1) it is somewhat surprising to see around 10% more outbound than inbound established connections. These differences in connection counts are cancelled out in the high level summary by differences in connection properties, as presented in the next subsection.
A major fraction (67%) of the established connections is closed properly by FIN segments in each direction, which seems to be quite low, considering that TCP resets should be a rare event according to the TCP standard (RFC 793). On the other hand, a prior study by Arlitt [1] highlighted that TCP connections are becoming more likely to be closed by RST segments (15%), mainly due to irregular web server and browser implementations. Comparing the behaviour of in- and outbound connection in our data we find that connections opened from inside Göteborg are more likely to be closed by proper FIN handshakes. This is compensated by a higher number of connections involving RST segments on the incoming link. While single RSTs in either direction can still be regarded as proper connection termination, the number of connections closed by FIN, followed by additional RST segments is surprisingly high (more than 30% on the inbound connections), even when considering Arlitts results. In fact, the fractions of connections closed by both FIN and RST segments sent by the client (the originator) are close to Arlitts numbers. (3.5% and 7.9% resp.) and are indeed mainly caused by web traffic. The main surprise is the large numbers of connections terminated by FINs and RSTs sent by the server (the responder), which are unproportionally large for inbound connections, meaning that they are closed by servers inside Göteborg. As main source of this behaviour a handful of hosts inside the student network could be identified,

according to their port numbers serving different kinds of popular P2P protocols. This reset behaviour is probably used to reduce the CPU and memory overhead introduced by connections entering the TIME_WAIT state on peers [1].
The 3.6% of unclosed connections lies close to the fraction of long-lasting connections, quantified in section 5.1. These unclosed connections are indeed mainly long lasting flows, and consequently carry almost 50% of all data carried by established connections. While 50% of these unclosed, long lasting incoming connections show destination port numbers of popular P2P protocols, the same port numbers account only for 10% of the outbound connections.

In addition to the high number of connections consisting of one SYN segment only, we also observed as many as 57 Million connections consisting of RST segments only (not shown in the tables). Of these single RST segments, 96% are seen on the outbound link, almost entirely in asymmetrical fashion, without any incoming segment triggering the resets. Only a handful source/destination pairs are responsible for these segments during short periods of time, so the first suspicion was that this could be reset attacks [26]. However, closer investigation showed no variations in sequence numbers or no other typical symptoms, so TCP reset attacks can be ruled out. We believe that the outbound link could be the return path for an asymmetrical routed denial of service (DoS) attack, generating the observed RST segments. Still, it is surprising that no similar behaviour could be observed to the same extent on the symmetrical routed data.

5.2. Quantification of P2P traffic

Since we expect P2P to have a huge impact on traffic characteristics, we tried to quantify P2P traffic for each direction with a simple port number analysis. Even though it is well known that P2P traffic is trying to hide itself and that port number methods strongly underestimate actual numbers [11][18], we believe that this analysis is still valid for comparing amounts of P2P connections between directions.
A list of common port-numbers for popular file-sharing protocols was identified, specifically for different DirectConnect, Bittorrent, Edonkey and Gnutella implementations. According to these port-numbers, outbound P2P connections carry around 13% of P2P packets and data, while for inbound connections this fraction is about twice as large with around 25%. Note, that these large volumes of data are carried by a small number of connections (less than 1%). Beside the probably quite large underestimation of these numbers, they indicate that P2P traffic is in fact at least about 2 times more common among inbound connections. The high amount of inbound established P2P connections, as already indicated in section 5.1, could be the result of a number of popular P2P peers inside Göteborg. Another possible explanation could be an increasing use of modern P2P clients (like RevConnect) inside Göteborg, triggering reverse connections from peers outside, on the Internet.

5.3. Connection properties

This section provides detailed information about different connection properties such as lifetime, size and packet count. The analysis deals only with bidirectional connections which have been established by a 3-way handshake. Ordering the TCP connections by data volume and number of packets carried shows that a small number of top connections accounts for most of the data and packets. This indicates the characteristically 'elephant and mice phenomenon', saying that the majority of Internet data is carried by a small percentage of large flows, so called elephants [15][20]. More specifically, outgoing connections appear to have less pronounced elephants, since it needs 0.08% and 0.17% to carry 50% of the total amount of data and packets respectively for outgoing connections, while only 0.07% and 0.14% are sufficient for 50% for inbound connections. This directional difference can be described even more clearly, considering that 3.9% of the outbound and as few as 0.9% of the inbound connections carry 90% of the data, and 26.3% and 12.2% respectively carry 90% of the packets seen in the particular direction.

Generally, artefacts of the client-server pattern (small requests, large data replies) can be observed for connections established in both directions. While outbound connections yield an average ratio of 1:1.6 in favour for incoming data, inbound connections show a higher ratio of 1:1.86 in favour of outgoing data. This means that the smaller number of inbound connections (around 45% of all connections) carries more data and more packets primarily in outgoing direction, according to the client-server pattern. This imbalance is cancelled out to an almost even ratio in the high-level view of sections 3 and 4. The imbalance in connections properties is mainly caused by the larger fraction of heavy incoming P2P connections.

The differences between in- and outbound connections are summarized in table 10 by means of statistical properties. In the table, mean, standard deviation (σ), median and 80th percentile (P80) are given for different connections properties per direction of the initial connection establishment. While mean and σ of connection lifetimes appear to be quite similar for both directions, the values for sizes and packet counts are significantly larger for inbound connections. It needs to be noted that some of the values and figures in this subsection are somewhat biased since the traces are limited to 20 min of duration. Long-lasting connections, which are likely to be elephants, are therefore not taken into account to the full extent. Especially the values for mean and σ can therefore to be considered as an underestimate, while median and P80 are less biased.

In order to be able to better interpret median and 80th percentile, we included figures for the distributions of connection lifetimes, sizes and packet counts. Figure 1 illustrates distribution of lifetimes in bins of 1sec. The magnified figure presents the first 25 seconds, with higher resolution of 15.6 ms bins. Figure 2 shows connection size distribution, summarized in bins of 1Kbyte. The insert magnifies the first 9 Kbytes with 20 Byte bin-size. Figure 3 finally illustrates packets counts, including magnification for the first 100 packets.

Property		mean	σ	median	P80
Lifetime in sec	out	18.2	60.7	1.8	16.6
	in	17.3	65.8	0.6	24.8
Size in Kbytes	out	61.0	2362	1.1	2.9
	in	81.5	3298	1.9	8.9
Packet Count	out	81.5	2289	11.5	22.0
	in	113.0	3538	11.5	21.0

Table 10: Statistical properties of TCP

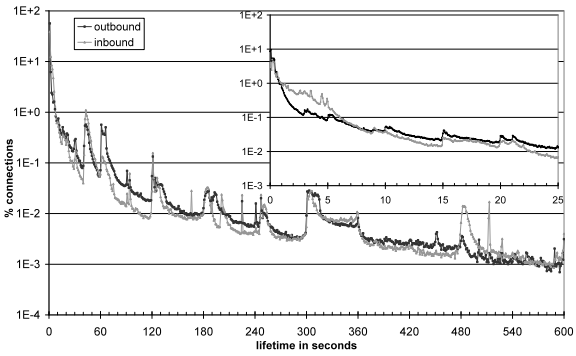


Figure 1: Conn. lifetimes with 1sec bins

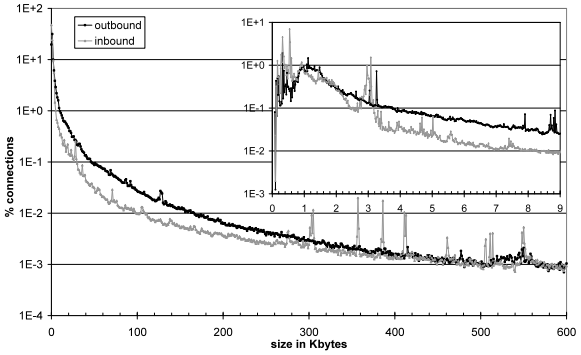


Figure 2: Conn. sizes with 1Kbyte bins (20

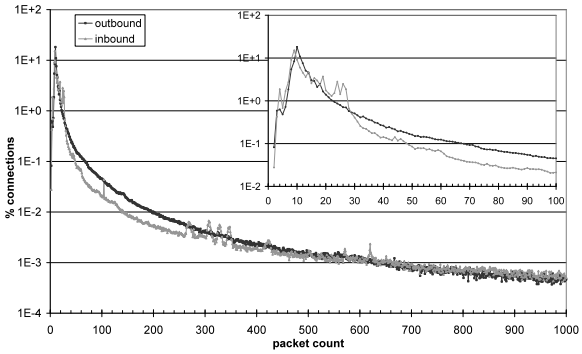


Figure 3: Packets per connection

Mori [19] presented mean values for flow durations on web and P2P flows extracted from inbound campus traffic in 2002. Web traffic yielded 9.5 sec mean, while P2P flow result in a mean of 307 sec. Projecting the values to our data, it can be concluded that the mean values of around 18 sec are a hybrid between web and P2P traffic, which is in fact the case due to University traffic on one hand and private student traffic on the other hand. Considering the underestimated nature of our values, it again indicates a quite substantial amount of long lasting P2P traffic on the measured links. Other studies, including Kim, Lan and Zhang [12][15][28], presented cumulative distribution figures, reporting of median values of about 1sec and P80 values of around 10 sec. In the SUNET data especially the 80th percentile is significantly larger, again proofing that connections in contemporary traces tend to be significantly longer due to an increased amount of P2P traffic. This property is more pronounced for inbound connections when comparing the P80 values for connection lifetime. Surprisingly, inbound connections do not only tend to be longer, but are also more likely to be shorter than 5 seconds compared to outbound connections. This is indicated by the median values, but can be seen nicely in the magnification of figure 1. The large number of incoming connections in this region can be explained by rejected login attempts on application level, like SSH or SMTP. In general, figure 1 shows a number of protocol timeouts, typically close to half minute or minute borders. For most of the times, the fractions of inbound connections lie below the outbound ones, which is compensated by a higher number of long lasting flows, as discussed earlier.

Regarding connection sizes, Mori [19] also presented mean values with 20.6 Kbytes for web flows, and as large as 5.8 Mbytes for P2P flows. As for lifetimes, the mean values for the presented data lie in between these extreme values. Earlier studies reported about median values of around 1 Kbytes and P80 values of between 1 and 10 Kbytes [15][28], which is similar to our findings. Even though in contrast to connection lifetimes, both median and P80 value are larger for inbound connections, there are peaks in the magnification of figure 2 for incoming connection sizes below 1Kbyte and around 3 Kbytes. According to a port analysis, the former stems from connections trying to exploit a known security hole in some MS SQL server versions on a handful of hosts inside Göteborg, while the latter can be explained by unsuccessful SSH login attempts, probably mainly intrusion attempts as well. Generally, figure 2 shows that inbound connections tend to be less likely to carry small amounts of data, which again indicates that there is a higher number of "elephants" carrying a lot of data on the incoming link. This seems to be connected to the similar behaviour found for connection lifetimes, even though there is not necessarily a strong correlation between duration and size, as reported by Lan and Zhang [15][28]. The spikes for inbound traffic seen in figure 2 between 300 and 550 Kbytes are results of connections from a single host to one host on destination port 2135. This is rather a special application than another security exploit, since except these small connections there are also a larger number of connections carrying a large amount of data between these hosts.

As illustrated in the magnification of figure 3, connections with less than 20 packets show very similar patterns for both directions, consequently resulting in similar median and P80 values. Nevertheless, the differences in the mean values as well as the lower values for the inbound graph in figure 3 shows that packet counts are to some degree correlated with connection sizes. As for connection sizes, the spikes between 20 and 30 packets are artefacts from unsuccessful SSH logins, and the spikes between 300 and 360 stem from the unidentified connections to port 2135.

5.4. TCP option usage

In earlier work, TCP options analysis was typically done by counting occurrences of different TCP options in all SYN and SYN/ACK segments seen in packet-level traces [10][21]. In our current work, the thorough connection analysis allows us to give better insight into options advertisements between clients and servers within single TCP connections. Since this analysis is focused on proper established connections only, attacking and scanning traffic, which might bias simple counts of SYN segments, are filtered out.

Table 11 summarizes TCP option employment for the four major TCP options types typically advertised during connection establishment. Fractions of connections carrying the particular option in SYN or SYN/ACK segments are given, split up for outbound and inbound established connections. The third column presents the fractions of connections advertising

the option in both initial segments, hence actually establishing the connection with the specific optional feature.

	MSS			WS			SACK			TS		
	SYN	SYN/ACK	both	SYN	SYN/ACK	both	SYN	SYN/ACK	both	SYN	SYN/ACK	both
outbound	100.00%	99.59%	99.59%	19.36%	15.46%	15.46%	93.67%	69.70%	69.70%	16.50%	12.32%	12.32%
inbound	99.94%	99.92%	99.85%	24.33%	23.85%	23.83%	97.22%	90.40%	90.38%	19.72%	18.51%	18.50%

Table 11: TCP options for inbound and outbound connections

In general, the numbers are in range of the reported values of the previous studies. The maximum segment size option (MSS) is used extensively by clients and servers for both directions. To our surprise, the window scale (WS), timestamp (TS) and selective acknowledgement permitted (SACK) options on the other hand are about 1.5 times more common among inbound connections. Looking at the destination port numbers for these connections, the difference can be explained by a much more diverse mix of applications among inbound connections in favour of primarily web traffic on port 80 in outgoing connections. The incoming connections include large fractions of recognized P2P protocols, but also substantial amounts of SMTP, SSH and MS SQL sessions, which are mainly break in attempts as discussed in section 5.3. These protocols are often used to carry more data than conventional web traffic, so it seems natural that clients and servers are interested in optimizing throughput by use of these TCP options.

6. UDP level

Since UDP offers no connection establishment or termination, we defined UDP flows as the sum of bidirectional packets observed between a specific tuple of source and destination IP and port numbers, taking advantage of the timeout value of 20 min given by the trace duration. In the 2x73 network traces, 68 million such UDP flows have been observed, carrying around 7% of the packets and only 2-3% of the data

Interestingly, 51 out of the 68 Million UDP flows (76%) carry less than 3 packets in either direction. Our first guess, that classical UDP services like DNS and NTP would be primarily responsible for these flows, proved to be wrong. In fact, only 5% and 1.7% of the small UDP flows serve DNS or NTP requests, respectively. On the other hand P2P overlay networks, such as Kademlia or other distributed hash table (DHT) protocols, are responsible for at least 18% of these small flows, where we expect this naïve port analysis to be a huge underestimate again. The purpose of these overlay networks is to keep the peers routing tables updated in a completely decentralized fashion. This is done periodically by sending DHT “pings” in small UDP packets, replied by the recipient. No significant difference between inbound and outbound DHT queries could be observed, which makes sense when considering the type and the nature of these overlay networks.

Based on the simple port classification, different common network attacks on UDP port numbers for MS SQL, MS messenger “spam” or Netbios were found to be responsible for at

least in 8% of the 51 Million short flows. These flows consisted in more than 90% of the cases of one inbound packet only, sometimes performing scans on entire IP ranges.
The two main sources for UDP flows, P2P overlay networks and attacking traffic, finally also explain the extreme amount of distinct IP addresses seen on the outside of the links measured (presented in table 2) since P2P network span the entire globe and experience a very high fluctuation in peering partners.

7. Summary and Conclusions

We presented directional differences found on recent packet level traces taken on links with medium aggregation level, carrying traffic from two major Universities, about a dozen of large student dormitories and a local exchange point. Since access to contemporary traffic on highly aggregated links is still uncommon, we believe that this study can contribute to a better understanding of the changing behaviour of the Internet. After short discussions about the two main factors responsible for the observed directional differences in our traces, malicious traffic and P2P traffic, this paper will be closed with summarizing conclusions.

7.1. Malicious traffic

Already the protocol breakdown revealed one outstanding long-duration UDP DoS attack originated within a major University in Göteborg, due to an DoS script injected from outside by exploitation of a known vulnerability. The fact that this attack was undetected by the network management tools in operation indicates the need for continuous refinement of network monitoring policies.
Despite this UDP burst, it can be said that basically every kind of malicious traffic is much more common in traffic coming from the main Internet. Already on a very high level analysis, incoming network scans were evident when analysing distinct IP addresses seen. There are about three times more rejected connections observed among inbound connections, with a majority of them being unreplied scanning attempts, but also a substantial number of immediate reset terminations. Around 90% of the counted header anomalies appeared on the incoming link, which goes hand in hand with the above mentioned scans. These packet header anomalies include inconsistencies in the IP flags, TCP header length and TCP connection flags field, which was discussed in more detail in an earlier study on the MonNet data [10]. Even though these header anomalies are very rare compared to the total number of packets, they indicated again skewed distribution of malicious traffic towards incoming traffic. The inconsistencies were shown to stem from network attacks trying to exploit protocol vulnerabilities as well as active OS fingerprinting tools.
Also the analysis of statistical connection properties within established connections revealed a large number of inbound login attempts to SSH, SMTP or MS SQL servers. Finally, on UDP level scanning traffic and security exploits were shown to happen in more than 90% of the cases within incoming traffic, which are as well in the order of millions in absolute numbers.

This summary of malicious behaviour confirms the suspicion that the main number of anomalies indeed originates on the outside, on the ”unfriendly” Internet. It was shown that anomalies are between 3 and 9 times more common among inbound data. Typical University campus networks, but even student networks, are comparable well behaving, probably due to higher configuration and administration efforts.

7.2. P2P traffic

Except the directional differences due to malicious traffic, P2P is a second source heavily influencing traffic properties. Even with a simple, underestimating port analysis, we could show that P2P traffic is a major part of the traffic, responsible for at least twice as much packets and volume among inbound traffic as compared to outbound traffic. Artefacts of P2P traffic were found in packet size distribution, TCP connection termination behaviour, TCP options and statistical connection properties. P2P traffic was also shown to be a major source for long-duration traces, especially among inbound connections. Additionally, P2P overlay traffic is responsible for the major amount of UDP flows, carrying typically less than 3 small sized packets, but being responsible for several millions of distinct IP addresses observed in the traffic. These short flows are furthermore hard to distinguish from malicious scanning or attacking traffic, which needs to be taken into consideration by network engineers and security experts working on sampled flow level analysis.

7.3. Conclusion

While some high-level analysis, like cumulated traffic volumes or protocol breakdown, could suggest an even distribution between inbound and outbound traffic, this study reveals that there are a number of significant directional differences found on different protocol levels. Especially the detailed TCP connection analysis, contrasting incoming and outgoing established connections by statistical means, revealed significant differences. Even though connections established in both directions show a typical client-server pattern, this behaviour is more pronounced among inbound connections. Generally, inbound connections, established from the outside, are shown to be more likely to carry larger volumes of data (elephants), larger number of packets and experience longer connection lifetimes. However, these differences, caused by the imbalance in P2P traffic, cancel out on high-level summaries because established inbound connections are on the other hand about 10% fewer than outbound connections.

First of all, the comprehensive analysis yielded required insights for network developers and traffic engineers. Furthermore, the results can be important input in order to improve quality and authenticity of future simulation models. Finally, the highlighted traffic anomalies are relevant for better understanding of security related issues like intrusion detection or detection of large scale attacks.

Acknowledgements

The authors want to thank Pierre Kleberger for his kind technical support and Tomas Olovsson for his valuable and constructive comments throughout the MonNet project.

References

[1] M. Arlitt and C. Williamson, "An Analysis of TCP Reset Behaviour on the Internet," Computer Comm. Review, vol. 35, 2005.

[2] N. Brownlee and kc claffy, "Understanding Internet Traffic Streams: Dragonflies and Tortoises," IEEE Communications Magazine, vol. 40, pp. 110-17, 2002.

[3] N. Brownlee and kc claffy, "Internet Measurement," IEEE Internet Computing, vol. 8, pp. 30-33, 2004.

[4] CiscoSystems, "IPsec VPN WAN Design Overview," Cisco Documentation, 2006.

[5] S. Donnelly, "Endace DAG Timestamping Whitepaper," Endace Withepapers, 2006.

[6] S. Floyd and E. Kohler, "Internet Research Needs Better Models," Computer Communication Review, vol. 33, pp. 29-34, 2003.

[7] C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely, and S. C. Diot, "Packet-level Traffic Measurements from the Sprint IP Backbone," Network, IEEE, vol. 17, pp. 6-16, 2003.

[8] A. Gerber, J. Houle, H. Nguyen, M. Roughan, and S. Sen, "P2P The Gorilla in the Cable," in National Cable & Telecommunications Association National Show. Chicago, IL, 2003.

[9] A. Householder, K. Houle, and C. Dougherty, "Computer Attack Trends Challenge Internet Security," Computer, vol. 35, 2002.

[10] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic with Focus on Header Anomalies," submitted for publication, Chalmers, Göteborg, Sweden, 2007.

[11] T. Karagiannis, A. Broido, M. Faloutsos, and kc claffy, "Transport Layer Identification of P2P Traffic," Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, Taormina, Sicily, Italy, 2004.

[12] M.-S. Kim, Y. J. Won, and J. W. Hong, "Characteristic Analysis of Internet Traffic from the Perspective of Flows," Computer Communications, vol. 29, pp. 1639-1652, 2006.

[13] K. Lan and A. Hussain, "The Effect of Malicious Traffic on the Network," Proceedings of the Workshop on Passive and Active Measurements (PAM), 2003.

[14] K.-C. Lan and J. Heidemann, "Rapid Model Parameterization from Traffic Measurements," ACM Transactions on Modeling and Computer Simulation, vol. 12, pp. 201-29, 2002.

[15] K.-C. Lan and J. Heidemann, "A Measurement Study of Correlations of Internet Flow Characteristics," Computer Networks, vol. 50, pp. 46-62, 2006.

[16] S. McCreary and kc claffy, "Trends in Wide-area IP Traffic Patterns - A View from Ames Internet Exchange," Cooperative Association for Internet Data Analysis - CAIDA, San Diego Supercomputer Center, San Diego 2000.

[17] M. Mellia, R. Lo Cigno, and F. Neri, "Measuring IP and TCP Behavior on Edge Nodes with Tstat," Computer Networks, vol. 47, pp. 1-21, 2005.

[18] A. W. Moore and K. Papagiannaki, "Toward the Accurate Identification of Network Applications," Lecture Notes in Computer Science, pp. 41-54, 2005.

[19] T. Mori, M. Uchida, and S. Goto, "Flow Analysis of Internet Traffic: World Wide Web versus Peer-to-Peer," Systems and Computers in Japan, vol. 36, pp. 70-81, 2005.

[20] T. Mori, M. Uchida, R. Kawahara, J. Pan, and S. Goto, "Identifying Elephant Flows Through Periodically Sampled Packets," Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, Taormina, Sicily, Italy, 2004.

[21] K. Pentikousis and H. Badr, "Quantifying the Deployment of TCP Options - a Comparative Study," IEEE Communications Letters, vol. 8, pp. 647-9, 2004.

[22] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An Analysis of Internet Content Delivery Systems," 5th symposium on Operating systems design and implementation, Boston, Massachusetts, 2002.

[23] C. Shannon, D. Moore, and kc claffy, "Beyond Folklore: Observations on Fragmented Traffic," IEEE/ACM Transactions on Networking, vol. 10, pp. 709-20, 2002.

[24] S. Tafvelin, "Presentation: QoS measurements," TERENA Networking Conference, Poznan, Poland, 2005.

[25] K. Thompson, G. J. Miller, and R. Wilder, "Wide-area Internet Traffic Patterns and Characteristics," IEEE Network, vol. 11, 1997.

[26] P. A. Watson, "Slipping in the Window: TCP Reset Attacks," Technical Whitepaper, 2003.

[27] J. Xu, J. Fan, M. Ammar, and S. B. Moon, "On the Design and Performance of Prefix-preserving IP Traffic Trace Anonymization," 1st ACM SIGCOMM Workshop on Internet Measurement, San Francisco, USA, 2001.

[28] Y. Zhang, L. Breslau, V. Paxson, and S. Shenker, "On the Characteristics and Origins of Internet Flow Rates," Computer Communication Review, vol. 32, pp. 309-322, 2002.

PAPER VII

Wolfgang John, Sven Tafvelin and Tomas Olovsson

Trends and Differences in Connection Behavior within Classes of Internet Backbone Traffic

PAM '08: Proceedings of the 9th Passive and Active Measurement Conference
Cleveland, Ohio, USA, 2008

Trends and Differences in Connection Behavior within Classes of Internet Backbone Traffic

Wolfgang John, Sven Tafvelin and Tomas Olovsson

Department of Computer Science and Engineering
Chalmers University of Technology, Göteborg, Sweden
{firstname.lastname}@chalmers.se

Abstract

In order to reveal the influence of different traffic classes on the Internet, backbone traffic was collected within an eight month period on backbone links of the Swedish University Network (SUNET). The collected data was then classified according to network application. In this study, three traffic classes (P2P, Web and malicious) are compared in terms of traffic volumes and signaling behavior. Furthermore, longitudinal trends and diurnal differences are highlighted. It is shown that traffic volumes are increasing considerably, with P2P-traffic clearly dominating. In contrast, the amount of malicious and attack traffic remains constant, even not exhibiting diurnal patterns. Next, P2P and Web traffic are shown to differ significantly in connection establishment and termination behavior. Finally, an analysis of TCP option usage revealed that Selective Acknowledgment (SACK), even though deployed by most web-clients, is still neglected by a number of popular web-servers.[1]

1 Introduction

Today, many network operators do not know which type of traffic they are carrying. This problem emerged mainly in the early 2000's, when P2P file sharing applications started to disguise their traffic in order to evade traffic filters and legal implications. Since then, the network research community started to draw increasing attention to classification of Internet traffic. Traditional port number classification was shown to underestimate actual P2P traffic volumes by factors of 2-3 [1], thus more sophisticated classification methods have been proposed. These methods are typically either based on payload signatures [2], statistical properties of flows [3] or connection patterns [4].
A number of articles also present properties of different traffic classes resulting from traffic classification. Gerber et al. [5] classified flow measurements from a tier-1 ISP backbone in 2003.

[1]This work was supported by SUNET, the Swedish University Network

Even if their classification method has been based on port numbers, they indicate a dominance of P2P applications. Sen et al. [6] investigated connectivity aspects of P2P traffic on different levels of aggregation (IP, prefix, AS) in 2002. The study was based on flow data collected at a single ISP, classified by a port number method. More recent articles from 2005 and 2006 present differences between P2P and non-P2P traffic in terms of flow properties such as size, duration and inter-arrival times [7, 8]. Perenyi et al. [8] additionally presents a comparison of diurnal patterns for P2P vs. non-P2P traffic.

This article presents the results of a classification of current Internet backbone data. The datasets do not include packet payloads, thus connection pattern heuristics [9] were used to classify the datasets. The classification approach, disregarding packet payload data, has the advantage of avoiding legal issues and has the capability to classify even encrypted traffic, which is gaining popularity among P2P traffic. We chose to focus on 3 main traffic classes: (1) P2P file sharing protocols; (2) Web traffic; (3) malicious and attack traffic. First, we show how these traffic classes develop over a time period of eight months by highlighting trends in traffic volumes and connection numbers, also pointing out some diurnal differences. Next, we present differences between the traffic classes in terms of connection signaling behavior. This includes success rates for TCP connection establishment, a breakdown of different TCP connection termination possibilities and TCP option usage within established connections.

To our knowledge, this is the first attempt to characterize differences and trends within traffic classes in terms of connection signaling, with exception of a brief discussion about connection termination in [10]. We provide a thorough analysis of differences and trends for the selected traffic classes, since they have a major impact on the overall traffic behavior on the Internet. It is of general importance to follow trends in contemporary Internet traffic in order to react accordingly in both infrastructure and protocol development. Furthermore a thorough analysis of specific connection properties reveals how different traffic classes are behaving 'in the wild'. Since the data analyzed was collected on a highly aggregated backbone during a substantial time period, the results reflect contemporary traffic behavior of one part of the Internet. These results are thereby not only valuable input for simulation models, they are also interesting for developers of network infrastructure, applications and protocols.

2 Data Description

The two datasets used in this article were collected in April (spring dataset) and in the time from September to November 2006 (fall dataset) on an OC192 backbone link of the Swedish University Network (SUNET). In spring, four traces of 20 minutes were collected each day at identical times (2AM, 10AM, 2PM, 8PM) as described in [11]. The fall dataset was collected at 276 randomized times during 80 days [12]. At each random time, a trace of 10 minutes duration was stored. To avoid bias when comparing the datasets, the 20 minute samples from spring were treated as two separate 10 minute traces. Furthermore, for this study traces from fall are only considered if collected during the time-window between 20 minutes prior and after the collection times of spring (e.g. 1:40AM-2:40AM).

When recording the packet level traces on the 2x10GB links, payload beyond transport layer was removed and IP addresses were anonymized due to privacy concerns. After further pre-

processing of the traces, as described in [12] and [11], a per-flow analysis was conducted on the resulting bi-directional traces. Flows are defined by the 5-tuple of source and destination IP, port numbers and transport protocol (TCP or UDP). TCP flows represent connections, and are therefore further separated by SYN, FIN and RST packets. For UDP flows, a flow timeout of 64 seconds was used [4]. The 146 traces in the spring dataset include 81 million TCP connections and 91 million UDP flows, carrying a total of 7.5 TB of data. The reduced fall dataset, consisting of 65 traces, includes 49 million TCP connections and 70 million UDP flows, carrying 5 TB of data. In both datasets, TCP connections are responsible for 96% of all data.

3 Methodology

The resulting 130 million TCP connections and 161 million UDP flows have been fed into a database, including per-flow information about packet numbers, data volumes, timing, TCP flags and TCP options. The flows have then been classified by use of a set of heuristics based on connection patterns. The classification method was introduced and verified on the April dataset, as described in [9]. The heuristics are intended to provide a relatively fast and simple method to classify traffic, which was shown to work well on traces even as short as 10 minutes. In the present study the flows are summarized into three different traffic classes: P2P (file-sharing); Web or HTTP (incl. HTTPS); Malicious and attack (i.e. scan, sweep and DoS attacks). Remaining traffic was binned in a fourth class, denoted 'others'. 'Others' includes mail, messenger, ftp, gaming, dns, ntp and remaining unclassified traffic. The latter accounts for about 1% of all connections. In this study, the focus is on trends and differences between P2P and Web traffic, with some notable observations from malicious traffic highlighted as well. Besides the traffic classification, an analysis of traffic volumes and signaling properties is carried out in two further dimensions: longitudinal trends between April and November and diurnal patterns between the four time clusters (times of day).

4 Trends in Traffic Volumes

Longitudinal trends in TCP traffic volumes have been analyzed by building time series for the three traffic classes within each of the four time clusters, representing times of day (2AM, 10AM, 2PM, 8PM). Due to space limitations, only a condensed time series of TCP traffic is illustrated in Fig.1. The x-axis of the graphs represent time, with one bar for each 10 minute long trace. The first row indicates an increase in traffic volume during 2006. While peak volume per 10 minutes lies at 70 GB in early April, volume reaches 85 GB in late April (right after Easter vacation). This trend continues, with peaks of 94 GB in September and finally 113 GB in November. During one specific interval on November 8 as much as 131 GB have been transfered via TCP. All peak intervals fall into the time cluster of 8PM. The second busiest time cluster in terms of traffic volumes is the one at 2PM. Transfer volumes during 2PM reach on average 80% of the peak values at 8PM. Nighttime and morning hours (2AM, 10AM) show the lowest activity with half the transfer volumes of the busy evening hours. This diurnal pattern is best visible in the April section of the first row in Fig.1.

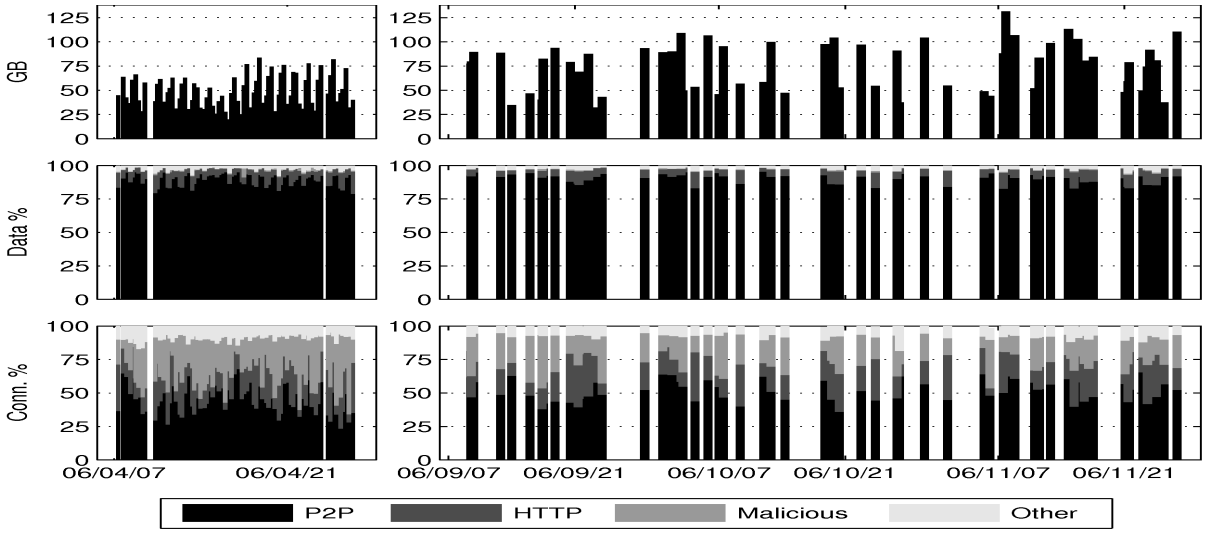


Figure 1: TCP data vs time (1st row); Appl. breakdown by data(2nd) and #conn.(3rd)

Even if there is an increase in data volumes of around 65% during a time period of eight months, the breakdown into traffic classes remains constant. P2P applications account constantly for as much as 93% and 91% of the data during evening and night time, respectively. During office hours (10AM, 2PM) the fraction of P2P data is reduced to 86%. HTTP, in contrast, is responsible for 9% of TCP data transfered during office hours, and drops down to 5% and 4% during evening and night time. This diurnal difference is explained by a network prefix analysis, yielding that most P2P traffic originates from student dormitories whereas Web traffic is commonly generated by Universities. The remaining data fractions account mainly for 'other' traffic, since malicious traffic and attacks tend to be single packet flows, not carrying substantial amounts of data.
The traffic breakdown in terms of connection numbers clearly shows that P2P connections typically carry higher amounts of data. Between 40% and 55% of the connections are classified as P2P, following the diurnal patterns of traffic volumes. HTTP connections account for 25% of all TCP connections during office hours, but drop down to 7% at night hours. Interestingly, the fractions of both P2P and HTTP connections (or connection attempts) increased slightly from April to November, while the fraction of malicious traffic decreased from around 30% to 20% during the same time. This development turns out to be a consequence of the constant nature of malicious traffic, such as scanning attacks. In absolute numbers, this traffic class remained remarkably constant during the eight months. Due to the increase in overall traffic volume, its relative fraction evidently was decreased. Since malicious or attack traffic shows neither longitudinal trends nor any significant diurnal pattern, we conclude that this type of traffic rather forms a constant 'background noise' in the Internet.

A similar analysis was also done for UDP flows. Even though larger in number, they are only responsible for 4% of all data. UDP data volumes during 10 minutes increased from peak values of 2.8 GB in April up to 4.6 GB in November. As in the case of TCP, peak intervals fall into the 8PM time cluster. Afternoon hours experience moderate UDP data volumes, and little UDP activity takes place during night and morning hours.

P2P flows over UDP carry in 76% of all cases less than three packets, which can be explained by signaling traffic as commonly used in P2P overlay networks such as Kademlia. In April, P2P flows are responsible for around 80% of UDP data volumes and connection counts, while the fraction has increased to about 84% in November. In absolute numbers, UDP P2P flow counts have even doubled from April until November, which shows that P2P applications deploying overlay networks via UDP are gaining popularity. Other traffic, including traditional UDP services like NTP or DNS, accounts on average for only 8% of the UDP flows. As for TCP, malicious traffic remains very constant in absolute numbers, which means that relative fractions decreased from 12% to around 8% in November.

5 Differences between Traffic Classes

The following subsection highlights differences between P2P, Web and malicious connections in terms of establishment and termination behavior. In the next subsection, TCP option deployment for P2P and Web connections is compared.

5.1 Differences in Connection Behavior

Fig.2 breaks down the success-rates of connection attempts for the three classes. Established connections include TCP flows with successfully carried out 3-way-handshakes. The second group of connection attempts did not fulfill 3-way-handshakes, but included an initial SYN packet. Finally, there are flows with no SYN seen. These are TCP sessions starting before the measurement interval. Such session fragments account for 13.5% of the 130 million connections seen. Malicious traffic usually consists of 1-packet flows only, which explains why only few malicious connection attempts fall into the no SYN category. In the further analysis, we will only focus on connections including initial SYN packets.
A notable trend can be observed in the P2P graph in Fig.2, where the fraction of unsuccessful connection attempts increased from an average of 49% in April to 54% in November. Web traffic on the other hand has significantly larger fractions of established connections, leaving only an average of 16.3% non-established. Malicious traffic is more likely to be established in the fall data, even though a majority of the malicious connections are still connection attempts. The

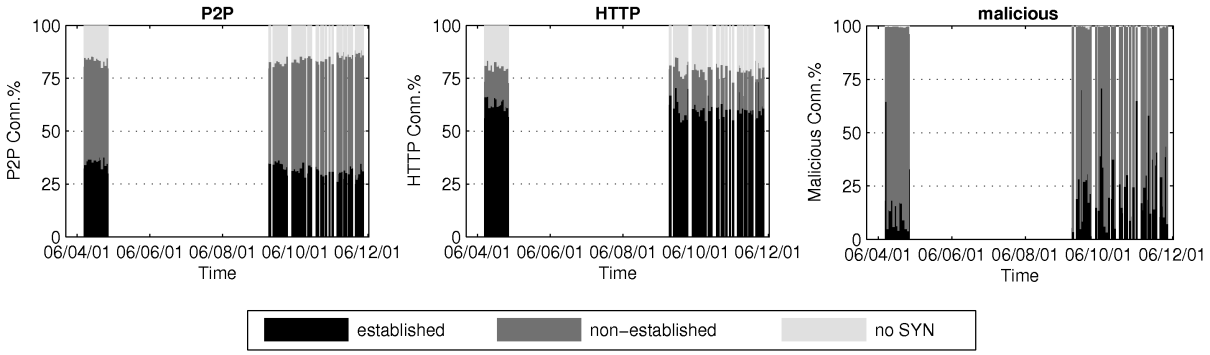


Figure 2: TCP Connection Breakdown

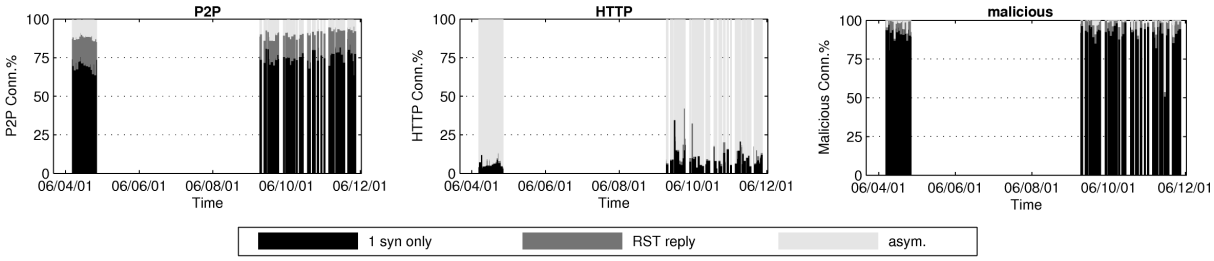


Figure 3: Breakdown of non-established TCP connections

increase in established attack connections is caused by an increase in login attempts to MS-SQL and SSH servers, with a few MS-SQL servers at a local University responsible for the majority of the attempts. According to SANS Internet Storm Center (ISC), malicious activities on both SSH (22) and MS-SQL (1433) ports increased significantly during 2006, which explains the trends seen here.
P2P and malicious connections reveal no diurnal patterns. Within Web traffic however, unsuccessful connection attempts account constantly for around 17.5% during all day, with exception of a drop to 10% during night time hours (2AM). We have no explanation for this phenomena other than HTTP connections are very rare in absolute number during night hours, which makes the statistical analysis more sensitive to behavior of individual applications or user groups.

Non-established connections: Non-established TCP connections have been further divided into connection attempts with one SYN packet only, attempts with direct RST reply, and asymmetrical traffic (Fig.3). Due to transit traffic and hot-potato routing, 13% of the connections are asymmetrically routed. It is not possible to observe a three-way handshake in these cases.

None of the traffic classes exhibits any significant diurnal pattern for non-established TCP connections. However, Fig.3 clearly highlights major differences between all three traffic classes. The already small fraction of non-established Web traffic (16.3% of all traffic) is mainly explained by asymmetrical traffic, and real unsuccessful connection attempts are very rare. Malicious traffic consists to a large degree of single SYN packet flows only. Single SYN flows are also dominating non-established P2P connections. While such connection attempts accounted for 71% in April, their fraction increase to 79% in November. This trend is also responsible for the increase of non-established P2P connections observed in Fig.2. Even if the high number of unsuccessful connection attempts within P2P traffic has been observed earlier [10], it is interesting to note that there is a clear trend in the fractions of one-SYN connections within P2P flows. The fraction increased by 23% (from 35% to 43%) within a period of 8 months.

Established Connections: Finally, established connections are broken down according to their termination behavior in Fig.4. Besides the proper closing approaches with one FIN in each direction or only one RST packet, as prescribed in the TCP standard, two unspecified termination behaviors have been observed. Connections closed by FIN, followed by an additional RST packet have been seen in direction of the initial SYN (typically the client) and the response (server). Finally, a number of connections were not closed during the measurement interval.

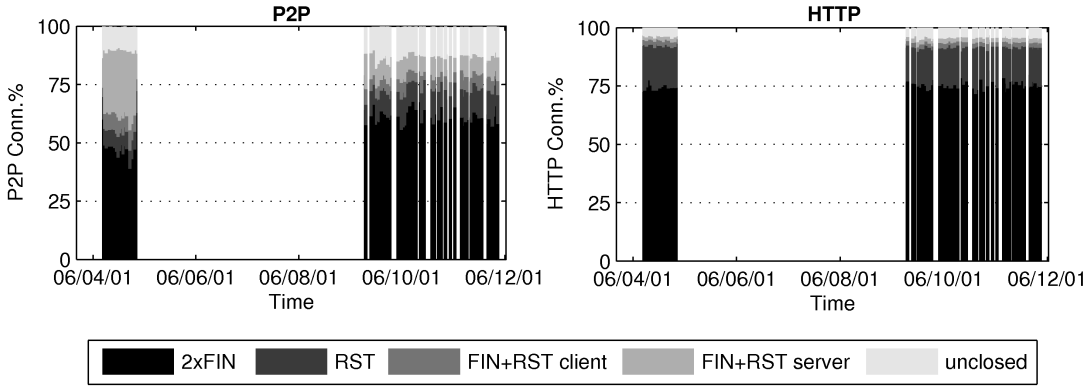


Figure 4: Breakdown of established TCP connections

The larger fraction of unclosed P2P connections is explained by the longer duration of P2P flows compared to Web traffic, as observed by Mori [7].

As for non-established connections, termination of Web connections neither shows significant trends nor diurnal patterns. HTTP connections are closed properly in 75% of all cases. Another 15% are closed by RST packets, mainly due to irregular web-server and browser implementations as noted by Arlitt [13]. FIN+RST behavior as well as unclosed connections (which corresponds to longer flows) are uncommon within Web traffic.

Even if there are no diurnal pattern observable, Fig. 4 indicates a significant change in termination behavior of P2P connections from spring to fall 2006. In April, only slightly less than half of the P2P connections have been closed properly with two FINs. As much as 20% of established P2P connections have been terminated with FIN plus an additional RST packet send by the server (or responding peer). A couple of popular hosts inside a student network have been identified as main source of this behavior. A commented text in the source code of a popular P2P client indicates that connections are closed with RST deliberately to avoid the TCP TIME_WAIT state in order to save CPU and memory overhead. In fall however, the fraction of FIN+RST terminations by the responder was reduced to around 8%, compensated by an increase in both valid TCP terminations, 2xFIN and single RST. Due to missing payload data, it was not possible to differentiate between different P2P software and version numbers. We suspect, that either the developers of the P2P application fixed this non-standard behavior in updated versions of the software, or the misbehaving P2P software lost popularity and was replaced by better behaving software by the users during 2006. However, the breakdown in Fig.4 shows that P2P traffic is mainly responsible for the large number of RST packets seen in todays networks.

5.2 Differences in Option Deployment

Finally, deployment of the most popular TCP options during connection established has been investigated for P2P and Web traffic (Table 1). For each of the four most popular TCP options, three different possibilities are distinguished: established - the option usage was successfully negotiated in SYN and SYN/ACK packets; neglected - the option usage was proposed in the SYN, but not included in the SYN/ACK; and none - the option was not seen in the connection.

(a) TCP Options in P2P Conn.

	MSS	SACK	WS	TS
estab.	99.9%	91.0%	14.9%	8.8%
neglected	0.1%	6.5%	0.6%	1.0%
none	0.0%	2.5%	84.5%	90.2%

(b) TCP Options in HTTP Conn.

	MSS	SACK	WS	TS
estab.	99.6%	65.7%	16.0%	13.4%
neglected	0.4%	27.9%	4.3%	4.3%
none	0.0%	6.4%	79.7%	82.3%

Table 1: Differences in TCP Option Deployment

Option usage turned out to be remarkably constant, with neither longitudinal nor diurnal trends. However, it is surprising to find such notable differences in option usage between traffic classes, considering that protocol stacks in the operating system, and not applications, decide about option usage. The MSS option is almost fully deployed, which agrees with the fact that the MSS option is set by default in all common operating systems. The SACK permitted option, in fact also a default option, is commonly proposed by initiating hosts, but is in 28% of the Web connections neglected. Interestingly, this fraction is significantly smaller in the case of P2P traffic, with only 6.5% neglecting SACK support.

While Linux hosts have the Window Scale (WS) and Timestamp (TS) options enabled by default, Windows XP does not actively use the options, but replies with WS and TS when receiving SYN packets with the particular option. This policy is well reflected by P2P connections, where WS and TS are rarely neglected, but either established or not used at all. HTTP connections do not really reflect this assumption, with 4.3% of WS and TS requests neglected by servers. However, WS and TS are established more often within Web traffic.

We suspect that the usage of WS and TS options within P2P traffic somewhat reflects the proportions of Linux (WS and TS enabled by default) and Windows systems (WS and TS disabled actively, but responding to request) on the links measured. The differences in option deployment for Web traffic however stem from a differing communication nature. While Web traffic represents classical client server communication, with one dedicated server involved, P2P represents a loose network of regular user workstations. Web-servers, as a central element, can thereby influence the behavior of larger numbers of connections. This suspicion is further confirmed by the fact that a majority of the HTTP connections neglecting usage of SACK are directed to less than 100 web-servers, which consistently do not respond with SACK options. Such central elements do not exist in P2P overlay networks. Furthermore, web-servers are more likely to be customized or optimized due to their specific task, whereas user workstations usually keep default settings of the current operating system. Some active measurement samples taken in October 2007 proved that popular web-servers, like google, yahoo and thePirateBay, still neglect SACK, WS or TS options.

6 Summary and Conclusions

In order to study trends and differences within the main traffic classes on the Internet, aggregated backbone traffic has been collected during two campaigns in spring and fall 2006 [12]. The collected packet level data has then been summarized on flow level. The resulting connections have finally been classified into P2P, Web and malicious traffic, using a connection pattern classification method [9]. An analysis revealed that overall traffic volumes are increasing for both TCP and UDP traffic, with highest activities at evenings. On diurnal basis, P2P and HTTP traffic exhibit different peak times. P2P traffic was found to be clearly dominating with 90% of the transfer volumes, especially during evening and night times. In contrast, HTTP traffic has its main activities (9% of the data-volumes) during office hours. Similar diurnal patterns have been observed in terms of connection numbers, even if P2P connections are not as dominating as in the case of data volumes. This indicates that P2P connections typically carry more data than Web traffic. Malicious and attack traffic is responsible for a substantial part of all TCP connections and UDP flows, but plays a minor role in terms of data volumes since it typically consists of 1-packet flows only. It was interesting to observe that the fraction of malicious TCP and UDP flows remained constant in absolute numbers both on diurnal and longitudinal basis, even though traffic volumes generally increased. This shows that malicious traffic (e.g. scanning attacks) forms a constant background noise on the Internet.

In terms of connection signaling behavior, major differences between the three traffic classes have been highlighted. The number of unsuccessful P2P connection attempts, which already dominated the P2P connection breakdown in spring, was shown to have increased further until fall. We conclude, that the large fraction (43%) of 1-packet flows on one hand and the large average data amounts per P2P connection on the other hand indicate a pronounced 'elephants and mice phenomenon' (Pareto principle) [7] within P2P flow sizes. Regarding termination behavior, P2P connections exhibit a clear trend towards higher fractions of proper closings in fall. HTTP connections on the other hand appear to behave comparable well according to specification at all times.

Finally, also TCP option deployment was shown to differ significantly between P2P and Web traffic. While P2P traffic rather reflects an expected behavior considering the default setting in popular operating systems, HTTP shows artifacts of the traditional client server pattern, with some dedicated web-servers neglecting negotiation for certain TCP options. This is especially true for the SACK option. We conclude that even though SACK is deployed by almost all P2P hosts and web-clients, a number of web-servers still neglect its usage. It is unclear to us, however, for which reasons web-server software or administrators would choose not to take advantage of certain TCP features, like SACK.

In the presented study, differences between traffic classes have been found in all aspects discussed, even if not always expected. The results provide researchers, developers and practitioners with novel, detailed knowledge about trends and influences of different traffic classes in current Internet traffic. The data analyzed was collected on a highly aggregated backbone link during a substantial time period, thus reflecting contemporary traffic behavior on one part of the Internet. Besides the general need of the networking and network security community to understand the nature of network traffic, information about behavior differences as seen 'in

the wild' can be important when developing network applications, protocols or even network infrastructure. Furthermore, the results form valuable input for future simulation models.

References

[1] A. W. Moore and K. Papagiannaki, "Toward the Accurate Identification of Network Applications," in *PAM: Proceedings of the Passive and Active Network Measurement Conference*, 2005.

[2] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-network Identification of P2P Traffic Using Application Signatures," ser. WWW: Proceedings of the 13th Int. World Wide Web Conference, 2004.

[3] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic Classification Through Simple Statistical Fingerprinting," *Computer Communication Review*, vol. 37, no. 1, 2007.

[4] T. Karagiannis, A. Broido, M. Faloutsos, and k. claffy, "Transport layer identification of p2p traffic," in *IMC: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, Taormina, Sicily, Italy, 2004.

[5] A. Gerber, J. Houle, H. Nguyen, M. Roughan, and S. Sen, "P2P The Gorilla in the Cable," in *National Cable and Telecommunications Association*, 2003.

[6] S. Sen and W. Jia, "Analyzing Peer-to-peer Traffic Across Large Networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, 2004.

[7] T. Mori, M. Uchida, and S. Goto, "Flow Analysis of Internet Traffic: World Wide Web versus Peer-to-Peer," *Systems and Computers in Japan*, vol. 36, no. 11, 2005.

[8] M. Perenyi, D. Trang Dinh, A. Gefferth, and S. Molnar, "Identification and Analysis of Peer-to-Peer Traffic," *Journal of Communications*, vol. 1, no. 7, 2006.

[9] W. John and S. Tafvelin, "Heuristics to Classifiy Internet Backbone Traffic based on Connection Patterns," in *ICOIN: Proceedings of the 22nd International Conference on Information Networking*, Busan, Korea, 2008.

[10] L. Plissonneau, J. L. Costeux, and P. Brown, "Analysis of Peer-to-Peer Traffic on ADSL," in *PAM: Proceedings of the 6th Passive and Active Network Measurement Workshop*. Boston, MA, USA: Springer-Verlag, 2005, pp. 69–82.

[11] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic and Header Anomalies Observed," in *IMC: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, San Diego, CA, USA, 2007.

[12] ——, "SUNET OC 192 Traces (collection)," http://imdc.datcat.org/collection/1-04L9-9=SUNET+OC+192+Traces (accessed 2008-02-05).

[13] M. Arlitt and C. Williamson, "An Analysis of TCP Reset Behaviour on the Internet," *Computer Communication Review*, vol. 35, no. 1, 2005.

PAPER VIII

Min Zhang, **Wolfgang John**, kc claffy and Nevil Brownlee

State of the Art in Traffic Classification: A Research Overview

PAM '09: 10th Passive and Active Measurement Conference, Student Workshop
Seoul, Korea, 2009

State of the Art in Traffic Classification: A Research Review

Min Zhang[1], Wolfgang John[2], kc claffy[3] and Nevil Brownlee[4]

[1]Beijing Jiaotong University, China. email: mia.minzhang@gmail.com

[2]Chalmers University, Sweden. email: wolfgang.john@chalmers.se

[3]CAIDA, UCSD. email: kc@caida.org

[4]CAIDA, UCSD. email: nevil@caida.org

1 Introduction

The Internet, while emerging as the key component for all sorts of communication, is far from well-understood. The goal of traffic classification is to understand the type of traffic carried on the Internet, which continually evolves in scope and complexity. For security and privacy reasons, many applications have emerged that utilize obfuscation techniques such as random ports, encrypted data transmission, or proprietary communication protocols. Further, applications adapt rapidly in the face of attempts to detect certain types of traffic, creating a challenge for traffic classification schemes. Research papers on Internet traffic classification try to classify whatever traffic samples a researcher can find, with no systematic integration of results. With the exception of machine learning techniques for traffic classification[1], we know of no complete overview of traffic classification attempts. To fill this gap, we have created a structured taxonomy of traffic classification papers and their datasets. To illustrate its utility, we use the taxonomy to answer the recently most popular question about traffic (*"How much is peer-to-peer file sharing?"*). Our survey also reveals open issues and challenges in traffic classification.

2 Research Review

Our review is based on 64 papers published between 1994 and 2008, starting with papers from top-ranked, peer-reviewed academic research conferences, and then including papers cited from this seeding set of papers, as well as follow-up papers written by the same authors.

We use the phrase *traffic classification* to refer to **methods** of classifying traffic **data sets** based on **features** passively observed in the traffic, according to specific **classification goals**. On a supplementary web page [2], we group papers into five categories: *survey, analysis, methodology, tools* and *others*. Analysis papers seek trustworthy numbers on traffic composition, while methodology papers focus on the methods of classification. We also provide a flexible, interactive table that supports selection of relevant attributes of papers, e.g., data sets, methods, goals, and main findings.

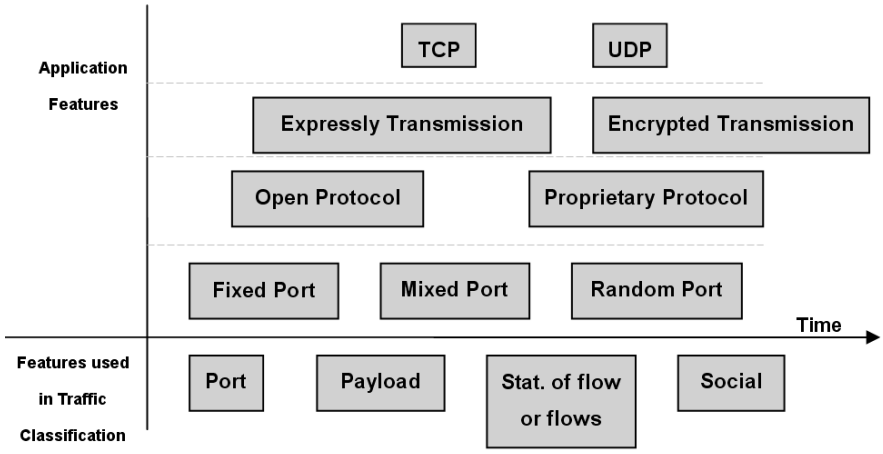


Figure 1: Trends of applications and features

2.1 Data Sets

Several public and private passive measurement infrastructures have provided a variety of data sets for Internet traffic classification studies. Based on our analysis, we find that these 64 papers make use of more than 80 data sets, which we classify based on *time of collection, link type, capture environments, geographic location, and payload length*.

2.2 Classification Goals and Features

Although traffic classification is a rather specific research field, the goals of these research papers are not identical. Some only have *coarse classification* goals, i.e., whether it's transaction-oriented, bulk-transfer, or peer-to-peer file sharing. Some have a *finer-grained classification* goal, i.e., the exact application generating the traffic.

Selection of traffic features used for classification evolves with application development. Media-rich entertainment applications - and associated attempts to discriminate against such applications - have inspired sophisticated obfuscation methods. Fig.1 gives a rough view of application and classification features. Recently, some applications (uTorrent, PPStream, PPLive) have changed from using TCP to UDP, a dramatic challenge for traditional traffic engineering.

Fifteen years ago, researchers could reasonably accurately classify traffic using TCP or UDP *port numbers*, but as applications began to use unpredictable ports, accurate classification requires *payload* examination. Examining payload is a controversial methodology due to privacy concerns, and is not even possible for encrypted payload, so researchers have studied techniques that are independent of packet content, such as *statistical features* based on network flows or underlying *social networks* to identify per-host behavior.

2.3 Methods

Methods to classify traffic at an application level include *exact matching*, e.g., of port number and payload; *heuristic methods*, applied e.g. on connection patterns to infer social networks; or

machine learning based on statistical features. We group machine learning methods into two categories: *Supervised Learning* and *Unsupervised Learning*. Naive Bayes, Decision Tree, NN, LDA, QDA, Bayesian Neural network are supervised learning algorithms; EM, AutoClass and K-Means are unsupervised learning algorithms [1].

3 Survey Analysis: How much P2P?

P2P traffic is one of the most challenging traffic types to classify. This is the result of substantial legal interest in identifying it and even more substantial negative repercussions to the user if P2P traffic is accurately identified. The misaligned incentives between those who want to use and those who want to identify P2P applications, together with the tremendous legal and privacy constraints against traffic research, render scientific study of this question near impossible. Even if possible, wide variation across links would prevent a simple numeric answer to the question of how much P2P traffic there is on the Internet.

Nonetheless, our taxonomy does reveal insights: the fraction of peer-to-peer file sharing traffic observed ranges from 1.2% to 93% across the 18 (out of 64) papers that provide such numbers. We also know that the average fractions reported have increased considerably from 2002 to 2006 (Table 1). Tables 2 and 3 show that results also vary widely by link and geographic location. Table 3 suggests that P2P is more popular in Europe, probably due to stricter policies (MPAA and RIAA) in North America. Note that the Asian results are from Japanese data sets, in which 1.34% and 1.29% are based on port numbers and therefore likely to significantly underestimate the fraction of P2P traffic. Furthermore, the amount of P2P traffic also varies by time of day, with higher fractions at night [3, 4].

One study[3] suggests that peer-to-peer applications are used more often at home than in the office. Finally, a study[4] in Europe found a higher fraction of P2P traffic on an European university link than some Canadian academics[3] found on their campus. Many of these numbers are based on statistical or host-behvioral classification, not the most reliable methods of detecting applications. More accurate methods involve examination of traffic contents (if unencrypted), which is fraught with legal and privacy issues.

Our taxonomy can allow similar analyses of other open questions, such as trends and development of traffic classes or features, yielding new insights into Internet traffic.

Table 1: P2P Range (Year)

Year	Range of P2P Volume	Paper
2002	21.5%	[5]
2004	9.19-60%	[6],[7],[8],[9],[10]
2006	35.1-93%	[11],[3],[12],[4]

Table 2: P2P Range (Link Location)

Year	Link Location	Range of P2P Volume	Paper
2004	Campus link	31.3%	[8]
2004	ADSL link	60%	[10]
2004	Backbone link	9-14%	[6],[9]
		17-25%	[7]

Table 3: P2P Range (Geographic Location)

Geo Location	Year	Range of P2P Volume	Paper
Europe	2005	60-80%	[13]
	2006	79-93%	[14],[4]
North America	2003	8%,10.7%	[6]
	2004	14%, 9.9%	[6]
	2003-04	9.2-70%	[7],[9],[15]
	2006	21-35%	[11],[3],[12]
Asia	2002	21.5%	[5]
	2005	1.34% (port-based)	[16]
	2008	1.29% (port-based)	[16]

4 Discussion

This research review, including 64 papers and more than 80 data sets, shows that traffic classification methods have evolved in response to the more sophisticated obfuscation techniques of network applications. We present a rough taxonomy of traffic classification approaches, based on features, methods, goals and data sets.

Our survey review also reveals shortcomings with current traffic classification efforts. First of all, the variety of data sets used does not allow systematic comparison of methods. Few research groups (can) share their datasets. Already true ten years ago, the field of traffic classification research still needs publicly available, modern data sets as reference data for validating approaches. This need however requires clear policies for data sharing, including accepted anonymization and desensitization guidelines. Secondly, the lack of standardized measures and classification goals is further amplifying the poor comparability of results. For example, there exists no clear definition for traffic classes such as P2P or file-sharing.

Despite these shortcomings, we showed how the taxonomy can shed insight on questions such as: *"how much of modern Internet traffic is P2P?"* Though we found some trends and indications, we have far too little data available to make conclusive claims beyond *"there is a wide range of P2P traffic on Internet links; see your specific link of interest and classification technique you trust for more details."*

References

[1] T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Communications Surveys*, 2008.

[2] CAIDA, "An Overview of Traffic Classification," 2009, http://www.caida.org/research/traffic-analysis/classification-overview/ (accessed 2009-02-17).

[3] J. Erman, A. Manhanti, M. Arlitt, I. Cohen, and C. Williamson, "Offline/realtime Traffic Classification Using Semi-supervised Learning," *Performance Evaluation*, vol. 64, no. 9–12.

[4] W. John, S. Tafvelin, and T. Olovsson, "Trends and Differences in Connections Behavior within Classes of Internet Backbone Traffic," *PAM*, 2008.

[5] M. Perenyi, T. Dang, A. Gefferth, and S. Monlnar, "Flow Analysis of Internet Traffic: World Wide Web versus Peer-to-Peer," *System and Computers in Janpan*, 2005.

[6] T. Karagiannis, A. Broido, N. Brownlee, kc claffy, and M. Faloutsos, "Is P2P Dying or Just Hiding?" *GLOBECOM*, 2004.

[7] T. Karagiannis, A. Broido, M. Faloutsos, and kc claffy, "Transport Layer Identification of P2P Traffic," *SIGCOMM*, 2004.

[8] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC Multilevel Traffic Classification in the Dark," *SIGCOMM*, 2005.

[9] M. Iliofotou, P. Pappu, and M. Faloutsos, "Graption: Automated Detection of P2P Applications Using Traffic Dispersion Graphs," *Technical Report, UC Riverside*, 2008.

[10] L. Plissonneau, J. Costeux, and P. Brown, "Analysis of Peer-to-Peer Traffic on ADSL," *PAM*, 2005.

[11] J. Erman, M. Arlitt, and A. Mahanti, "Traffic Classification Using Clustering Algorithms," *SIGCOMM*, 2006.

[12] J. Erman, A. Manhanti, M. Arlitt, I. Cohen, and C. Williamson, "Identifying and Discrimination between Web and Peer-to-Peer Traffic in the Network Core," *WWW*, 2007.

[13] M. Perenyi, T. Dang, A. Gefferth, and S. Monlnar, "Identification and Analysis of Peer-to-Peer Traffic," *Journal of Communications*, 2006.

[14] W. John and S. Tafvelin, "Heuristics to Classify Internet Backbone Traffic based on Connection Patterns," *ICOIN*, 2008.

[15] A. Madhukar and C. Williamson, "A Longitudinal Study of P2P Traffic Classification," *MASCOTS*, 2006.

[16] K. Cho, K. Fukuda, H. Esaki, and A. Kato, "Observing Slow Crustal Movement in Residential User Traffic," *CoNEXT*, 2008.

PAPER IX

Wolfgang John and Sven Tafvelin

Heuristics to Classify Internet Backbone Traffic based on Connection Patterns

ICOIN '08: Proceedings of the 22nd International Conference on Information Networking
Busan, Korea, 2008

Heuristics to Classify Internet Backbone Traffic based on Connection Patterns

Wolfgang John and Sven Tafvelin

Department of Computer Science and Engineering
Chalmers University of Technology, Göteborg, Sweden
{firstname.lastname}@chalmers.se

Abstract

In this paper Internet backbone traffic is classified on transport layer according to network applications. Classification is done by a set of heuristics inspired by two previous articles and refined in order to better reflect a rough and highly aggregated backbone environment. Obvious misclassified flows by the existing two approaches are revealed and updated heuristics are presented, excluding the revealed false positives, but including missed P2P streams. The proposed set of heuristics is intended to provide researchers and network operators with a relatively simple and fast method to get insight into the type of data carried by their links. A complete application classification can be provided even for short 'snapshot' traces, including identification of attack and malicious traffic. The usefulness of the heuristics is finally shown on a large dataset of backbone traffic, where in the best case only 0.2% of the data is left unclassified.

1 Introduction

Reliable classification of Internet traffic based on network applications is still an open research issue. However, network operators need to know the type of traffic they are carrying, amongst others in order to improve network design and provisioning and to support QoS and security monitoring. Ongoing measurements will furthermore reveal trends and changes in the usage of network applications. A good example is the shift in the early 2000's, when P2P file sharing replaced HTTP as the Internet's 'killer application', implying not only changes in data volumes, but also in traffic properties.

Different approaches to classify network traffic exist. Traditionally, traffic was classified based on *source and destination port numbers*. While this approach is very simple and does not require any packet payload, it is highly unreliable in modern networks. This is especially true for most P2P applications, which are trying to disguise their traffic in order to evade traffic filters and legal implications. It was shown that pure port number analysis underestimates actual P2P traffic volumes by factors of 2 to 3 [1].

A more reliable technique involves analysis of *packet payloads*. This approach can potentially provide highly accurate results given a complete set of payload signatures [2]. Beside the high effort of keeping the set of signatures updated, this method relies on network traces including packet data, which is uncommon due to privacy and legal concerns. Furthermore matching payload signatures on high-speed links is not trivial and poses high processing requirements.

A more recent classification technique is based on *statistical properties* of flows. A promising feature of these methods is that they are neither relying on port numbers nor on packet payload. However, the success of such 'statistical fingerprints' highly depends on the accuracy of the training data used. Ensuring accuracy and authenticity of the training sets is still an open issue [3], especially for disguised P2P flows.

Finally, network data can be classified according to *connection patterns*. Instead of looking at individual packets or flows, sequences of flows to or from a specific endpoint are matched with a set of predefined heuristics [4, 5]. These heuristics typically don't require packet payload and could potentially even disregard port numbers.

We initially intended to classify Internet backbone data in order to investigate the influence of P2P applications on traffic properties. Consequently it was planned to apply an existing and verified classification technique. Since our available datasets did not include packet payload and accurate training data, payload signatures or statistical fingerprinting could not be applied. Thus applying straight-forward connection pattern heuristics was the obvious approach. Karagiannis et al. [4] presented a set of two heuristics for transport layer identification of P2P traffic, including seven rules for removing false positives. The paper verifies that their method can identify 95% of P2P flows, with around 10% false positives compared to a carefully carried out payload analysis on OC-48 backbone data. Additionally, Perenyi et al. [5] proposed an updated set of six heuristics to identify and analyze P2P traffic, based on very similar ideas like Karagiannis. These heuristics were verified against traffic generated in a lab environment, yielding a hit ratio for P2P traffic of over 99%, with less than 1% false positives or unclassified P2P flows.

After applying the approaches of both Karagiannis and Perenyi to our data, it turned out that their results differ substantially. Furthermore, obvious false positives were detected in our data with both classification methods. As a result, we propose a refined combination of the heuristics by Karagiannis and Perenyi including some additions. The modifications were necessary to make the classification suitable for relatively short traces of a harsh Internet backbone environment, including highly aggregated and diverse traffic with a substantial amount of attacking and malicious traffic. Besides being based on the verified heuristics of Karagiannis and Perenyi, the results where further verified by manual inspection. Flows, which are not classified as P2P traffic by all three applied sets of heuristics are separately discussed regarding their most probable traffic class, thereby identifying obvious misclassification.

2 Data Description

Our dataset was collected during 20 days in April 2006 on the OC192 backbone of the Swedish University Network (SUNET). During this period, four traces of 20 minutes were collected each day at identical times (2AM, 10AM, 2PM, 8PM), as described in [6] and [7]. After recording the packet level traces on the 2x10 Gbit/s links, payload beyond transport layer was removed and IP

addresses were anonymized due to privacy concerns. A per-flow analysis was conducted on the resulting bidirectional traces, where flows are defined by the 5-tuple of source and destination IP and port numbers as well as transport protocol. TCP flows represent connections, and are therefore further separated by SYN, FIN and RST packets. UDP flows are separated by a timeout of 64 seconds. The 73 traces in the dataset sum up to 10.7 billion packets, containing 7.5 TB of data. We identified 81 Million TCP connections and 91 Million UDP flows, with the TCP connections carrying 97% of all data. The further analysis deals only with TCP connections, even though the classification heuristics have been successfully applied to UDP flows as well.

3 Proposed Heuristics

The set of heuristics proposed in this paper is strongly inspired by the heuristics by Karagiannis [4] and Perenyi [5], and will therefore be presented briefly only. The classification is based on connection patterns, but in some cases also port numbers are taken into account. Besides the rules for filtering out P2P traffic (H1-H5), a number of heuristics are used to remove false positives from flows suspected to be P2P traffic (F1-F10). These 'false positive' rules in turn can be used to classify other types of traffic, as shown in section 5. In contrast to Perenyi's approach, most of our proposed heuristics (with exception of H5 and F10) are first applied independently to all flows and are then prioritized. We apply these heuristics to our dataset in 10 minute intervals, which means that every interval is analyzed self-contained, without memory of previous intervals. Even though such memory could improve the accuracy of the results, our approach has the advantage to allow operators to classify snapshots of their traffic fast and in an ad hoc fashion. We will show that even 10 minute intervals can provide satisfying results. The proposed heuristics include a number of thresholds which might be adjusted. For our data the thresholds used were derived empirically through experiments on a number of traces. In the following list of heuristics, *(K)* (Karagiannis) or *(P)* (Perenyi) indicate by which previous method the heuristic was inspired, while *(J)* (John) marks newly introduced rules.

H1: TCP/UDP IP Pairs:(K),(P). This rule exploits the fact that many P2P applications use TCP for data transfer and UDP for signaling traffic. Source and destination IP pairs, which concurrently use TCP and UDP are therefore marked as P2P hosts. All flows to and from these hosts are marked as potential P2P flows. Concurrent here means usage of TCP and UDP within the 10 minutes interval. Karagiannis identified some non-P2P applications which show a similar behavior, such as netbios, dns, ntp and irc (Table 3 in [4]). UDP flows from these applications are excluded from this heuristic based on their port-numbers.

H2: P2P Ports:(P). Even though many P2P applications choose arbitrary ports for their communication, approx. one third of all P2P traffic can still be identified by known P2P destination port numbers [1]. Furthermore, it seems disadvantageous for non-P2P applications to deliberately use well known P2P ports for their services, since traffic on these ports is often blocked by traffic filters in some networks. Flows to and from port numbers listed in Table 3 of [5], enriched with additional P2P ports, are marked as potential P2P traffic.

H3: Port Usage:(P). In normal application, the operating system assigns ephemeral port numbers to source ports when initiating connections. These numbers are often iterating through a configured ephemeral port space. It is very unusual, that the same port numbers are used within short time periods. This however can be the case for P2P applications with fixed ports assigned for signaling traffic or data transfer. If a source port on a host is repeatedly used within 60 seconds, the host is marked as P2P host, and all flows to and from this host are marked as potential P2P flows.

H4: P2P IP/Port Pairs:(K). If listening ports on peers in P2P networks are not well known in advance, they are typically propagated to other peers by some kind of signaling traffic (e.g. an overlay network). This means that each host connecting to such a peer will connect to this agreed port number, using a random, ephemeral source port. As noted by Karagiannis, P2P peers usually maintain only one connection to other peers, which means that each endpoint (IP,port) has at least the same number of distinct IP addresses (#sIP) and number of distinct ports (#sPort) connected to it. If #sPort-#sIP$<$2 and #sIP$>$5, the host is considered as P2P host, and all flows to and from this host are marked as potential P2P.

F1: Web IP/Port Pairs:(K). Web traffic on the other hand typically uses multiple connections to one server. For this reason hosts are marked as web-hosts, if the difference between #sPort and #sIP connected to an endpoint (IP,port) is larger than 10, the ratio between #sPort and #sIP is larger than two and at least 10 different IPs are connected to this endpoint (#sPort-#sIP$>$10 and #sPort/#sIP$>$2 and #sIP$>$10). All flows with http port numbers (80, 443, 8080) to and from these webhosts are then marked as web traffic.

F2: Web:(P). To further identify web traffic, we follow Perenyi's heuristic number 2, taking advantage of the fact that web clients typically not only use multiple, but even parallel connections to webservers. Hosts with parallel connections to a http port are considered as webservers. All flows to and from web servers on http ports are marked as web traffic.

F3: DNS:(K). Traditional services like dns sometimes use equal source port and destination port numbers. As suggested by Kargiannis, we mark endpoints (IP,port) as non-P2P, if it includes flows with equal source- and destination port and port numbers smaller than 501. All flows to and from this endpoint are then marked as non-P2P traffic.

F4: Mail:(K). Hosts receiving traffic on mail ports (smtp, pop, imap) and in the same analysis interval also initiate connections to port 25 on other hosts are considered to be mailservers. All flows to and from mailservers are marked as mail traffic.

F5: Messenger:(K). Popular messenger and chat servers (icq, yahoo, msn, jabber, irc) tend to have long uptimes and rarely change IP addresses, especially when maintained by commercial providers such as Microsoft and Yahoo. To improve the accuracy of the results, in this heuristic we therefore take advantage of the whole 20 day long dataset. Hosts, connected to by at least 10 different IPs on well known messenger ports within a period of at least 10 days, are marked as messenger servers. All traffic to and from these hosts on known messenger ports is classified as messenger traffic.

F6: Gaming:(J). Popular game servers (currently only the most common online games Half-Life and World of Warcraft) are identified in the same fashion as messenger servers. All traffic to and from the game servers on well known gaming ports is classified as gaming traffic.

F7: Ftp: (J). Ftp was not taken into account by Karagiannis, while Perenyi implicitly included it as part of its 'well known port' rule. Identifying data transfer in passive ftp remains a problem. Active ftp data transfer on the other hand can easily be marked as ftp traffic identified by an initiating sourceport number of 20, as used by ftp servers to actively serve their requesting clients.

F8: non P2P Ports:(P). As noted by Perenyi, destination ports are still suitable to identify traffic of some common applications. Our set of well known non-P2P ports includes netbios, dns, telnet, ssh, ftp, mail, rtp and bgp. All flows to the listed destination ports are marked as non-P2P flows.

F9: Attacks:(J). This rule is probably the most significant improvement to the original heuristics. While Perenyi does not take malicious traffic into account at all, Karagiannis rules out simple network scans as false positives. We first identify suspicious pairs of source IPs and destination Ports (*AttackPairs*). All flows with source IP and destination port inside the list of AttackPairs are then marked as attacks. AttackPairs are identified by three different cases:

a) *Sweep*: The ratio between number of destination IPs (#dIP) and number of destination ports (#dPort) from a certain host is greater than 30. This means that one host is connecting to a lot of hosts with only a few different port numbers, as typically the case when scanning IP ranges for vulnerabilities on specific ports.

b) *Scan*: The ratio between #dIP and #dPort is less than 0.33 and #dIP is less than 5. This would be the case if one host is scanning a small number of specific, dedicated targets on a large number of different ports.

c) *DoS*: #dIP is less than 5, #dPort is less than 5 and the average number of conn. per sec (conn/s) is greater than 6. This behavior represents 'hammering' attacks, where one host is trying to overload a few targets (typically one) by opening connections to a few services very frequently.

F10: unclassified, known non-P2P Port:(J). Up to this point all heuristics mark flows independent of each other. All flows left unmarked until now are neither suspected to be P2P traffic nor obvious cases of non-P2P traffic. We believe it is safe now to apply a port number classification on the previously unclassified flows. All flows, whose source- or destination port number matches a set of well-known non-P2P port numbers including (http, messenger, game) are classified non-P2P, if not classified by any heuristics (H1-H4, F1-F9).

H5: unclassified, long flow:(P) After removing well known applications from the unclassified flows, we mark remaining unclassified flows which carry more than 1 MB of data in one direction or have connection durations of over 10 minutes as P2P flows. This rule is based on Perenyis heuristic 6, even though we believe it is a very weak rule. However, there is a large probability, that such long flows in fact are P2P flows.

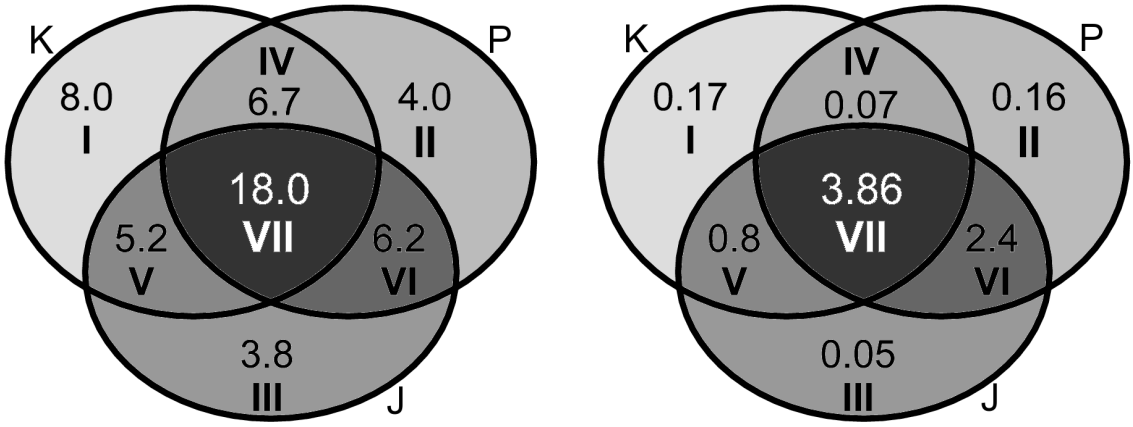
After running an analysis on our dataset based on the presented heuristics, we classify all flows as P2P traffic which have been classified by one or more of the heuristics H1-H5, and at the same time NOT being classified by any of the false positive heuristics F1-F10. In Section 4, flows marked by H5 are included to P2P traffic. However, in Section 5 we chose to treat traffic classified by this heuristic separately.

Weaknesses: The above suggested mixture of connection pattern and port number classification has some weaknesses. First of all, the analysis interval can greatly influence the success of the heuristics, especially for those analyzing connection patterns. Longer intervals yield better results given that the various empirical thresholds are adjusted. A natural border for the analysis interval is obviously given by memory and computational constraints. Additionally, there is a risk with too long intervals since activities on the Internet are often short lived, and e.g. a host doing a scanning campaign on port 80 might simply surf the Internet an hour later. Another problem in this context are networks behind NATs or with dynamically assigned IP addresses. A second weakness is the length of the traces used. For connections established before the measurement interval the initiator is unknown, and it is unclear which host is source and which is destination. Additionally there is typically some asymmetrically routed traffic in backbone networks, which needs to be considered as special case when implementing the heuristics. Furthermore, heuristics based on connection patterns are depending on a certain amount of connections per host during the analysis interval. Finally, heuristics relying on empirical thresholds are not fail-proof, and it is possible to come up with examples for false positives for any of them. However, both Karagiannis and Perenyi proved that these heuristics can be effective when carefully prioritizing the different rules.

4 Verification of the proposed Heuristics

To verify the proposed adjustments, we classified our backbone data by each of the three sets of heuristics (Karagiannis, Perenyi and our own proposal in section 3). For each flow, a bitmask was set in a database according to matching rules. This method allowed us to analyze intersections between the three approaches separately - meaning flows marked as P2P traffic by either one, two or all three of the approaches. The results are illustrated by the Venn diagrams in fig.1, presenting connection counts (a) and amount of data (b) in absolute numbers. The three circles represent P2P flows classified by the different rule-sets (Karagiannis left, Perenyi right, new proposal beneath). The following paragraphs will discuss the different intersections (IS I-VII), thereby motivating the proposed modifications and additions to the original approaches.

IS I: This intersection represents flows classified as P2P by Karagiannis only. A number of updated rules identified these flows as false positives. Rule F9 (attacks) marked 53% of them, often classified as known non-P2P ports by Perenyi. This is plausible, considering that these connections are mainly 1-packet flows, directed to popular scanning ports (135, 139, 445). Rule F2 (web) classified another 25% of these connections, carrying 40% of the data in this intersection. Since parallel connections to http ports are a strong indication for web traffic, F2 is regarded as a reliable rule. F8 (non P2P-ports) accounts for 15% of these connections, carrying 43% of the data, mainly on ports for rtp, ssh and mail. This is plausible, since it is common that



(a) #connections (10^6) (b) Amount of data (TB)

Figure 1: P2P traffic by Karagiannis (K), Perenyi (P) and new proposal (J)

these applications carry large amounts of data, so there is no reason considering them as P2P flows. The remaining flows are either marked by F7 (active ftp) or F10 (unclassified, but known non-P2P port).

IS II: In this intersection, 99% of the data was classified as P2P by Perenyi's 'long flow' rule only. This is obviously Perenyi's weakest heuristic, since it simply considers any flow carrying more than 1 MB of data or lasting longer than 10 minutes as P2P. 75% of this data is considered as false positive according to F10. Unclassified by any other heuristic, a pure port number classification marks these flows as web flows according to their destination http ports. Another 10% are marked as web traffic by F2. The remaining data was classified by F4 (mail), F5 (messenger) and F6 (gaming), all three considered to be accurate rules, taking connection patterns and port number into account. In terms of connection numbers, 95% of the connections in IS II are again identified as false positives by F9 (attacks) with similar properties as in IS I.

IS III: All of the flows only classified as P2P by the proposed heuristics are unclassified by Perenyi. Even Karagiannis left 45% unclassified, with the remaining 45% classified by the non-P2P IP/Port Pair rule. In [4] this rule was identified as unreliable if less than 5 IPs are connected to an IP/Port Pair. Since in H4 this restriction was taken into account, it is plausible to include the flows marked as P2P in IS III based on combinations of H4 and/or H3 (port usage).

IS IV: The flows classified as P2P by both Karagiannis and Perenyi are in 98% of the cases again marked as false positives by F9 (attacks), carrying very little data. In terms of data, Perenyi's 'long flow' rule and Karagiannis' IP/Port Pair rule are responsible for 90% of the data in this intersection. As discussed above, both rules are considered rather weak. Since additionally none of the refined P2P heuristics (H1-H4) matched, rule F10 (unclassified, but well known port) is reason enough to exclude 80% of this flows as false positives (mainly targeting http ports). The remaining flows have been marked by F1 (web pairs), F5 (messenger) and F6 (gaming).

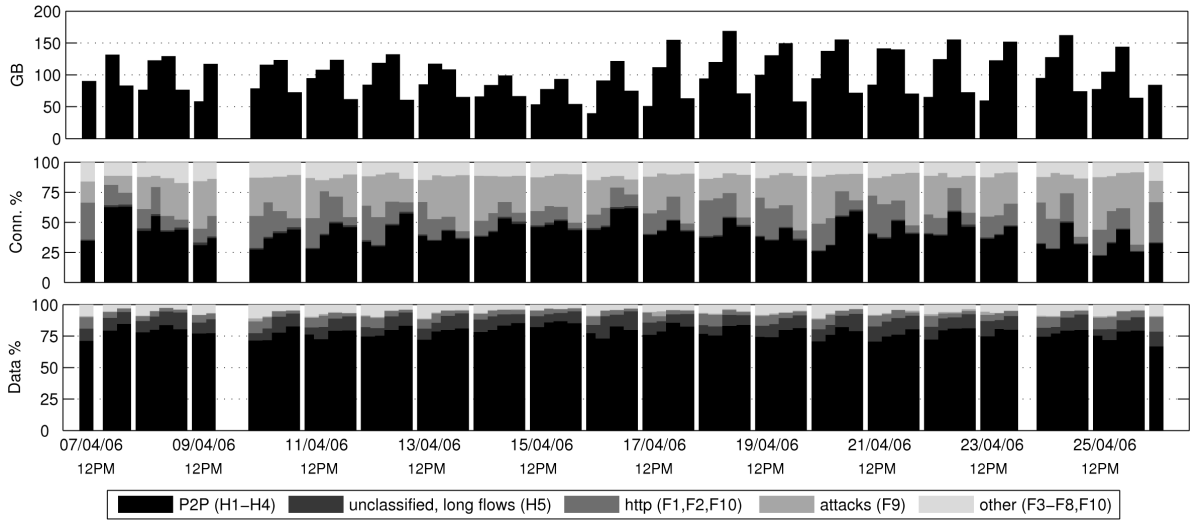


Figure 2: TCP data vs trace times (1st row); Appl. breakdown by #conn. (2nd); Appl. breakdown by data carried (3rd)

IS V: In this intersection, flows are entirely unclassified by Perenyi. Since these flows are classified as P2P by both Karagiannis and the proposed heuristics, there is no reason not to consider them as P2P traffic.

IS VI: Perenyi's 'long flow' rule identified 77% of the data in this large intersection as P2P, with the remaining connections classified according to known P2P port numbers. The proposed heuristics on the other hand classify 88% of these flows as P2P by H2-H4, accounting for 72% of data. Most of the data is even classified by 2 or 3 of the heuristics. The remainder (685 GB) is classified by H5 (long flows) only, and will therefore be treated as a special category in our results section. Karagiannis leaves a large part (60%) of this intersection unclassified, with the rest classified by the non-P2P IP/Port Pair rule, which is an inaccurate rule for endpoints with few connected hosts as noted above. Since there is no strong indication to rule out flows as false positives, they are classified as P2P except for the 685 GB by H5 (long flows).

IS VII: Data in this intersection is classified as P2P by both Karagiannis and Perenyi, and no false positives were identified by the proposed heuristics. Consequently, there is no reason not to consider this intersection as P2P.

5 Classification Results

We finally applied the proposed heuristics to our data traces (Section 2). Fig.2 represents time series of classified network protocols. The x-axis of the graphs represents time, with one bar for each trace time (2AM, 10AM, 2PM and 8PM). Four traces on three days (07/04, 09/04, 23/04) had to be discarded due to measurement errors. The remaining whitespaces between bars represent the 8 hour measurement break between 2AM and 10AM, which means that each continuous block represents 4 traces collected in the order of [10AM, 2PM, 8PM, 2AM]. The

first graph shows total amount of TCP data in GByte versus trace times. The second and third row illustrate application breakdown for the particular trace in terms of connection numbers and data volumes.

In the connection breakdown, only four categories are visible, since flows classified by H5 are too small in number to show up in this graph. Anyhow, these 31,000 long flows are responsible for almost 10% of the TCP data. Typically, these flows begin and end outside the measurement period and transfer data between hosts, which do not generate additional traffic on our links. Since our classification method is based on connection patterns, insufficient connection numbers for a particular host reveal a weakness of this method. In the data breakdown on the other hand, flows classified by F9 (attacks) are not visible. Even though attacks represent between 8 and 60% of the flows, they carry less than 1% of the data on average. This also proves the power of F9, since it effectively detects DoS attacks and network scanning, which typically show up as short 1-packet flows only, carrying no payload data. P2P flows (flows matching H1-H4, while not matching any of the false positive rules F1-F10) account for an average of 42% of the connections. On the other hand, they carry between 66 and 87% of the traffic, with an average of 79%. This indicates once more the success of the heuristics, since P2P flows are expected to carry more data on average than non-P2P flows. On this dataset, the proposed heuristics left as little as 1% of the connections and 0.2% of the data unclassified (except the flows classified by H5).

While a careful analysis of these results need to be done as future work, the short result section should indicate the power and usefulness of the proposed heuristics.

6 Summary and Conclusions

This article proposes a set of heuristics for classifying backbone-type data according to applications. The proposed heuristics are intended to provide researchers and network operators with a comparably simple[1] method to get insight into the type of data carried by their links. Furthermore these heuristics work on traces as short as 10 minutes, which allows operators to classify snapshots of their traffic relatively fast, by only adjusting applied thresholds and parameters empirically. The heuristics can be used to classify backbone traffic according to a number of applications, including P2P traffic, web traffic and other common applications. Furthermore, we introduce a new rule that successfully identifies network attacks, which is an additional feature for network operators and researchers interested in network security or intrusion detection issues. Some of the proposed heuristics are based on two existing methods. Besides relying on the verification methods of these original heuristics, a careful analysis of the resulting classifications was carried out, pinpointing obvious cases of false positives. Both previous sets of heuristics overestimate the number of P2P flows, mainly because attacking traffic is not taken into account accordingly. On the other hand, both methods underestimate the amount of P2P data on the links. By combining the successful rules of the two methods and adding new, necessary rules, a set of refined and updated heuristics is presented. The heuristics are successfully applied to a large collection of backbone data, yielding a valuable breakdown of applied appli-

[1] Simple, because it does not require packet payloads, updated payload signatures, and training data.

cation protocols. When considering the few large flows classified by the H5 rule as P2P traffic, the proposed heuristics leave only 0.2% of the data unclassified.

Acknowledgements

This work was supported by SUNET, the Swedish University Network.

References

[1] A. W. Moore and K. Papagiannaki, "Toward the Accurate Identification of Network Applications," in *PAM: Passive and Active Measurement Conference*, 2005.

[2] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-network Identification of P2P Traffic Using Application Signatures," ser. WWW: 13th International World Wide Web Conference, 2004.

[3] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic Classification Through Simple Statistical Fingerprinting," *SIGCOMM Comp. Comm. Rev., 37(1)*, 2007.

[4] T. Karagiannis, A. Broido, M. Faloutsos, and k. claffy, "Transport Layer Identification of P2P Traffic," in *IMC: Internet Measurement Conference*, Taormina, Sicily, Italy, 2004.

[5] M. Perenyi, D. Trang Dinh, A. Gefferth, and S. Molnar, "Identification and Analysis of Peer-to-peer Traffic," *Journal of Communications*, vol. 1, no. 7, 2006.

[6] W. John and S. Tafvelin, "Analysis of Internet Backbone Traffic and Header Anomalies Observed," in *IMC: Internet Measurement Conference*, San Diego, California, USA, 2007.

[7] ——, "SUNET OC 192 Traces, April 2006," http://imdc.datcat.org/collection/1-04HN-W=SUNET+OC+192+Traces\%2C+April+2006 (accessed 2007-12-07).

PAPER X

Erik Hjelmvik and **Wolfgang John**

Statistical Protocol IDentification with SPID: Preliminary Results

SNCNW '09: Swedish National Computer Networking Workshop

Uppsala, Sweden, 2009

Statistical Protocol IDentification with SPID: Preliminary Results

Erik Hjelmvik[1] and Wolfgang John[2]

[1]Independent Network Forensics and Security Researcher, Sweden
erik.hjelmvik@gmail.com

[2]Chalmers University of Technology, Sweden
wolfgang.john@chalmers.se

Abstract

Identifying application layer protocols within network sessions is important when assigning Quality of Service (QoS) priorities as well as when conducting network security monitoring. This paper introduces a Statistical Protocol IDentification algorithm (SPID) utilizing various statistical flow and application layer data features. We have identified application layer protocols by comparing probability vectors created from observed network traffic to probability vectors of known protocols. Promising preliminary results are presented, showing average precision of 100% and recall of 92% for a small set of protocols within traffic traces from an access network. To further improve the results, a number of ongoing and future directions with SPID are discussed, such as optimization of the attribute meters and improving robustness against different network environments.

1 Introduction

Today, there is an increasing need for reliable classification of network traffic according to application layer protocols. Traffic classification is required for operational purposes, including QoS and traffic shaping mechanisms, optimization of network design, and resource provisioning. Furthermore, understanding the type of traffic carried on networks facilitates detection of illicit traffic, such as network attacks and related security violations. Modern firewalls, NATs and IPSs need to be able to reliably identify network protocols in order to implement fine-grained and secure access policies. Besides the apparent interest of operators and researchers to understand trends and changes in network usage, there have been a number of political and legal discussions about Internet usage (e.g. RIAA vs PirateBay), which further amplifies the importance of accurate traffic classification methods.

Currently, there are roughly four approaches to classify network traffic according to application protocols [1]. Traditionally, traffic was classified with sufficient precision by simply looking at TCP/UDP *port numbers*. With the advent of P2P file sharing systems and their legal implication due to copyright concerns, more and more applications started to use unpredictable dynamic port ranges or reused well-known ports of other applications, which yields poor results for port classification methods on modern network data [2, 3].

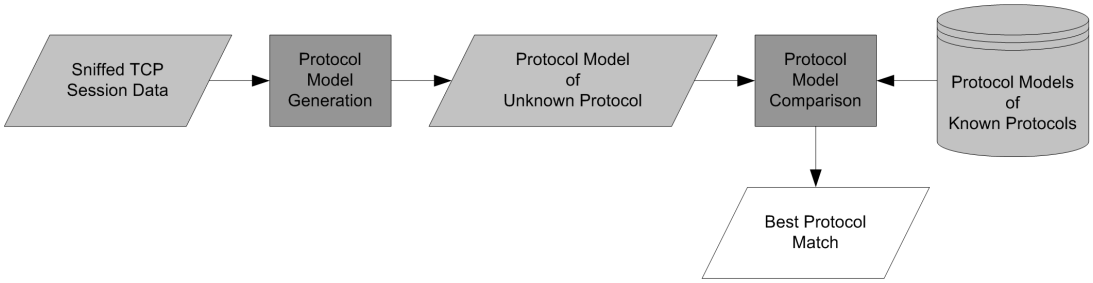


Figure 1: Protocol identification data flow

This development led to a wide use of *deep packet inspection* (DPI) for classification, which means inspection of packet payloads for known string patterns [4, 5]. DPI is currently the most reliable way to classify traffic, which explains its popularity in commercial tools. However, examination of (user) application layer data causes substantial legal and privacy concerns. Furthermore, DPI with static signatures is resource-expensive and does not work on encrypted traffic, which is becoming common as a reaction to legal threats.

As a result, the researchers began to work on classification techniques independent from payload inspection. One such approach is to classify traffic based on *social behavior* of hosts by looking at their connection patterns [6, 7, 8]. While some of these methods are quite successful, they are only able to classify traffic in rough categories, such as mail, web, and P2P traffic.

Another recent, payload independent approach is classi-fication based on *statistical flow properties* such as duration, packet order and size, inter-arrival times, etc. [9, 10].

In this paper we introduce SPID, the Statistical Protocol IDentification algorithm [11]. The SPID framework is built to perform protocol identification based on simple statistical measurements of various protocol attributes. These attributes can be defined by all sorts of packet and flow data, ranging from traditional statistical flow features to application level data measurements, such as byte frequencies and offsets for common byte-values. In this sense SPID is a hybrid technique, utilizing efficient generic attributes, which can include *deep packet inspection* elements by treating them in the same way as *statistical flow properties*. A proof-of-concept (PoC) application for the SPID algorithm is available at SourceForge[1].

2 SPID Design Goals

The main goal of the SPID algorithm is to reliably identify which application layer protocol is being used in a network communication session in an easy and efficient fashion. SPID should not only be able to classify traffic into rough, coarse-grained traffic classes (such as P2P or web), but in fine-grained classes on a per-protocol basis, which would enable detailed QoS assignments and security assessment of network flows.

Many application layer protocol identification schemes used today rely on signatures or patterns that usually occur in protocols, e.g. 'BitTorrent protocol', 'SSH-' or 'GET / HTTP/1.1'.

[1] http://sourceforge.net/projects/spid/

A problem with looking for such static patterns is that the fingerprints need to be manually created, which means that network traffic and protocol specifications need to be studied and abstracted in order to create a reliable identification pattern. However, creation of application layer signature patterns can be automated, as shown by Park et al. [12].

Several protocols use obfuscation and encryption in order to prevent identification through static pattern-based signatures. Protocols that utilize such obfuscation techniques include the Message Stream Encryption (MSE) protocol (applied e.g. by BitTorrent) and Skype's TCP protocol. Manually creating signatures for proprietary protocols lacking documentation - such as Skype, Spotify's streaming protocol and botnet command-and-control (C&C) protocols - can be very troublesome.

An important design goal of SPID is therefore to replace the use of pattern matching techniques with entropy based comparisons of probability distributions. Doing so eliminates the need for manually extracting inherent properties of protocols, since the SPID algorithm has the ability to automatically deduce properties from training data. The training data used, however, needs to be pre-classified, which can be done through manual classification by experts or by active approaches, as in Szabo et al.[13]. A further goal of SPID is to allow protocol models to be updated easily as new training data becomes available, without having access to the previously used training data.

The required manual efforts for adding a new protocol are thereby shifted from detailed protocol analysis to assembling training data for that particular protocol. This is an important change since manual creation of static protocol patterns is a time consuming task, and new protocols continuously appear. Many new protocols are furthermore proprietary and undocumented binary protocols, which require advanced reverse engineering in order to manually generate protocol patterns.

The SPID algorithm does not require support for pattern-matching techniques, such as regular expressions. By providing a generic XML based format to represent protocol model fingerprints, SPID is designed to be both platform and programming language independent.

Further operational key requirements for the algorithm are:

1. Small protocol database size
2. Low time complexity
3. Early identification of the protocol in a session
4. Reliable and accurate protocol identification

The motivation for requirements 1 and 2 are that it should be possible to run the SPID algorithm in real-time on an embedded network device with limited memory and processing capabilities. Requirement 3 should enable the use of the results from the SPID algorithm in a live traffic capturing environment in order to provide QoS to an active session in real-time, block illicit traffic or store related traffic for off-line analysis. An implicit goal is therefore that protocols should be identifiable based on the first few packets with application layer data.

Index	0	...	80 ('P')	81 ('Q')	82 ('R')	83 ('S')	84 ('T')	...	255
Counter vector	7689	...	1422	502	1001	1482	2644	...	3276
Probability vec.	0.026	...	0.004	0.002	0.003	0.005	0.008	...	0.011

Table 1: Example of an Attribute Fingerprint: Byte frequency for HTTP

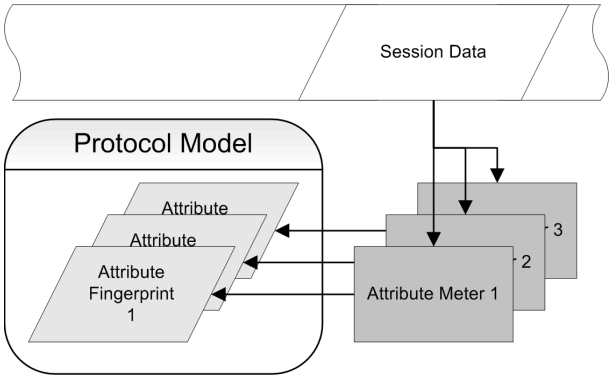


Figure 2: Generation of protocol models

3 Overview of the SPID framework

As illustrated in Fig. 1, SPID performs protocol identification by comparing the protocol model of an observed session to pre-calculated protocol models of known protocols.

3.1 Protocol Models

Protocol models contain a set of *attribute fingerprints* (Fig. 2). Fingerprints are created through frequency analysis of various attributes, such as application layer data or flow features, and are represented as probability distributions. The PoC SPID application uses over 30 *attribute meters*[2], which are the functions that provide the distribution measurements for each specific attribute. An example of such an attribute meter is the basic ByteFrequencyMeter, which measures the frequency with which all of the possible 256 bytes occur in the application layer data. Other attribute meters perform much more advanced analysis of various properties in a session, such as measuring the frequency of various request-response combinations (e.g. HTTP behavior, where a 'GET' request is followed by an 'HTTP' response or FTP behavior where a '220' message is replied to with a 'USER' command). The SPID algorithm also makes use of flow measurements (that do not require inspection of application layer data), such as packet sizes, packet inter-arrival times and packet order number- and direction combinations.

Attribute fingerprints are represented in the form of probability distributions. This means that the data for each fingerprint is represented by two arrays (vectors) of discrete bins: one array of counter bins and one of probability bins (Table 1). Values of the counter vectors represent the number of times an observation (analyzed packet) has caused the associated attribute meter to trigger that particular index number in the vector. Probability vectors are normalized versions

[2]http://spid.wiki.sourceforge.net/AttributeMeters

of the counter vectors, with all values in every probability vector summing up to 1.0. In this paper a vector length of 256 is used; an implementation of the SPID algorithm can, however, use any length for these vectors.

3.2 Generation of Protocol Models

For observed sessions, a protocol model is created upon session establishment (e.g. after the TCP three-way handshake), consisting of a set of attribute fingerprints. Every packet with application layer data belonging to a session is called an observation. Each such observation is then fed into the attribute meters, which provide measurements that are stored in the session's protocol model. Upon receiving such a measurement, the protocol model increments the fingerprint counters accordingly. For illustration, we assume an attribute fingerprint for the ByteFrequencyMeter from the first data packet observed in a HTTP session, i.e. a HTTP GET command. The counters would be incremented to

- 3 for the counter at index 84 (since there are three T's in 'GET / HTTP/ 1.1')
- 2 for counters at index 32, 47 and 49 (space, '/' and '1')
- 1 for counters at index 71, 69, 72, 80 and 46
- 0 for all other counters

All other attribute fingerprints belonging to the same protocol model will also increase their counters based on the sets of indices that are returned from their respective attribute meter. Subsequent packets in the same session will cause the fingerprint counter values to further increment. However, since one design goal of SPID is to keep time complexity low, we want to show in future work that utilizing only the first few packets provides sufficient precision.

Protocol models for known protocols are generated from real network packet traces. These traces need to be pre-classified, either manually or automatically [13], to be usable as training data for the SPID algorithm. The pre-classified training data is converted to protocol model objects (one per protocol) by generating protocol models for each session and merging (i.e. adding) the fingerprints of the same protocol and attribute type.

The more sessions are merged together for each protocol, the more reliable the fingerprint will be. As a rule of thumb, we found that 10% of the fingerprints' vector lengths (i.e. approximately 25) turned out to be a rough measurement of the minimum number of training sessions needed to build a reliable protocol model.

3.3 Comparison of Protocol Models

Fingerprints of an observed session are compared to fingerprints of known protocol models by calculating the Kullback-Leibler (K-L) divergence [14] (also known as relative entropy) between the probability distributions of the observed session and each protocol model, ranging from 0 (identical distributions) to ∞. The K-L divergence is a value that represents how much extra information is needed to describe the values in the observed session by using a code, which is optimized for the known protocol model instead of using a code optimized for the

session protocol model. The best match for an observed session is the attribute fingerprint which yields the smallest K-L divergence according to Equation 1. P_{attr} and $Q_{attr,prot}$ represent the probability vectors for a specific attribute of an observed session and of a known protocol model respectively.

$$D_{KL}(P_{attr}||Q_{attr,prot}) = \sum_i P_{attr}(i) * log_2 \frac{P_{attr}(i)}{Q_{attr,prot}(i)} \quad (1)$$

Protocol models of observed sessions are finally compared to protocol models of known protocols by calculating the K-L divergences of the models' attribute fingerprints. The best protocol match is the one with the smallest average K-L divergence of the underlying attribute fingerprints. A good approach is to assign a threshold value, where only K-L divergence average values below the threshold are considered matches. If none of the known protocol models match, the session is classified as 'unknown' in order to avoid false-positives for known models.

4 Preliminary Results and Analysis

For the following evaluation, the SPID Algorithm PoC application version 0.3 was used. The SPID PoC application was set to only analyze the first 20 TCP packets in each session and used a K-L divergence threshold of 2.25, which proved to be a good value after a series of empirical tests. However, a thorough evaluation of the impact of the treshhold values is subject of our future work.

We define a *session* as bi-directional TCP flows[3] identified by the 5-tuple[4]. Furthermore, an observed TCP three-way handshake (i.e. a *SYN* or *SYN+ACK* packet) followed by at least one packet with application layer data is required to qualify as a session suitable for classification by SPID.

The SPID PoC application is designed to only identify the application layer protocol in sessions that satisfy the flow requirements described above. The SPID algorithm can, however, be used to identify protocols in any communication scheme where there is a notion of a session, i.e. a uni- or bi-directional flow. This implies that the SPID algorithm can also be used to identify protocols that are transported or tunneled within other protocols such as UDP, HTTP, NetBIOS, DCE RPC, ISO 8073 or even SSL. This generalized functionality is not yet included in the SPID PoC application.

The training data for the protocol models is built from a collection of manually classified TCP sessions from private sources as well as public sources, such as DEFCON 10 CCTF[5], Honeynet.org[6], OpenPacket.org[7] and pcapr[8].

The validation data is a subset of a capture file provided by Szabo et al.[13]. This trace was collected on an access link with capture length of 96 bytes, ie. 42 bytes of application layer

[3] A bi-directional flow consists of the data sent in both directions

[4] A set of: source IP and port, destination IP and port, and transport protocol

[5] http://cctf.shmoo.com/

[6] http://www.honeynet.org/scans/

[7] https://www.openpacket.org/

[8] http://www.pcapr.net/

Protocol	TS	VS	TP	FN	FP	Precision %	Recall %	F %
BitTorrent	31	1245	1221	24	0	100.0	98.1	99.0
eDonkey	19	3535	2744	791	0	100.0	77.6	87.4
HTTP	101	1333	1293	40	0	100.0	97.0	98.5
SSL	73	30	26	4	0	100.0	86.7	92.9
SSH	43	2	2	0	0	100.0	100.0	100.0

Table 2: Validation results per protocol. The columns represent: Protocol identified; #Training Sessions (TS); #Validation Sessions (VS); #True Positives (TP); #False Negatives (FN); #False Positives (FP); % Precision; % Recall; and F-Number.

data. Since the sessions are pre-classified per client-side application, and applications might use multiple protocols concurrently, additional port filtering helped to generate a validation trace consisting of five TCP application layer protocols only:

- ***BitTorrent***: Azureus sessions, excl. ports 80, 10000, 10010 (HTTP) and 27001 (version check protocol)
- ***eDonkey***: eMule sessions, excl. port 80 (HTTP)
- ***HTTP***: Internet Explorer sessions, excl. port 443 (SSL)
- ***SSL***: Internet Explorer sessions to port 443 (HTTP)
- ***SSH***: PuTTY and WinSCP sessions

4.1 Validation Results

The results of SPID, with protocol models build from training data for these five protocols, are summarized in Table 2. Following [15], *Precision* (or accuracy), *Recall* (or hit-ratio), and the combined *F-Measure* are defined according to Equations 2 to 4, where TP, FN and FP stand for *True Positives*, *False Negatives* and *False Positives* respectively.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F - Measure = \frac{2 * Precision * Recall}{Precision + Recall} \tag{4}$$

The results show that the SPID algorithm has the ability to perform very good identification of HTTP and BitTorrent sessions, while performing less complete for eDonkey (78% recall). On average, SPID yields the following results for the five protocols analyzed:

- *Precision*: 100.0% (no false positives)
- *Recall*: 91.9% (few missed sessions)
- *F-Measure*: 95.6% (combined measure)

4.2 Analysis of the Results

The low recall for eDonkey is believed to be due to the limited number of training sessions (only 19) available for this protocol. A richer set of training data will likely provide better results. However, eDonkey is known to be difficult to identify due to the very limited deterministic application layer data. While many existing classifiers are prone to generate false positives for eDonkey [16], SPID produces no false positives on the validation data with a KL divergence threshold of 2.25.

Both BitTorrent and SSH are considered easy to identify using application layer data, since they both start with static protocol banner strings [4, 17]. HTTP on the other hand is a more loosely described protocol, allowing much more freedom in the implementation. The fact that all HTTP traffic in the validation data stemmed from Internet Explorer traffic might explain why SPID achieved such good results, since the used HTTP training data mainly stems from web browsers. We would expect slighly lower recall if HTTP traffic from other applications had been present in the validation data set, such as traffic from HTTP tunnelers or SOAP based Web services.

5 Ongoing and future Work

Even though the current proof-of-concept application shows that SPID can successfully identify network sessions of various protocols, there is still a lot of work to be done.

As a first step, additional protocol models will be created and tested in order to be able to identify most modern Internet traffic. Besides the required training data for new protocol models, existing models will be enhanced with diverse training data from different network locations. It is therefore planned to get in contact with as many interested parties as possible in order to accumulate a database with protocol models with enough natural variation. These groups can include academic institutions, network developers, private individuals or any other interested parties. Please feel free to contact the authors if you would like to contribute.

Another crucial step will be to develop an improved validation framework. Since publicly available, pre-classified data as used in this paper (provided by Szabo et al. [13]) is very rare, we plan to adopt a similar approach like Kim et al. [15]. We will pre-classify our own data using an updated DPI method as introduced by Karagiannis et al. [6] in order to get a reference point when evaluating the performance of SPID.

We then plan to empirically test the accuracy of different attribute meters compared to pre-classified reference data. It is desirable to keep the number of attribute meters low in order to reduce CPU and memory complexity, so we hope to obtain a reduced, optimized set of meters. To each attribute meter we will provide recommendations, such as application protocols or network environments where this specific meter turned out to be especially powerful.

After defining an optimal set of attribute meters, the robustness of this set is planned to be tested against impact of the K-L divergence threshold and effects of different network environments. Some attribute meters, such as those depending on packet payload, are expected to perform similar in different environments. However, meters on flow features like inter-arrival times might be less robust when applied on flows from backbone links with much higher line-

speed compared to LAN links. Besides LAN and access link data, we will have the possibility to test SPID on traces collected on 10Gbit/s backbone links [18].

Protocol identification on backbone links will, however, require some adjustments to the current SPID application. As shown in [19], routing symmetry on highly aggregated links is rare, which means that bi-directional flow data can no longer be assumed. Furthermore, attribute meters disregarding packet payload will become more important, since payload inspection on backbone links is often prohibited due to privacy concerns and legal implications.

6 Summary and Conclusions

In this paper we presented SPID, the Statistical Protocol IDentification algorithm. SPID is utilizing various statistical packet and flow features in order to identify application layer protocols by comparison of probability vectors to protocol models of known protocols.

Initial results have been obtained when identifying a small set of protocols within a pre-classified set of flows collected on an access link. These results are very promising, showing 100% average precision with a recall of 92%. However, a number of interesting and relevant future directions with this approach are discussed, such as optimization of the flow features used or testing the robustness of the algorithm against different network environments, ranging from LAN to backbone links.

We believe that SPID has the potential to become a simple and efficient classification algorithm, providing accurate and fine-grained identification of network flows on application-protocol level. This is important for operational purposes, such as network provisioning, assignment of Quality of Service (QoS) priorities and network security monitoring. Furthermore, current discussions about legal aspects of P2P file-sharing applications add additional value to accurate traffic classification methods such as SPID.

Acknowledgements

The authors want to thank Geza Szabo, Atanas Atanasov, Hyunchul Kim, Tomas Karagiannis and the other authors of [15] and [6] for sharing their code and datasets. Furthermore, Erik is grateful to Jörgen Eriksson for his support and wants to thank his wife Sara for continuous support and motivation.

Erik Hjelmvik was supported by the Swedish Internet Infrastructure Foundation (.SE). Wolfgang John was supported by SUNET, the Swedish University Network.

References

[1] M. Zhang, W. John, kc claffy, and N. Brownlee, "State of the Art in Traffic Classification: A Research Review," *PAM Student Workshop*, 2009.

[2] A. W. Moore and K. Papagiannaki, "Toward the Accurate Identification of Network Applications," *PAM: Passive and Active Measurement Conference*, 2005.

[3] A. Madhukar and C. Williamson, "A Longitudinal Study of P2P Traffic Classification," *MASCOTS*, 2006.

[4] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-network Identification of P2P Traffic Using Application Signatures," *WWW*, 2004.

[5] L7-filter, "Application layer packet classifier for linux," 2009, http://l7-filter.sourceforge.net/ (accessed 2009-04-02).

[6] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC Multilevel Traffic Classification in the Dark," *SIGCOMM*, 2005.

[7] W. John and S. Tafvelin, "Heuristics to Classify Internet Backbone Traffic based on Connection Patterns," *ICOIN*, 2008.

[8] M. Iliofotou, H. Kim, M. Faloutsos, M. Mitzenmacher, P. Pappu, and G. Varghese, "Graph-based P2P Traffic Classifcation at the Internet Backbone," *IEEE Global Internet Symposium*, 2009.

[9] J. Erman, M. Arlitt, and A. Mahanti, "Traffic Classification Using Clustering Algorithms," *SIGCOMM*, 2006.

[10] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic Classification Through Simple Statistical Fingerprinting," *SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 1, pp. 5–16, 2007.

[11] E. Hjelmvik, "The SPID Algorithm - Statistical Protocol IDentification," www.iis.se/docs/The_SPID_Algorithm_-_Statistical_Protocol_IDentification.pdf (accessed 2009-04-02).

[12] B.-C. Park, Y. J. Win, M.-S. Kim, and J. W. Hong., "Towards Automated Application Signature Generation for Traffic Identification," *NOMS: Network Operations and Management Symposium*, 2008.

[13] G. Szabo, D. Orincsay, S. Malomsoky, and I. Szabo, "On the Validation of Traffic Classification Algorithms," *PAM: Passive and Active Measurement Conference*, 2008.

[14] S. Kullback and R. A. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 49–86, 1951.

[15] H. Kim, kc claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet Traffic Classification Demystified: Myths, Caveats, and the Best Practices," *ACM CoNEXT*, 2008.

[16] E. Bursztein, "Probabilistic Identification for Hard to Classify Protocol," in *WISTP: Workshop in Information Security Theory and Practices*, 2008.

[17] Y. Zhang and V. Paxson, "Detecting Backdoors," in *USENIX Security Symposium*, 2000.

[18] W. John, S. Tafvelin, and T. Olovsson, "Passive Internet Measurement: Overview and Guidelines based on Experiences," *Computer Communications*, vol. 33, no. 5, 2010.

[19] M. Dusi, W. John, and kc claffy, "Observing Routing Asymmetry in Internet Traffic," *http://www.caida.org/research/traffic-analysis/asymmetry/ (accessed 2009-04-02)*, 2009.