ALGORITHMS AND TOOLS FOR ANONYMIZATION OF THE INTERNET TRAFFIC

by

Tanjila Farah

B.Sc., BRAC University, 2010

A Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Applied Science

in the
School of Engineering Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Spring 2013

APPROVAL

Name: Tanjila Farah

Degree: Master of Applied Science

Title of Thesis: Algorithms and Tools for Anonymization of the Internet Traffic

Examining Committee: **Veselin Jungic**,
Adjunct Professor
Department of Mathematics
Chair

Ljiljana Trajković, Senior Supervisor
Professor
School of Engineering Science

Parvaneh Saeedi, Supervisor
Assistant Professor
School of Engineering Science

Steve Hardy, Internal Examiner
Professor Emeritus
School of Engineering Science

Date Approved: 26 February, 2013

Partial Copyright Licence

Abstract

Collecting network traffic traces from deployed networks is one of the basic steps in understanding communication networks and ensuring adequate quality of service. Traffic traces may be used for in network management, traffic engineering, packet classification, and for analyzing user behavior. They may also be used for identifying and tracking network anomalies and for maintaining network security. For privacy and security reasons, monitored traffic traces should be anonymization before they may be shared. The goal of anonymization is to preserve traffic properties while enforcing the privacy policies. Various tools and techniques have been implemented for trace anonymization such as: Crypto-PAn, Anontool, FLAIM, ip2anonip, IP-Anonymous, and TCPdprive. These tools use a number of anonymization algorithms such as black-marker, random permutations, truncation, pseudo-anonymization, and prefix-preserving pseudo-anonymization. In this thesis, we present a survey of anonymization tools and algorithms. We propose and implement a new anonymization tool called Anonym, which allows the user to execute multi-level anonymization and to display analysis results. Finally, we evaluate the impact of the anonymization on the anonymized traffic data.

Acknowledgments

This thesis would not have been possible without the support of several thoughtful and generous individuals. My advisor, Professor Ljiljana Trajković, who has provided tremendous insight and guidance both within and outside of the realm of communication networks. My sincere thanks to Prof. Emeritus Steve Hardy, Prof. Parvaneh Saeedi, and Prof. Veselin Jungic for being my committee members and for providing valuable comments and suggestions. Many thanks to Prof. Emeritus Kohshi Okumura for his valuable feedback.

My sincere thanks to Eva María Cavero for her wonderful friendship and support. I would like to extend my appreciation to my colleagues Rajvir Gll, Soroush Haeri, Ravinder Paul, and Nabil M. Al-Rousan for reviewing my thesis.

My deepest gratitude goes to my family for their unconditional love and support throughout my life: my parents for their love, encouragement, advice, and support and my sisters Trisha and Aurpa for being there of me.

Contents

List of Tables

List of Figures

List of Programs

Chapter 1

Introduction

The Internet is currently the the most significant communication medium. It provides an easy yet fast method of information sharing and communication. The Internet and its enormous capabilities have opened immense possibilities to people. From business to education, health care to agriculture, each and every potential arena is enlightened by the proficiencies of the Internet. With its openness and accessibility, the Internet communication has created opportunities worldwide. However, this great medium with all its competences also has its shortcomings.

The openness and accessibility of the Internet also pose privacy and security threats to the Internet users. The Internet security has become a global phenomenon. Hence, it is vital to enforce network monitoring and management. Measurement, characterization, and classification of Internet traces provide information to the network administrators to better manage their network, help improve the Internet performance, and enhance security of the Internet users .

Internet traffic traces are the most significant resource for network researchers, analysts, administrators, designers, educators, and adversaries. Network trace logs are made available by the network service providers or are artificially generated. Real time network analysis relies on collection of trace logs from network service providers. However, sharing network trace logs may reveal the network architecture, user identity, and user information. This makes the network vulnerable to attacks. Collecting network traffic traces from a deployed network requires installment of special purpose devices. Maintaining this type of setup is offer expensive.

Lack of trust between network service providers and research communities is an obstacle

to this vital research area. One solution is anonymization of network traffic traces. Trace anonymization implies modifying a trace to hide the identity and sensitive information contained in trace logs. This process is widely used and there are a number of tools available to anonymize network trace logs. We present here a brief survey of the literature dealing with traffic trace anonymization.

1.1 Related Work

Network trace anonymization is not a new concept. With the growth of the Internet, the concept of sharing network logs has become rather popular. Traffic anonymization was introduced because of the growing concern for network security and user privacy. Anonymization is a compromise between maintaining the research value of the trace and the user privacy. For example, anonymization replaces all occurrence of "Steve" by "Michel" and hence an analyst may correlate data related to an identifier without knowing the initial identifier.

Network traffic logs include data packets. Each packet consists of a packet header and packet payload. The header part of the log files contains information about source and destination Internet Protocol (IP) addresses, port numbers, protocol, type of packet, and other information related to the source and destination of the packet.

Réseaux IP Européens (RIPE) [1], Route Views [2], and Corporative Association for Internet Data Analysis (CAIDA) [3] provide network traffic traces to the research community after anonymizing the collected traces. The fields in a trace should be anonymized and various anonymization techniques and anonymization tools have been considered in the literature [4].

Types of available log files are: pacp, bgpdump, nfdump, iptable, pccat, and netflow. The log files differ in their structure and information they contain. The IP address field is the most relevant for anonymization. IP addresses may disclose information about end users as well as the network architecture. Prefix preserving anonymization is the most frequently used algorithm for IP address anonymization technique. A tool based on this technique is Crypto-PAn [5], [6]. An extension to the key structure was introduced to Crypto-PAn tool to improve its performance [7]. The Anontool supports multiple types of log files [8], [9]. Anontool enables per-field anonymization and provides various anonymization algorithms. Anontool performs faster when compared with other existing anonymization tools. The FLAIM anonymization tool has an extensive application program interface [10]. Both Anontool and

FLAIM support multiple anonymization algorithms for each field of a log file.

There are various trade-offs between privacy, security, and efficiency of the anonymization of data [11], [12] and limitations in the existing anonymization models [13]. Finding a balance between security requirements of the organization that shares the trace logs and their research usefulness has been a topic of research interests [14].

Trace anonymization should preserve the research value of the trace and, hence, the analysis results of un-anonymized and anonymized trace data should lead to similar results. Trace analysis performed by network administrators have been discussed in the past [15]. A thorough analysis of statistical distributions and the best fit test to show the pattern of traffic in a circuit switched network have been reported [16]. Methods and results of network-wide traffic analysis, volume analysis, and detection of anomalous traffic have been addressed [17]. Analysis of un-anonymized and anonymized traces indicate no significant difference between the two [18].

1.2 Contribution

This thesis provides a detailed study of the fields of anonymization and describes the network traffic trace providers and their traces. We analyze various anonymization algorithms based on the anonymized fields and describe three existing anonymization tools. We propose and implement *Anonym*, a new anonimyzation tool that anonymizes stored traffic traces and provides options for traffic analysis. We demonstrate that anonymization does not radically change the statistics of the anonymized data. A summary of the contributions follows:

1.2.1 Implementation of the Anonym Tool

We have developed the Anonym tool to anonymize time-stamps, the IP addresses (IPv4 and IPv6), the Media Access Control (MAC) addresses, port numbers, and packet length fields. Anonym is a MATLAB-based tool that supports pcap and mrt format input files and provides options to convert these text files to pre-anonymized dataset files. The pre-anonymized files are the parsed input files that contain data columns corresponding to fields to be anonymized. The Anonym tool supports multiple anonymization algorithms and data analysis options. The anonymization results and the graphs of data analysis are displayed in the output and figure screens. The results of anonymization are saved in the same format as the input file (pcap or mrt). The Anonym tool supports Linux and Windows environments.

1.2.2 Validation of the Anonym Tool

We compare the Anonym tool with Anontool and FLAIM to validate the performance of the developed tool. We select the same input file for the three tools and compared the output.

1.2.3 Analysis of Un-anomymized and Anomymized Datasets

The developed Anonym tool provides options to analyze datasets. These options include the distribution of various protocols, analysis of traffic volume as function of time, and statistical distributions of packet length. We apply statistical modeling and analysis on both un-anonymized and anonymized datasets using the options provided by the Anonym and show that anonymization does not significantly change the statistics. The traffic traces are processed in a human readable form.

1.3 Thesis Outline

The organization of this Thesis is as follows: Chapter 2 starts with description of collection of network traffic traces and various applications of traffic traces. We describe the organizations that collect traffic traces and provide examples of traffic traces provided by these organizations. In Chapter 3, we addressed anonymization, algorithms for anonymization, anonymized fields, and three existing anonymization tools. In Chapter 4, we describe the Anonym tool and its functionalities, simulation results, statistical modeling, analysis of network performance, and comparison of un-anonymized and anonymized datasets. We conclude with Chapter 5.

Chapter 2

Collection of Network Traffic

The Internet is a collection of Autonomous Systems (ASes). These ASes exchange information and deliver data to the end users. The process of delivering data generates network traffic. Maintenance of network performance and Quality of Service (QoS) rely on network traffic characteristics. Network operators ensure QoS through traffic engineering and network management. It is important to collect and explore the network traffic to systematically analyze and understand the traffic characteristics. Hence, network operators continuously collect and monitor network traffic. Collected traffic is used for traffic engineering, understanding the Internet network topology, and for security analysis. These applications assist in the development of advanced methodologies and techniques to facilitate reliable network operations, conceptualize the network traffic patterns, and optimize network resources. To facilitate these applications, many organizations collect network traffic data and provide collected data to the research community. In this Chapter, we describe the applications of collected network traffic and four organizations that collect data for the research community.

2.1 Applications

2.1.1 Traffic Engineering

Traffic engineering helps the network operators to determine average and high traffic loads. It also identifies traffic that is a result of network attacks [17]. The measurement, characterization, modeling, and control of the Internet traffic requires its understanding. A balanced traffic load is necessary to operate a network smoothly and ensure the QoS. Traffic

engineering deals with network troubleshooting, protocol debugging, traffic workload characterization, performance evaluation, capacity planning, and network attack detection and diagnosis [19]. The roles of traffic engineering are:

1. Network troubleshooting: The Internet network is not perfect. A small malfunction may significantly disrupt or degrade performance of an entire network. Examples of such scenarios are illegal packet sizes, incorrect addresses, or security attacks. In such events, comprehensive measurements from the operational network may often provide a network administrator with the information required to address and solve these issues [19].

2. Protocol debugging: Network traffic measurements provide resources to ensure the correct operation of new protocols and applications. Implementation of any new protocol always faces risks of co-existence with the existing protocols. There are several factors that motivate protocol implementations. Protocols have to meet the de-facto standards. They should co-exist with multiple protocols that provide different services. They should have the option to be modified without effecting the other protocols and applications that depend on them [20]. Protocol debugging assists in determining required improvements to make them compatible to the existing standards.

3. Workload characterization: Network traffic measurements may be used for the workload characterization, which employs statistical techniques to determine properties describing a network application or a protocol. Workload characterization is important for the design of network protocols and applications. Genuine Internet traffic includes various sizes and types of packets, which generate network load and affect network capabilities. Therefore, when designing a network, it is very important to consider the type of packets that occur most frequently in the Internet streams [21] .

4. Performance evaluation: Network traffic measurements help evaluate the performance of Internet protocols and applications. They help identify performance bottlenecks. Recognizing these issues helps develop new versions of protocols and applications designed to provide better performance for the end users.

5. Capacity planning: Network capacity refers to maximum load that a network carries. The utility of a network depends on the capacity planning. Knowing traffic characteristic helps network operators to plan a link load, router capacity, link upgrade, and

schedule network upgrading. A function of traffic engineering is to optimize various parameters. This may be achieved through capacity and traffic management. The role of capacity management includes measuring bandwidth availability, routing control, network delay, delay variation, packet loss, and throughput. Capacity planning is very important for the network operators. For example, effective network capacity planning includes network load balancing and distribution. Planning for network capacity utilization needs constant evaluation of the load in the network and helps identify areas of vulnerability as well as opportunities to better utilize existing resources in the network. This also helps effective implementation of new and redundant paths throughout the network to lower the risk of a single point of failure [22]. Capacity planning parameters are often adjusted by analyzing the network traffic.

2.1.2 Discovering the Internet Network Topology

Network traffic is also used to infer the Internet network topology, which is important for developing technologies, protocols, algorithms, policies, and new network infrastructure [23]. Internet is a complex network. Mapping the Internet network is important for the design of new protocols and applications because it helps visualize the effect on the Internet architecture. It helps the design of new networks and routing policies. It also helps the Internet service providers (ISPs) determine cost of network planning and network management.

2.1.3 Network Security Analysis

Network traffic data are used to enhanced network security by analyzing the network-wide regular behavior to interpret abnormal events such as anomalies, network attacks, and Internet viruses. Anomalies are network events that degrade network performance. Analyzing the network traffic data helps determine the spreading of network anomalies, classify them, and devise diagnostic strategies. Network attacks are events caused by an adversary to eavesdrop into traffic, attain sensitive information, and impair the network. These attacks may occur in different layers of the Internet, as listed in Table 2.1 [17].

Table 2.1: Network attacks in different layers [17].

	Physical layer	Network layer	Transport layer	Application layer
Operational events	Fiber-cuts, accidents	Errors in BGP configurations	Middle box mis-configurations	Configuration of wrong DNS
Network abuse	MAC flooding	Route hijacks	Attacks on TCP	SYN-flood DOS, spam, worms
End-user behavior		Overlay vs. underlay routing	Middle boxes	Flash crowd peer -to peer experiments
Design and bugs	Bugs in 802.11 link layer protocol	ISIS and BGP interactions	TCP fairness violations	HTTP deadlocks, bugs in OS, distributed applications

2.2 Network Traffic Collection

Network traffic is collected using techniques that are either router based or non-router based. Router based techniques of collecting network traffic require additional deployments in network infrastructure. Router based technique may collect traffic from all Organizational Unique Identifier (OSI) layers. That makes the collection process difficult because different organizations may not allow deployment of additional equipment in their networks. This type of arrangement is expensive. Non-router based techniques use software such as Wireshark [24] for capturing network traffic. Even if the infrastructure is available, capturing all traffic from a high bandwidth link at the line speed is not feasible. Furthermore, all traffic information is not useful. Due to these limitations, collection and analysis of accurate data is necessary [25]. Extracting useful information from a huge amount of data is also a difficult process [17].

Various organizations collect different types of data and provide dataset to the research community. In this Thesis, we have examined datasets from four sources: BCNET [26], CAIDA [3], Route Views project from the University of Oregon [2], and Réseaux IP Européens (RIPE) [1]. In this Section, we discuss these organizations, their data collection process, and give examples of the collected data.

2.2.1 BCNET Data Collection

BCNET is the hub of advanced telecommunication network in British Columbia (BC), Canada. It manages the BCs Optical Regional Advanced Network (ORAN). The fiber optic network span of BCNET is 1,400 km. It delivers up to 72 optical wavelengths of up to 10 Gbps to five major cities in BC. BCNET is located in Vancouver Figure 2.1 shows the connection of Prince George, Kamloops, Kelowns, and Victoria with BCNET. This advanced network offers unconstrained bandwidth to research and higher education institutions of BC. This collaboration makes it suitable to address unique research requirements among the research institutes [26].

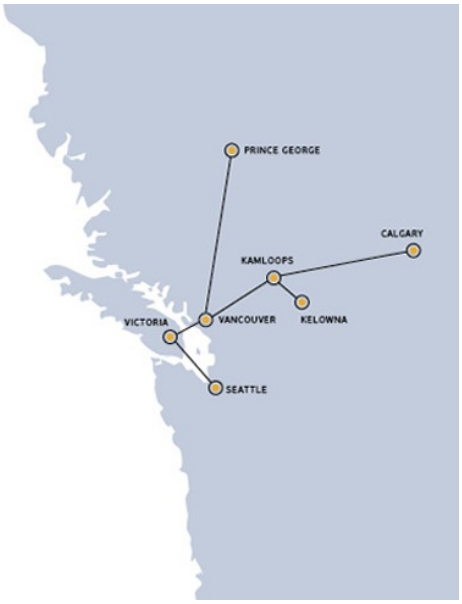


Figure 2.1: British Columbia's advanced network [26].

In collaboration with Communication Networks Laboratory (CNL) at Simon Fraser University, BCNET has installed network traffic capture infrastructure. This infrastructure involves installing a data capture device in the BCNET's network operation center (NOC). As shown in Figure 2.2, the architecture of BCNET NOC has three backbone providers: Shaw Telecom, Tata Communication, and Bell Canada. BCNET's current network consists of two 10-Gbps links and one 1-Gbps link connected to three routers. For capture purposes, a data capture device NinjaBox 5000 is installed in the network as shown in Figure 2.2. The optical test access point (TAP) adjacent to BCNET routers splits the optical signals

in two parts: 70% of the signal is sent to routers for processing while the remaining 30% of the signal is directed to a traffic filtering device [27], [28]. Optical TAP is an efficient method of passively monitoring a link. Nearly all types of networks may be monitored with taps. Optical TAP splitting means dividing the light beam into two separate output paths by a given ratio, which in this case is 70:30 [29] . The traffic filtering device filters the traffic based on a set of parameters and then transmits it to NinjaBox 5000. NinjaBox 5000 is pre-configured with ENDACE Data Acquisition and Generation (DAG) 5.2X card. Endace DAG provides nanosecond granularity for collecting each packet. This provides 100% capture of all packets and ensures the data capture accuracy and reliability [30].

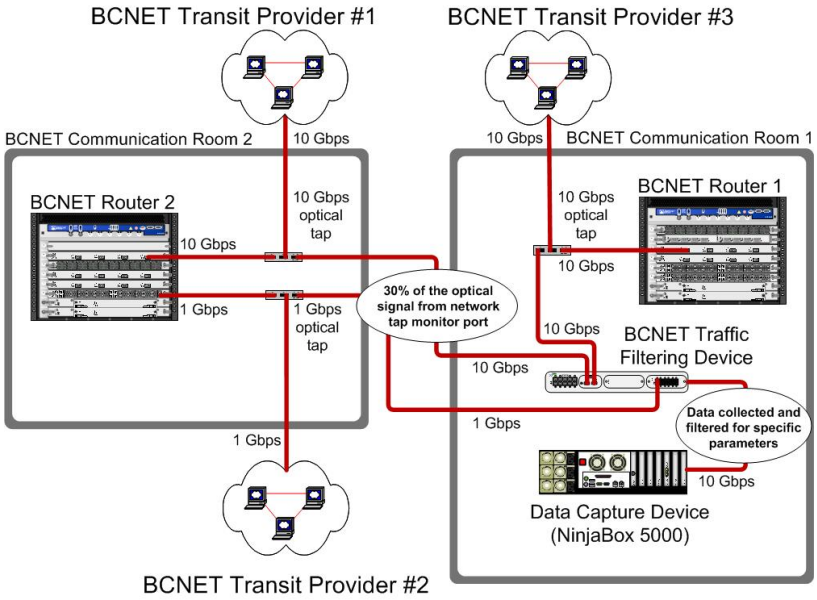


Figure 2.2: The BCNET network operation center [26].

Currently, only Border Gateway Protocol (BGP) update messages are captured by the BCNET. The captures last for various time lengths. A sample packet collected from BCNET is shown in Figure 2.3. It includes the time of the packet capture, source address, destination address, protocol, length of the packet, Ethernet address, and port numbers [26].

Sharing an un-anonymized traffic trace is not common in the Internet research community. The SFU Ethics Board and the Privacy Commissioner of Canada has approved the collection of un-anonymized data from BCNET under the condition that capture may not reveal any data that would constitute personal information or individual identification. To address this issues, the data collected do not include data payload and information from

the network that could be associated with individual users. Current research objective is to characterize, model, and analyze traffic that is collected from a deployed high-speed communication network. The solution to privacy concerns includes a trace collecting system that is designed so neither research staff nor the BCNET administrators may extract privacy-sensitive information. The collected data include only protocol headers from network layer, transport layer, and link layer. Hence, the packet payload contents are safely ignored and cannot be observed.

No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 2607:ffb0:0:4000::1 2607:ffb0:0:4000::2 BGP 201 UPDATE Message
Frame 1: 201 bytes on wire (1608 bits), 201 bytes captured (1608 bits)
Ethernet II, Src: Cisco_ e7:a1:c0 (00:1b:0d:e7:a1:c0),
Dst: JuniperN_ 3e:ba:bd (78:19:f7:3e:ba:bd)
Internet Protocol Version 6, Src: 2607:ffb0:0:4000::1
(2607:ffb0:0:4000::1), Dst: 2607:ffb0:0:4000::2
(2607:ffb0:0:4000::2) Transmission Control Protocol,
Src Port: bgp (179), Dst Port: 63014 (63014),
Seq: 1, Ack: 1, Len: 107 Border Gateway Protocol

Figure 2.3: A sample of collected BCNET datasets (April 2012).

2.2.2 CAIDA Data Collection

The Cooperative Association for Internet Data Analysis (CAIDA) [3] is an organization that of supports the global Internet community with independent, objective, and empirically-based analyses of publicly available Internet traffic data. CAIDA collects data from various commercial, education, research, and government organizations. Several types of data are collected: traffic, topology, traffic statistics, traffic summary, and worm summary. Examples of a sample of worm summary, security, and traffic type data are shown in Tables 2.2, 2.3, and 2.4, and Figure 2.4 respectively. The captured data are used for analysis of routing and addressing, network topology, security, traffic analysis, policy, and visualization. These datasets are publicly available. However, some datasets need access permissions from CAIDA [3].

Table 2.2: Summary of connection types and the number of computers affected by the Witty worm (March 2004) [3].

The Internet connection types	Percentage of the Witty worm-infected population	The total count of infected computers
Broadband	43.70	24,432
XDSL	26.04	14,561
Dialup	17.34	9,696
Cable	11.43	6,390
T1	1.45	812
ISDN	0.03	15

Table 2.3: Summary of the Witty worm (March 2004). [3]

Country	Percentage of the Witty worm-infected population	The total count of infected computers
USA	37.09	20,079
Canada	4.79	2,594
Japan	3.00	1,625
Australia	2.85	1,540
Korea	1.72	933
Argentina	1.71	926
Netherlands	1.70	920
Spain	1.67	903
Taiwan	1.63	880
Hong Kong	1.49	805

No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 196.222.23.126 42.211.89.60 TCP 56 50339 <macromedia-fcs [ACK]
Seq=1 Ack=1 Win=65535 Len=0 [Packet size limited during capture]
Frame 1: 56 bytes on wire (448 bits), 44 bytes captured (352 bits)
Raw packet data Internet Protocol Version 4, Src: 196.222.23.126
(196.222.23.126), Dst: 42.211.89.60 (42.211.89.60)
Transmission Control Protocol, Src Port: 50339 (50339),
Dst Port: macromedia-fcs (1935),
Dst:10.1.6.18 (10.1.6.18) Transmission Control Protocol, Src Port: 32803 (32803),
Seq: 1, Ack: 1, Len: 0 [Packet size limited during capture: TCP truncated]

Figure 2.4: A sample packet from CAIDA topology dataset (June 2011) [3].

2.2.3 Route Views Data Collection

University of Oregon Route Views Project was established to provide a tool for the network operators to get information about global routing systems and view of ASes. Data are collected from various backbone routers from locations around the world. Route Views data are used to detect routing anomalies in the network, visualize BGP routing changes and the growth of routing tables and map ASes, announced prefixes, and IPv4 address spaces to the countries responsible for routing. The data collected by Route Views are in the Multi-Threaded Routing Toolkit (MRT) format. Currently, there are 15 Route Views routers each peering with multiple ASes contributing to this project. These data show BGP routing table information [2]. An example of the Route Views data is shown in Table 2.5. The table gives information of time when packets are captured, BGP message type, source address, AS numbers, destination address, and the network prefix.

Table 2.5: A sample of the collected Route Views dataset (September 2012) [2].

1346500800|B|96.4.0.55|11686|0.0.0.0/0|11686 19151|IGP|
96.4.0.55|0|0||NAG||
1346500800|B|4.69.184.193|3356|1.0.0.0/24|3356 15169|
IGP|4.69.184.193|0|0|3356:3 3356:22 3356:100 3356:123 3356:575 3356:2012|NAG||
1346500800|B|216.218.252.164|6939|1.0.0.0/24|6939 15169|
IGP|216.218.252.164|0|0||NAG||
1346500800|B|168.209.255.23|3741|1.0.0.0/24|3741 15169|
IGP|168.209.255.23|0|0||NAG||
1346500800|B|206.24.210.102|3561|1.0.0.0/24|3561 3356 15169|
IGP|206.24.210.102|0|0||NAG||
1346500800|B|144.228.241.130|1239|1.0.0.0/24|1239 15169|
IGP|144.228.241.130|0|0||NAG||

2.2.4 RIPE

Réseaux IP Européens (RIPE) is a collaboration of organizations operating Internet across Europe, Middle East, parts of Africa, and Asia. The goal of RIPE is to assist the operation of the Internet network in the covered regions. RIPE operates the Routing Information Service (RIS) for collecting and storing the Internet traffic traces from several locations. RIPE provides data in the form of databases to facilitate the network operators. These databases contain datasets that contain information regarding IP address allocations and assignments, routing policies, and forward domain names. RIPE is not an officially authorized organization. Their databases are public. The goal of the RIS database is to collect routing information between ASes by capturing the BGP announcement messages. This information gives an overview of the network routes.

Data collection setup for the RIS database operates by using three programs: a route collector, a database, and a user interface. A diagram of the RIS data collection mechanism is shown in Figure 2.5. The RIPE Network Coordination Center (NCC) uses the collector program that collects traffic from the participating peers. These collected datasets are then transferred to the RIPE NCC. RIS uses the database program to store the collected

network traffic in the database. Any participating peer may access this database using the user interface program. These data are used for debugging the network, comparing policies registered in the routing registry and the policies actually announced, discovering fake routes, and detecting route flaps [1].

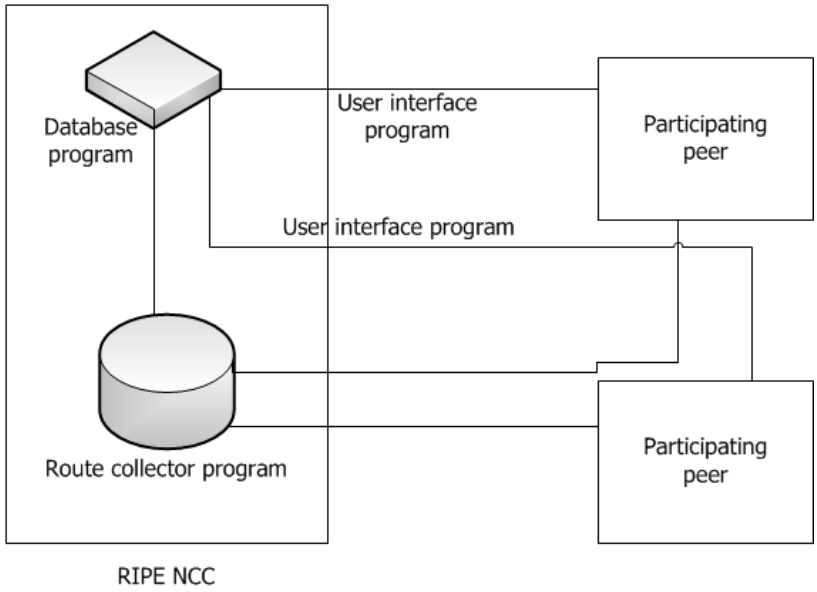


Figure 2.5: The RIS data collection diagram.

A sample of the BGP table collected by RIPE is shown in Table 2.6. The BGP version, collection time, BGP message type, source address, AS numbers, destination address, and the network prefix are shown in Table.

Table 2.6: A sample of dataset collected from RIPE (July 2003) [1].

```
BGP4MP|1059688781|W|194.42.48.11|559|203.196.161.0/24
BGP4MP|1059688782|W|194.42.48.51|12350|216.6.63.0/24
BGP4MP|1059688782|W|194.42.48.51|12350|216.6.97.0/24
BGP4MP|1059688782|A|194.42.48.51|12350|
128.96.128.0/17|12350 3356 209 116 116 116 116|IGP|
194.42.48.51|0|0|3356:3 3356:86
3356:575 3356:666 3356:2006|NAG||
BGP4MP|1059688782|A|194.42.48.51|12350|192.4.80.0/20|12350
3356 209 116 116 116 116|IGP
|194.42.48.51|0|0|3356:3
3356:86 3356:575 3356:666 3356:2006|NAG||
BGP4MP|1059688782|A |194.42.48.51|12350|192.4.112.0/20|
12350 3356 209 116 116 116 116|IGP
|194.42.48.51|0|0|3356:3 3356:86 3356:575 3356:666 3356:2006|NAG ||
BGP4MP|1059688782|A|194.42.48.51|12350|1 92.4.144.0/20|
12350 3356 209 116 116 116 116|IGP
|194.42.48.51|0|0|3356:3 3356:86 3356:575 3356:666 3356:2006|NAG|
```

Table 2.4: Security dataset of the Code Red worm (August 2001) [3].

Start time (s)	End time (s)	Top level domain	Country	Latitude	Longitude	AS number
997427710.383985996	997427710.383985996	es	Spain	40	-4	3352
997763653.148015022	997776078.443505049	arpa	Korea	37	127.5	4766
997593570.365121007	997593570.365121007	arpa	China	35	105	4134
997389453.753429055	997389453.753429055	es	Spain	40	-4	3352
997123519.696187019	997185684.352787971	com	USA	33.03	-96.74	7132
997001899.762797952	997084907.646922946	arpa	China	35	105	4134

Chapter 3

Anonymization

Anonymization is the modification of a network traffic data in order to protect the identity of the end points in the shared dataset. Anonymization attempts to preserve some set of properties of the network traffic useful for a given application while ensuring that the data cannot be traced back to the specific networks, hosts, or users generating the traffic. Internet researchers face a significant challenge when collecting traffic traces because of the fundamental conflict between the end-user privacy and the research value of data. The goal of anonymization is to remove the ability to identify the connection between two end-points while preserving the usefulness of the data.

Network traffic data implies that a sequence of packets flows from source to destination end points, also known as network flow. A network flow contains multiple fields. A flow is generally defined by five fields (5-tuple): source address, destination address, source port number, destination port number, and protocol. A flow record includes additional fields related to the flow such as: packet length, MAC address, flow number, Autonomous System (AS) numbers, window size, maximum segment size, and payload length. Depending on the capture specifications and methods, these fields may be encrypted, fragmented, or unavailable in a flow record. Some of these fields uniquely identify end-points while other fields identify the pattern and behavior of a network user. Anonymization processes are concerned with all the fields to increase the anonymity of the data [31]. Various anonymization algorithms are available. The usually anonymized fields are IP addresses, Media Access Control (MAC), port numbers, packet length, timestamps, and counters. Various anonymization tools are available. These tools anonymize multiple types of log files using various policies. These policies are used to define types of log files and anonymization algorithms used in

various fields.

In Section 3.1, various anonymization algorithms are described. In Section 3.2, data fields, reason for anonymization, and anonymization algorithms are discussed. In this Thesis, we considered tools available from the CAIDA website. Anonymization tools are described in Section 3.3.

3.1 Anonymization Algorithms

Anonymization algorithms or anonymization primitives provide various levels of anonymization and various levels of restrictions to the datasets. Algorithm such as *Black marker* might destroy the dataset while *Prefix-preserving* algorithm preserves the structure of the dataset. The strength and efficiency of an anonymization algorithm is important to preserve the research value of the anonymized dataset [10]. Common anonymization algorithms are listed:

- *Black marker* anonymization algorithm is the most extreme method of anonymization. It deletes or replaces entire information in a field with a fixed value. This method may be applied to any field of a network flow. Although this anonymization technique protects the data in the deleted fields, it also reduces the usefulness of the anonymized dataset. Shown is an example of a BGP packet showing BGP version, time, source IP, AS number, and destination address with prefix, respectively:

  ```
  BGP4MP| 1059688781| W | 194.42.48.11| 559| 203.196.161.0/24
  ```

  If the *Black marker* anonymization is applied to the AS number filed, the value 559 is replaced with zero and the packet will become:

  ```
  BGP4MP| 1059688781| W| 194.42.48.11| 0| 203.196.161.0/24
  ```

- *Enumeration* algorithm works on an well-ordered set of data. The process starts with sorting the data. After sorting the data, the algorithm chooses a value higher then the first value of the sorted dataset. Then this value is added to all the data points in the dataset. This algorithm may not be applied to all fields. The enumerated data is useful for the analysis requiring strict sequencing but not for the analysis requiring good timing information. An example of an *Enumeration* algorithm is shown in Table 3.1. The process starts with a set of un-anonymized data. The set is then sorted in

ascending order. Sorting makes 150 to be the first value of the dataset. The algorithm then selects a higher value of 200 to be added to each data point. The anonymized dataset is shown in the right column of Table 3.1.

Table 3.1: Enumeration algorithm.

Un-anonymized packet length	Sorted	Anonymized
458	150	658
654	254	854
254	458	454
478	478	678
586	586	786
150	654	350

- *Hash* algorithm replaces the data with a fixed size bit string. It uses cryptographic Hash function. Any change in the data will influence the hash value. Sometimes, the results of hash function are shorter than the field value. Therefore, the hash algorithm is easy to break. For example, if the hashed value of an IPv4 address is shorter than 32 bits, dictionary attacks on hashes become very possible. This makes the algorithm week. Another form of Hash algorithm is the Hash Message Authentication Codes (HMAC). This algorithm combines the hash function with a secret key. This prevents the dictionary attacks. This algorithm is used to anonymize text and binary data.

- *Partitioning* algorithm partitions a set of possible values into subsets by a well-defined equivalence relation and a canonical example for each subset is chosen. The anonymization function then replaces every value with the canonical value from the subset to which it belongs [10].

- *Precision degradation* algorithm removes the most precise components of a field. This is mostly used for time stamp anonymization. This process might collapse many time-stamps into one. The anonymized data may not be generally useful for applications that require strict sequencing of flows or precise time-stamp information.

- *Permutation* algorithm is mostly used for anonymization of IP and MAC addresses. It applies a random permutation on the address using a set of possible addresses. This algorithm uses two hash tables: one to map from un-anonymized to anonymized IP

addresses and the other to store all anonymized addresses. There are many variations of permutation functions, each of which has trade-offs in terms of performance. Permutation functions are random and prefix-preserving permutations.

- *Prefix-preserving pseudonymization* algorithm is similar to permutation algorithm. It is a direct substitution system. This algorithm states that if two IP addresses share the first n bits then their anonymized IP addresses will also share the first$n$ bits. The set of anonymized addresses are generated from a block cipher. A block cipher is a method that uses a cryptographic key to generate anonymized addresses. It is used for IP address anonymization. Prefix-preserving pseudonymization algorithm preserves the structure by preserving the prefixes values. Cryptographic keys are used in this algorithm to keep the mapping consistent. An example of this technique is shown in Table 3.2.

Table 3.2: An example of the input and output of Crypto-PAn that uses prefix-preserving pseudonymization algorithm.

IP address	Anonymized IP address
10.1.3.143	117.14.240.136
10.1.6.18	117.14.240.18
192.168.1.2	252.103.242.113
212.204.214.114	*228.71.168.109*
212.204.214.114	*228.71.168.109*

- *Random time shift* algorithm adds a random offset to every value within a field in a dataset. It creates possible set of values to be shifted. It then selects a value from those sets for shifting.

- *Truncation* algorithm is used to anonymize IP and MAC addresses. This algorithm deletes a portion of the data while keeping the rest unchanged. Truncation removes n least significant bits from a field value and replaces them with zeros. This technique is effective to make an end-point non-identifiable.

- *Reverse Truncation* algorithm is used to anonymize IP and MAC addresses. It removes n most significant bits from an IP address and replaces them with zeros. This technique is effective to make a network address or an organization non-identifiable.

- *Time Unit Annihilation* is a type of partitioning algorithm. It is used for time stamp anonymiazation. This algorithm annihilates a portion of the time unit by replacing it with zeros.

3.2 Fields and Relative Anonymization Process

3.2.1 IP Addresses Anonymization

A flow contains source and destination IP addressees. The IP address may uniquely identify hosts in the network. The IP address field is recognized as the most important field to anonymize. Adversaries try to identify the mapping of IP addresses in the anonymized dataset to reveal the hosts and the network. There are multiple anonymization algorithms available to anonymize IP addresses. These algorithms are described in Table 3.3.

Table 3.3: IP address anonymization algorithms.

Anonymization algorithm	Function
Truncation	Truncation replaces a host address with a network address by replacing the least significant bits from IP addresses with zeros. This technique is effective for making hosts non-identifiable and it preserves information that may be used to identify an organization, a geographic region, a country, or a continent.
Reverse truncation	Reverse truncation replaces the network address portion of the IP address with zeros. This technique anonymizes the network bits of an IP address. This anonymization technique destroys the network within the data set.

Permutation	Permutation replaces each IP address with an address selected from the set of possible IP addresses. It preserves the uniqueness of the original address. The selection function is often random. Permutation does not preserve structural information of a network. However, it does preserve the unique count of IP addresses.
Prefix-preserving pseudonymization	Prefix-preserving pseudonymization preserves the structure of subnets at each level while anonymizing IP addresses. If two IP addresses have an n bit common prefix, then two anonymized IP addresses will also have the n bit common prefix.
Black marker	Black marker anonymization completely deletes the data fields and protects data from the risk of disclosure.

3.2.2 MAC Addresses Anonymization

MAC addresses uniquely identify devices on the network. The MAC address information may not be used to locate an end within a network but it may be used to uniquely identify an end device. MAC addresses combined with external databases are mappable to device serial numbers and to the organizations or individuals who purchased the devices. MAC addresses are also often used in constructing IPv6 addresses and, as such, may be used to reconstruct the low-order bits of anonymized IPv6 addresses in certain circumstances.

The MAC address structure contains an Organizational Unique Identifier (OUI) as the three most significant bytes of the address. This OUI additionally contains indication bits that show whether the address is locally or globally administered. MAC address anonymization algorithms are described in Table 3.4.

Table 3.4: MAC address anonymization algorithms.

Anonymization algorithms	Function
Truncation	Truncation replaces a host device address with a organization address by replacing the least significant bits from IP addresses with zeros. It keeps bits of OUI, which identify the manufacturer while removing the least significant bits that identify the particular device.
Reverse Truncation	Reverse truncation zeros out the OUI by removing the first 24 bits. Reverse truncation is effective for making device manufacturers partially or completely unidentifiable within a dataset. However, it may cause uncertainty by introducing collisions of truncated MAC addresses.
Permutation	Permutation replaces each MAC address with a randomly selected address from a set of possible MAC addresses. It preserves the uniqueness of the original address. The selection function is often random. It preserves the unique count of network records in the traffic trace.
Structured pseudonymous	Structured pseudonymous is similar to the permutation algorithm but controls the anonymization precess in such a way that OUI (the most significant three bytes) is permuted separately from the node identifier. This algorithm is useful when the uniqueness of OUIs should be preserved.
Black marker	Black marker anonymization completely deletes a given field.

3.2.3 Time-stamps Anonymization

The time-stamp field itself does not reveal any user information but it may be used as part of attacks against other anonymization algorithms or for user profiling. Time-tamps may be used in traffic injection attacks that uses known information about a set of traffic generated or otherwise known by an attacker to recover mappings of anonymized fields. Time-stamps are also used to identify certain activity by response delay and size fingerprinting, which compares response sizes and inter-flow times in anonymized data to the known values. Time-stamp anonymization algorithms are described in Table 3.5.

Table 3.5: Time-stapms anonymization algorithms.

Anonymization Algorithms	Function
Precision degradation	Precision degradation removes the most precise components of a time-stamp. This has the effect of potentially collapsing many time-stamps into one. With this technique, time precision is reduced and sequencing may be lost. However, the information about the time an event occurred is preserved. The anonymized data may not be generally useful for applications that require strict sequencing of flows.
Enumeration	Enumeration preserves chronological order in which events occurred while removing time information. Time-stamps are substituted by equidistant time-stamps starting from a randomly chosen start value. Enumerated data is useful for applications requiring strict sequencing but not for delay and jitter measurement for quality-of-service (QoS) validation.
Random shifts	Random shifts add a random offset to every time-stamp within a dataset. This reversible substitution technique therefore retains duration and inter-event interval information as well as the chronological order of flows.

Black marker	Black marker algorithm removes all time-stamp information. These data may be used for counting total volume or unique occurrences of other flow keys in an entire dataset [31].

3.2.4 Counter Anonymization

Counters such as number of packets per flow are subject to fingerprinting and injection attacks against anonymization or user profiling. Data sets with anonymized counters are useful only for analysis purposes that do not need precise magnitudes of activity. Counter anonymization algorithms are described in Table 3.6.

Table 3.6: Counter anonymization algorithms.

Anonymization Algorithms	Function
Precision degradation	Precision degradation algorithm removes lower-order bits of the counters, treating all counters in a given range as having the same value. Depending on the precision reduction, the information about the relationships between similar-sized flows is lost.
Binning	Binning may be seen as a special case of precision degradation. In binning the counter ranges are not uniform.
Random noise addition	Random noise addition adds a random value to each value of a field in a dataset. It is used to keep relative magnitude information. It minimizes the relationship information while avoiding fingerprinting attacks against anonymization.

Black marker	Black marker algorithm completely removes information of a counter. This type of anonymization is only recommended for analysis tasks that have no need to evaluate the removed counter such as counting only unique occurrences of other flow keys.

3.2.5 Port Numbers and Protocol Numbers Anonymization

These fields partially identify the applications that generated the traffic in a given trace. This information may be used in fingerprinting attacks to reveal that a certain application with suspected vulnerabilities is running on a network where the trace is collected from. These fields are anonymized using three algorithms described in Table 3.7.

Table 3.7: Port numbers and protocol numbers anonymization algorithms.

Anonymization algorithms	Function
Binning	For source and destination ports anonymization, this algorithm organize the ports in low (0-1023) and high (1024-65535) order sets to identify services from ephemeral ports without identifying individual applications. For protocol field anonymizaiton, this algorithm preserves 1, 17, and 80 for ICMP, TCP, and HTTP traffic while the rest of the protocols into a single bin to mitigate the use of uncommon protocols in fingerprinting attacks.
Permutation	Permutation algorithm replaces the field value by a unique set of values. It might use a hash function to preserve uniqueness.
Black marker	Removes all the information.

3.3 Anonymization Tools

3.3.1 Crypto-PAn

Cryptography-based Prefix-preserving Anonymization or Crypto-PAn is an IP address anonymizer. It uses cryptography based methods to anonymize the IP addresses. The benefit of this anonymization algorithm is that the structure of the networks and subnets are preserved and anonymized [33]. Properties of Crypto-PAn are [6]:

One-to-one mapping: The mapping from the original IP addresses to an anonymized IP addresses is one-to-one.

Prefix-preserving: The IP address anonymization is prefix-preserving: if two un-anonymized IP addresses share a k-bit prefix, their anonymized mappings will also share a k-bit prefix.

Consistent across traces: Crypto-PAn allows multiple traces to be anonymized in a consistent way over time and across locations: the same IP address in different traces is anonymized to the same address even though the traces might be sanitized separately at different time and/or at different locations.

Cryptography-based: A cryptographic key is used to employ anonymization of IP addresses. Anonymization consistency across multiple traces is achieved by using of the same key. The construction of Crypto-PAn preserves the secrecy of the key and the (pseudo) randomness of the mapping from an original IP address to its anonymized counterpart.

Unlike other anonymization tools, Crypto-PAn does not create tables to map IP addresses. Instead, it uses semantic block cipher code and uses the same key to encrypt or decrypt data. For this reason, if data is transferred to a different machine, the result of anonymization or de-anonymization will remain the same. Hence, consistency of the anonymization is preserved. The block cipher used in this tool is the Rijndael block cipher [32]. The hex key is provided by the user for the encryption mechanism. The structure of Crypto-PAn is shown in Figure 3.1.

Crypto-PAn does not support any log files. It takes input ".dat" file that should include only IP addresses to be anonymized and outputs a file with anonymized IP addresses. Crypto-PAn command line is:

```
.\crptopan.exe input.dat >output.dat
```

Anonymization using Crypto-PAn is shown in Table 3.8. Italicized IP addresses show the prefix-preserving and consistent characteristics of Crypto-PAn. The IP addresses share the

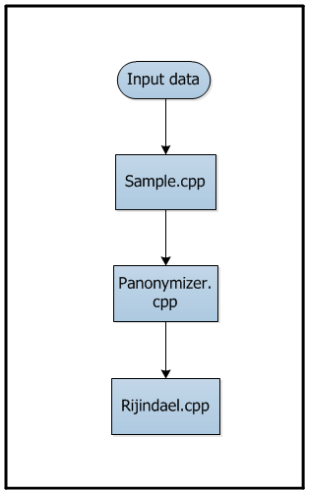


Figure 3.1: Structure of Crypto-PAn.

first 8 and 16 bits of prefix and, hence, during anonymization the same IP prefix is given to the addresses with the common prefix bits.

Table 3.8: The input and output of Crypto-PAn.

IP address	Anonymized IP address
10.1.3.143	*117.14.240.136*
10.1.6.18	*117.14.246.18*
192.168.1.2	252.103.242.113
212.204.214.114	228.71.168.109
212.204.214.114	228.71.168.109
10.1.3.143	*117.14.240.136*
10.1.6.18	*117.14.246.18*
10.1.6.18	*117.14.246.18*
192.168.1.2	252.103.242.2
192.168.1.1	252.103.242.1

Various extensions have been added to Crypto-PAn. One of them is the Lucent's Extensions to Crypto-PAn [34].

Lucent's Extensions to Crypto-PAn

A research group from Lucent introduced extensions to Crypto-PAn by adding properties and correcting the exixting source code. These changes include:

- Crypto-PAn's original and anonymized addresses often have common bits which makes it easy to identify the IP addresses in anonymized traces. To solve this issue, randomization has been introduced in this tool.
- The code has been simplified.
- Source code of this tool is Open secure sockets layer (SSL).

Lucent's Crypto-PAn command line:

```
.\sample sample trace raw.dat >output.txt
```

Anonymization using Lucent's Crypto-PAn is shown in Table 3.9.

Table 3.9: Anonymization with Lucent's Crypto-PAn.

IP address	Anonymized IP address
10.1.3.143	*140.121.242.160*
10.1.6.18	*140.121.245.73*
10.1.6.18	*140.121.245.73*
10.1.3.143	*140.121.242.160*
192.168.1.2	22.92.182.135
212.204.214.114	1.57.253.38
212.204.214.114	1.57.253.38
192.168.1.2	22.92.182.135
10.1.3.143	*140.121.242.160*
192.168.1.2	22.92.182.135

3.3.2 Anontool

Anontool is an open source anonymization tool used to anonymize stored and online traffic traces [35]. It is dependent on a Anonymization Application Program Interface (AAPI) [10] and allows per filed anonymization on a packet or trace log [8].

It allows anonymization of pcap, netflow v5, and netflow v9 log files. The AAPI allows implementation of very simple anonymization policies such as removing payload and complex policies such as altering multiple fields in the HTTP protocol [9]. Anontool allows multiple anonymization algorithms for each field. The algorithms and related fields are shown in Table 3.10.

The anonymization process in Anontool follows four steps:

Table 3.10: Algorithms supported by Anontool.

Fields	Anonymization algorithm
IP addresses	Prefix preserving, Map, Zero
Ethernet addresses	Zero
TCP ports	Map, Zero
TCP/UDP payload	Strip, Zero, Hash

- Cook: Protocols packets such as HTTP, FTP are divided into multiple packets when sent across the network. These packets are reassembled before anonymization is performed. Otherwise, packets with different protocols may not be separable. In the cooking function, all packet are combined as one application level packet. This function also stores the list of headers before cooking.
- Filtering function: The filtering function distinguishes parts of traffic stream and applies policies to anonymized each fields.
- Anonymization function: The anonymization function alters the fields of packets according to the policies defined in the filtering step [9].
- Uncook: After the anonymization function is performed, the uncooking function takes place. An essential property of anonymization is to keep the structure of the trace log intact. The uncooking function acts based on this property. It accumulates the application level packets into its original form by taking the list of header files and by adding these to the appropriate payload. Each packet passes through each of these steps as illustrated in Figure 3.2.

Anontool command line for a pcap log is:

```
./anonymize.tool f input.pcap c a PREFIX t MAP output.pcap\\
```

This command defines the function for anonymization tool to -c fix the checksum bit, -a *Prefix preserve* the IP addresses, and -t map the port numbers. The output of the command is shown in Table 3.11.

3.3.3 FLAIM

Framework for Log Anonymization and Information Management (FLAIM) is a multi level anonymization tool [36]. It supports the anonymization of various logs. It provides the

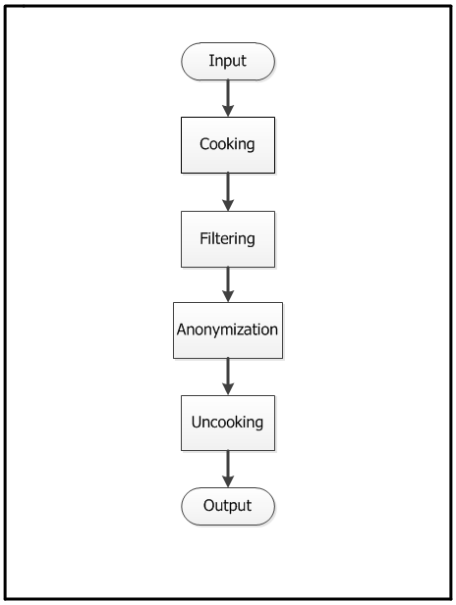


Figure 3.2: Anonymization steps in Anontool.

XML based policy engine that validates and analyzes XML policies. It also provides a simple Application Programming Interface (API), which controls how parsing modules may pass records back and forth within the FLAIMs anonymization engine [37]. Multiple levels of anonymization imply that this tool supports multiple anonymization algorithms for different fields in various types of log files. The algorithms supported by FLAIM are listed in Table 3.12. This table shows various fields that appear in different log files and anonymization algorithms applied on those fields by the FLAIM tool [10]. FLAIM supports pacp, iptable, nfdump, and process accounting module (pacct) type log files.

The architecture of FLAIM consists of FLAIM-Core and FLAIM-Module [21]. The FLAIM-Core consists of the anonymization engine and the XML based policy manager. The FLAIM-Core loads dynamic libraries responsible for I/O and analysis at runtime. There is a module for each type of log FLAIM supports. These modules provide the policy for anonymizing each type of logs. FLAIM was tested on Linux 2.6, Mac OS 10.4-5, FreeBSD 6.1, and OpenBSD 3.9. In this study we used Fedora 16.0. FLAIM dependents on LIBXML, LIBXSLT, and OPENSSL libraries. To run FLAIM, user has to install the FLAIM-Core and the Module [37]. Each module has to be installed separately. FLAIM command line for a pcap module is:

Table 3.11: Anontool anonymization results.

Input:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 *10.1.3.143 10.1.6.18* TCP 74 *32803 h323hostcall* [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: *10.1.3.143 (10.1.3.143)*,
Dst:*10.1.6.18 (10.1.6.18)*
Transmission Control Protocol, Src Port: *32803 (32803)*,
Dst Port: *h323hostcall (1720)*, Seq: 0, Len: 0

Output:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 *117.14.240.136 117.14.246.18* TCP 74 *tcpmux compressnet*[SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1 TSval=5538248
TSecr=0 WS=1
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: *117.14.240.136 (117.14.240.136)*,
Dst: *117.14.246.18 (117.14.246.18)*
Transmission Control Protocol, Src Port: *tcpmux (1)*,
Dst Port: *compressnet (2)*, Seq: 0, Len: 0

```
./flaim -m pcap -p samplepcap.apolicy.xml -i input.pcap -o output.pcap
```

This command defines the function FLAIM to anonymize a module of pcap file using samplepcap.apolicy.xml policy. The result of executing this command is shown in Table 3.13. The table shows the anonymization of a single packet in a pcap log file. The policy for anonymizing the pcap log file enables FLAIM tool to anonymize source and destination IP addresses, user datagram protocol info, MAC addresses, and port numbers.

Table 3.12: Algorithms supported by FLAIM.

Fields	Anonymization algorithm
IP addresses	Truncation, Black marker, Prefix-preserving, Random permutation
Time stamp	Time unit annihilation, Random shift, Enumeration
Mac address	Truncation, Black marker, Random permutation
Host name	Black marker, Hash, Hash Message authentication codes (HMAC)
Port number	Black marker, Random permutation
Network protocol	Black marker
IP id	Black marker
IP options	Black marker
ICMP codes and types	Black marker
Dont fragment bit	Black marker
TCP window size	Black marker
Initial tcp sequence number	Black marker
TCP options	Black marker
Misc IP field	Black marker

Table 3.13: FLAIM anonymization results.

Input:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 10.1.3.143 10.1.6.18 TCP 74 32803 h323hostcall [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1: 74 bytes on wire (592 bits),74 bytes captured
(592 bits) Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 10.1.3.143 (10.1.3.143),
Dst: 10.1.6.18 (10.1.6.18)
Transmission Control Protocol, Src Port: 32803 (32803),
Dst Port: h323hostcall (1720), Seq: 0, Len: 0

Output:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 10.1.3.0 10.1.250.28 TCP 74 0 65535 [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: f1:98:77:ab:fd:05 (f1:98:77:ab:fd:05),
Dst: Iskratel_ 00:00:00 (00:d0:50:00:00:00)
Internet Protocol Version 4, Src: 10.1.3.0 (10.1.3.0),
Dst: 10.1.250.28 (10.1.250.28)
Transmission Control Protocol, Src Port: 0 (0),
Dst Port: 65535 (65535), Seq: 0, Len: 0

Chapter 4

The Anonym Tool

A raw packet trace includes various data fields, that may contain sensitive data. Hence, it is important to carefully choose the fields to be anonymized. Not all data fields require anonymization. Various fields to be anonymized and anonymizaton algorithms have been given in in Chapter 3. Network data may be protected by controlling the access to data or by applying preservation mechanism such as anonymization of the datasets [4]. We have developed the Anonym tool to anonymize five fields. The tool supports pcap and mrt format input files. Anonym is a MATLAB-based tool. The Graphical User Interface (GUI) of the Anonym tool is shown in Figure 4.1. In this Chapter, we discuss validation tests and the functionalities of Anonym.

4.1 Validation Tests

Tests are performed in order to validate the implementation of the Anonym tool. These tests include comparison of the anonymized outputs of Anonym tool, Anontool [8], and FLAIM [10]. We used the same input file for validation purpose. Samples of the anonymized outputs of the three tools are shown in Tables 4.1, 4.2, and 4.3 .

4.1.1 Assumptions

We adopted several assumptions when developing the Anonym tool. Anonymizing the traffic trace is performed off-line. For the analysis section of the Anonym tool, we considered analysis of time-stamps and packet lengths. All functions are configured using the MATLAB

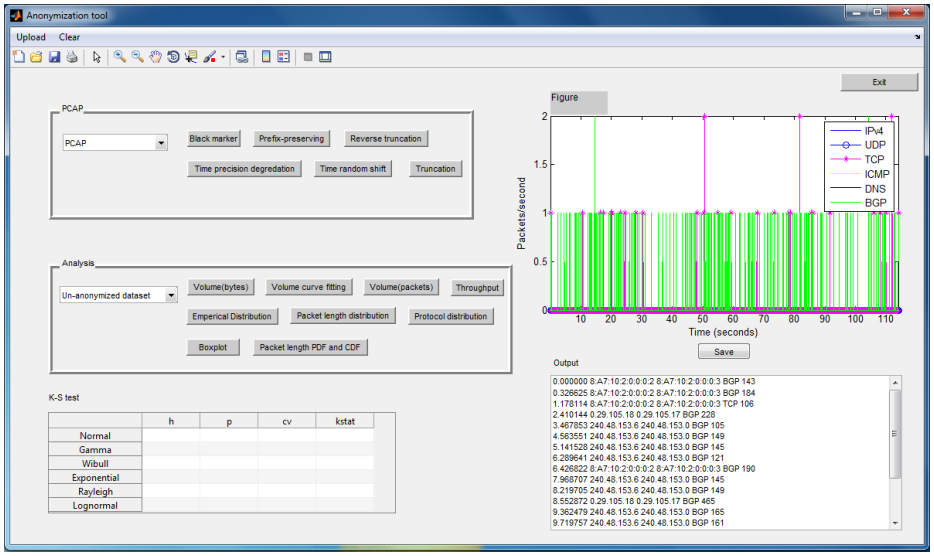


Figure 4.1: GUI of the Anonym tool.

default options. The default values may be modified for the better results.

4.2 The Functionalities of Anonym

4.2.1 Converting pcap and mrt Files

Anonym provides an option to convert a pcap and mrt format text files to pre-anonymized dataset files. The pre-anonymized dataset file is the parsed input file that contains data columns corresponding to the following fields: time, source IP, destination IP, protocol, MAC address, and packet length fields. These fields vary with the type of the input file. Samples of pcap and mrt format files are shown in Figures 4.2 and 4.3, respectively. The structures of these files differ and, hence, parsing mechanism for anonymization of these files also differ. A sample of this parsed dataset is shown in Figure 4.4.

Table 4.1: The results of Anontool anonymization are shown in blue color.

Input:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000010 64.251.87.209 64.251.87.210 TCP 128 32803 h323hostcall [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1:128 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 64.251.87.209 (64.251.87.209),
Dst:64.251.87.210 (64.251.87.210)
Transmission Control Protocol, Src Port: *32803 (32803)*,
Dst Port: h323hostcall (1720), Seq: 0, Len: 0
Output:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000010 110.13.240.136 110.13.246.18 TCP 128 tcpmux compressnet[SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1 TSval=5538248
TSecr=0 WS=1
Frame 1: 128 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 110.13.240.136 (110.13.240.136),
Dst: 110.13.246.18 (110.13.246.18)
Transmission Control Protocol, Src Port: *tcpmux (1)*,
Dst Port: compressnet (2), Seq: 0, Len: 0

No. Time Source Destination Protocol Length Info
1 0.000000 192.168.0.102 239.255.255.250 SSDP 175
M-SEARCH HTTP/1.1
Frame 1: 175 bytes on wire (1400 bits), 175 bytes captured (1400 bits)
Ethernet II, Src: HonHaiPr_ e8:5a:d9 (78:dd:08:e8:5a:d9),
Dst: IPv4mcast_ 7f:ff:fa (01:00:5e:7f:ff:fa)
Internet Protocol Version 4, Src: 192.168.0.102
(192.168.0.102), Dst: 239.255.255.250 (239.255.255.250)
User Datagram Protocol, Src Port: 65404 (65404),
Dst Port: ssdp (1900) Hypertext Transfer Protocol

Figure 4.2: A sample of the pcap file.

Table 4.2: The results of FLAIM anonymization are shown in blue color.

Input:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000010 64.251.87.209 64.251.87.210 TCP 128 32803 h323hostcall [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1: 128 bytes on wire (592 bits),74 bytes captured
(592 bits) Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 64.251.87.209 (64.251.87.209),
Dst: 64.251.87.210 (64.251.87.210)
Transmission Control Protocol, Src Port: 32803 (32803),
Dst Port: h323hostcall (1720), Seq: 0, Len: 0
Output:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000010 64.251.3.0 64.251.250.28 TCP 128 0 65535 [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1:128 bytes on wire (592 bits), 128 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:20:17 (00:04:76:22:20:17),
Dst: Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 64.251.3.0 (64.251.3.0),
Dst: 64.251.250.28 (64.251.250.28)
Transmission Control Protocol, Src Port: 0 (0),
Dst Port: 65535 (65535), Seq: 0, Len: 0

BGP4MP|1059688781|W|194.42.48.11|559|203.196.161.0/24
BGP4MP|1059688781|A|194.42.48.11|559|207.157.57.0/24|559
20965 11537 10490 3464|IGP|194.42.48.11|0|0 ||NAG||
BGP4MP|1059688782|W|194.42.48.51|12350|24.216.174.0/24
BGP4MP|1059688782|W|194.42.48.51|12350|24.216.82.0/24
BGP4MP|1059688782|W|194.42.48.51|12350|216.6.63.0/24
BGP4MP|1059688782|W|194.42.48.51|12350|216.6.97.0/24

Figure 4.3: A sample of the mrt file.

Table 4.3: The results of Anonym anonymization are shown in blue color.

Input:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000010 64.251.87.209 64.251.87.210 TCP 128 32803 h323hostcall [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1: 128 bytes on wire (592 bits),74 bytes captured
(592 bits) Ethernet II, Src:3Com_ 22:20:17 (00:04:76:22:20:17),
Dst:Iskratel_ 10:01:66 (00:d0:50:10:01:66)
Internet Protocol Version 4, Src: 64.251.87.209 (64.251.87.209),
Dst: 64.251.87.210 (64.251.87.210)
Transmission Control Protocol, Src Port: 32803 (32803),
Dst Port: h323hostcall (1720), Seq: 0, Len: 0
Output:
No. Time Source Destination Protocol Length User Datagram Protocol Info
1 0.000000 240.48.153.6 240.48.153.0 TCP 228 0 65535 [SYN]
Seq=0 Win=5840 Len=0 MSS=1460 SACK_ PERM=1
TSval=5538248 TSecr=0 WS=1
Frame 1:128 bytes on wire (592 bits), 74 bytes captured (592 bits)
Ethernet II, Src: 3Com_ 22:0:0 (00:04:76:22:0:),
Dst:Iskratel_ 10:0: (00:d0:50:10:0:0)
Internet Protocol Version 4, Src: 240.48.153.6 (240.48.153.6),
Dst: 240.48.153.0 (240.48.153.0)
Transmission Control Protocol, Src Port: 0 (0),
Dst Port: 65535 (65535), Seq: 0, Len: 0

```
0.000000 2001:4958:10:2::2 2001:4958:10:2::3 BGP 143
0.326625 2001:4958:10:2::2 2001:4958:10:2::3 BGP 184
1.178114 2001:4958:10:2::2 2001:4958:10:2::3 TCP 106
2.410144 64.251.87.209 64.251.87.210 BGP 228
3.467853 206.47.102.206 206.47.102.201 BGP 105
4.563551 206.47.102.206 206.47.102.201 BGP 149
```

Figure 4.4: A sample of the pre-anonymizefile.

4.2.2 Anonymization Fields

The Anonym tool performs anonymization on time, source and destination IP addresses, source and destination MAC addresses, and packet length fields. The options of anonymizing IPv6 addresses are not available in the existing anonymization tools. The IPv6 addresses were introduced to the Internet in June 2012. The traces of IPv6 addresses in network traffic dataset were minimal earlier. The IPv6 uses 128 bit addresses and has an address structure different than the IPv4, as shown in Figure 4.5. The first two lines show the IPv6 address while the next two lines show the IPv4 address. Prefix preserving anonymization process is applied on IP addresses. The MAC address is assigned to the network interface to uniquely identify a device. The Anonym tool provides *Truncation* and *Reverse truncation* algorithms to anonymize MAC addresses. The *Truncation* anonymization process is shown in Figure 4.6.

Un-anonimyzed dataset:
0.000000 2001:4958:10:2::2 2001:4958:10:2::3 BGP 143
1.178114 2001:4958:10:2::2 2001:4958:10:2::3 TCP 106
2.410144 64.251.87.209 64.251.87.210 BGP 228
4.563551 206.47.102.206 206.47.102.201 BGP 149
Anonimyzed dataset:
0.000000 8:A7:10:2:0:0:0:2 8:A7:10:2:0:0:0:3 BGP 243
1.170000 8:A7:10:2:0:0:0:2 8:A7:10:2:0:0:0:3 TCP 206
2.410000 0.29.105.18 0.29.105.17 BGP 328
4.560000 240.48.153.6 240.48.153.0 BGP 249

Figure 4.5: Anonymization process sample of the IP addresses.

4.2.3 Anonymization Options

The Anonym tool provides various anonymization options for each field. We implemented the *Precision degradation, Random shift,* and *Black marker* algorithms for the time-stamp field anonymization. Anonym supports *Black marker, Prefix-Preserving, Truncation,* and *Reverse truncation* algorithms for the IP address anonymization. *Truncation* and *Reverse truncation* algorithms are used to anonymize the MAC addresses. *Enumeration* and *Black*

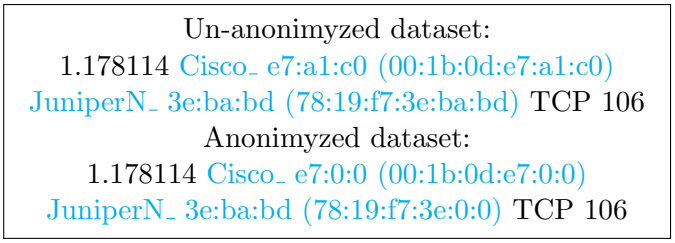


Figure 4.6: Anonymization process sample of the MAC addresses.

marker algorithms are implemented for anonymization of the packet length field. Examples of the algorithms supported by the Anonym tool are shown in Table 4.4.

4.2.4 Option for the Kolmogorov-Smirnov (K-S) Test

Anonym provides an option to employ the Kolmogorov-Smirnov (K-S) test on the datasets [38]. This test is used to determine if a dataset matches specific distribution and the difference between the empirical cumulative distribution function (ECDF) and the tested distribution for a given dataset. The test is also known as goodness of fit tests that measures of the compatibility of a dataset with a theoretical probability distributions function [41]. Anonym supports six hypothesis distributions for testing: Normal, Gamma, Weibull, Exponential, Rayleigh, and Lognormal. The K-S test results include three parameters:

The h value indicates if the dataset follows a tested distribution. This values range between 0 and 1 [16]. Value 0 indicates that the dataset follows the tested distribution while value 1 indicates otherwise. The p value indicates that the probability of a tested distribution matching the ECDF of a dataset. This value ranges between 0 and 1. Value 1 indicates that the dataset matches the tested distribution while the value 0 indicates otherwise. P-value is an approximation of the probability that the tested hypothesis is not false [39]. The k value implies the distance between the ECDF and the test distribution. If this value is less then the critical value, the distribution is rejected [40]. We tested Gamma, Rayleigh, Lognormal, and Weibul distributions of un-anonymized and anonymized datasets. The results of the tests are shown in Table 4.5. None of the distributions is a good fit for the analyzed datasets [16]. A distribution fitting the data is rejected if the test value is h=1. Anonymization does not affect the performance of distribution because the results are the same for un-anonymized and anonymized datasets.

Table 4.4: Results of the Anonym algorithms.

Anonymization algorithms	Time-stamp value	IP address	MAC address	Length
	1.178114	64.251.87.209	Cisco_ e7:a1:c0 (00:1b:0d:e7:a1:c0)	143
Precision degradation	1.170000	Not applicable	Not applicable	Not applicable
Random shift	0.117000	Not applicable	Not applicable	Not applicable
Black marker	0.00000	1.1.1.1	Not applicable	0
Truncation	Not applicable	64.251.0.0	Cisco_ e7:0:0 (00:1b:0d:e7:0:0)	Not applicable
Reverse truncation	Not applicable	0.0.87.209	Cisco_ 0:a1:c0 (0:0:0:0:a1:c0)	Not applicable
Prefix-preserving	Not applicable	0.29.105.18	Not applicable	Not applicable
Enumeration	Not applicable	Not applicable	Not applicable	243

Table 4.5: K-S test results for CDF of packet length data.

Distribution	Parameter	Un-anonymized data	Anonymized data
Gamma	h	1	1
	p	0	0
	k	0.1297	0.1557
Rayleigh	h	1	1
	p	0	0
	k	0.2190	0.2189
Lognormal	h	1	1
	p	1.9891e-197	0
	k	0.0868	0.1233
Weibull	h	1	1
	p	0	0
	k	0.1552	0.1972

4.2.5 Additional Options

The anonymization results and the graphs of data analysis are displayed in the output and figure windows. The results of anonymization are saved in the same format as the input file (pcap or mrt).

4.2.6 Data Analysis Options

Anonym provides various data analysis options. The supported analysis types are volume (bytes), volume curve fitting, volume (packets), empirical distribution, packet length distribution, throughput, protocol distribution, boxplot, and packet length PDF and CDF. The analysis options of Anonym are shown in Figure 4.7. In this Section, we describe the functions of the analysis options of the Anonym tool. We analyze the effect of the anonymization to the statistics of the datasets.

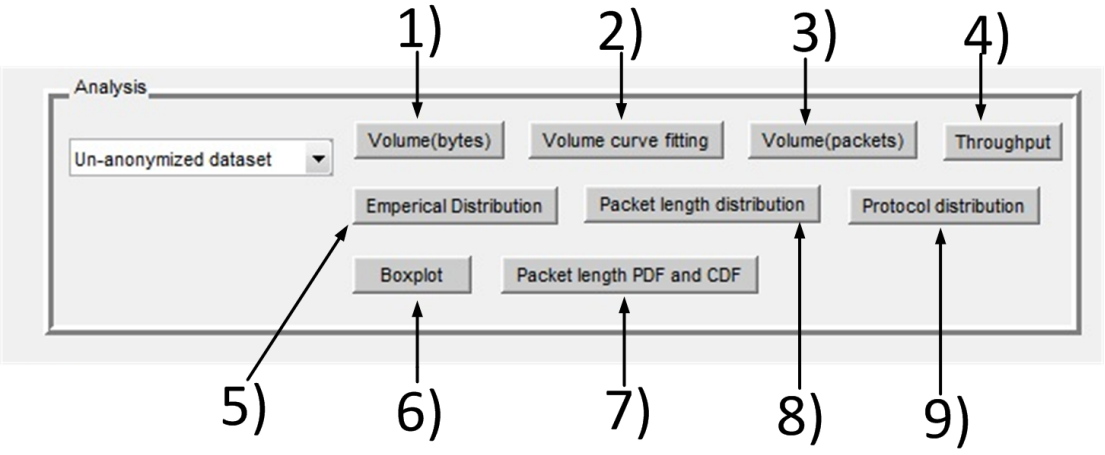


Figure 4.7: Analysis interface of the Anonym tool.

4.2.6.1 Volume (bytes) Option

Analysis of traffic volume is used for classification of network traffic and for network performance management. Volume of traffic is the number of bytes or packets over a period of time. Through volume analysis, the Internet traffic trace data is analyzed as a function of time to find invariant patterns in the trace data. Volume analysis assists in differentiating regular and anomalous traffic. It provides statistics such as the average load and the bandwidth requirements. It is also used to measure the network usage by customers for billing purposes. Traffic generated by the Border Gateway Protocol (BGP), Transmission Control Protocol (TCP), and User Datagram Protocol (UDP) is shown in Figure 4.8. The data is collected from BCNET between May and September 2012. The same traffic pattern appears for un-anonymized and anonymized datasets.

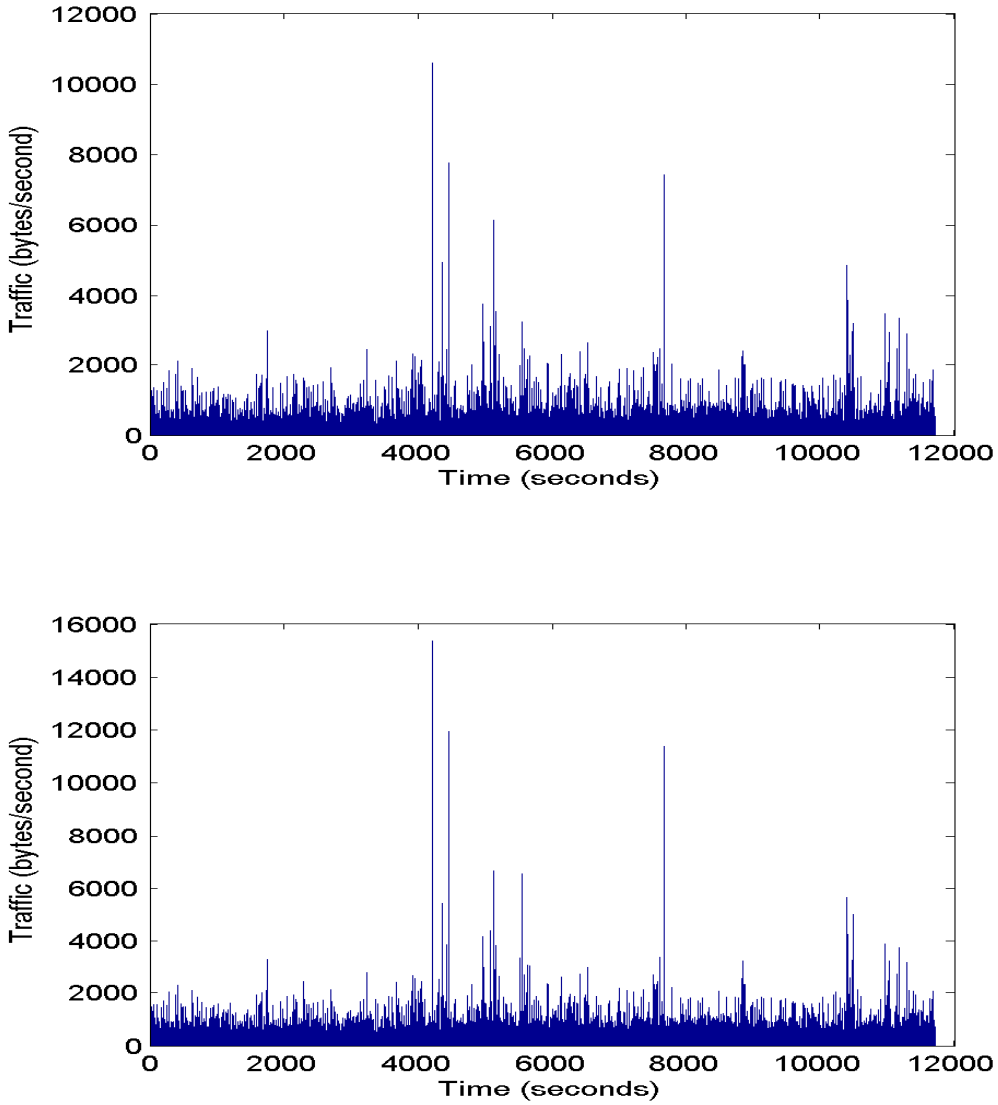


Figure 4.8: Traffic volume in bytes: un-anonymized (top) and anonymized (bottom) datasets.

Minimum, maximum, mean, median, and standard deviation for un-anonymized and anonymized datasets are shown Table 4.6. The values have been increased by 100 bytes due to applying the *Enumeration* algorithms for the anonymized dataset. The *Enumeration* algorithm is described in Chapter 3. Even though the values were shifted by 100 bytes, the standard deviation value remain unchanged change indicating the same structure of the

datasets.

Table 4.6: Statistics of datasets.

Statistics	Un-anonymized dataset	Anonymized dataset
Minimum	60	160
Maximum	1514	1614
Mean	246.2475	346.2475
Median	157	257
Standard deviation	259.4509	259.4509

4.2.6.2 Volume Curve Fitting Option

The volume curve fitting option fits statistical models to a dataset in order to obtain the best fit values of the parameters. This option displays run-sequence plots that graphically summarize a dataset, as shown in Figure 4.9. The un-anonymized and anonymized figures indicate no significant variance. Volume curve fitting option provides choices to fit datasets to exponential, polynomial, Fourier, Gaussian, Weibull, and sum of sine distributions. This option also provide statistics of these distributions: Sum of Squares Due to Error (SSE), the correlation between the response values and the predicted response values (R-Square), Adjusted R-square, and Root Mean Squared Error numbers (RMSE). We considered exponential, polynomial, and Fourier distributions to fit un-anonymized and anonymized datasets, as shown in Figure 4.10. Both graphs show the same fit properties for each fit pattern. We used the default parameters to fit these datasets. Using different parameters may provide better fitting distributions.

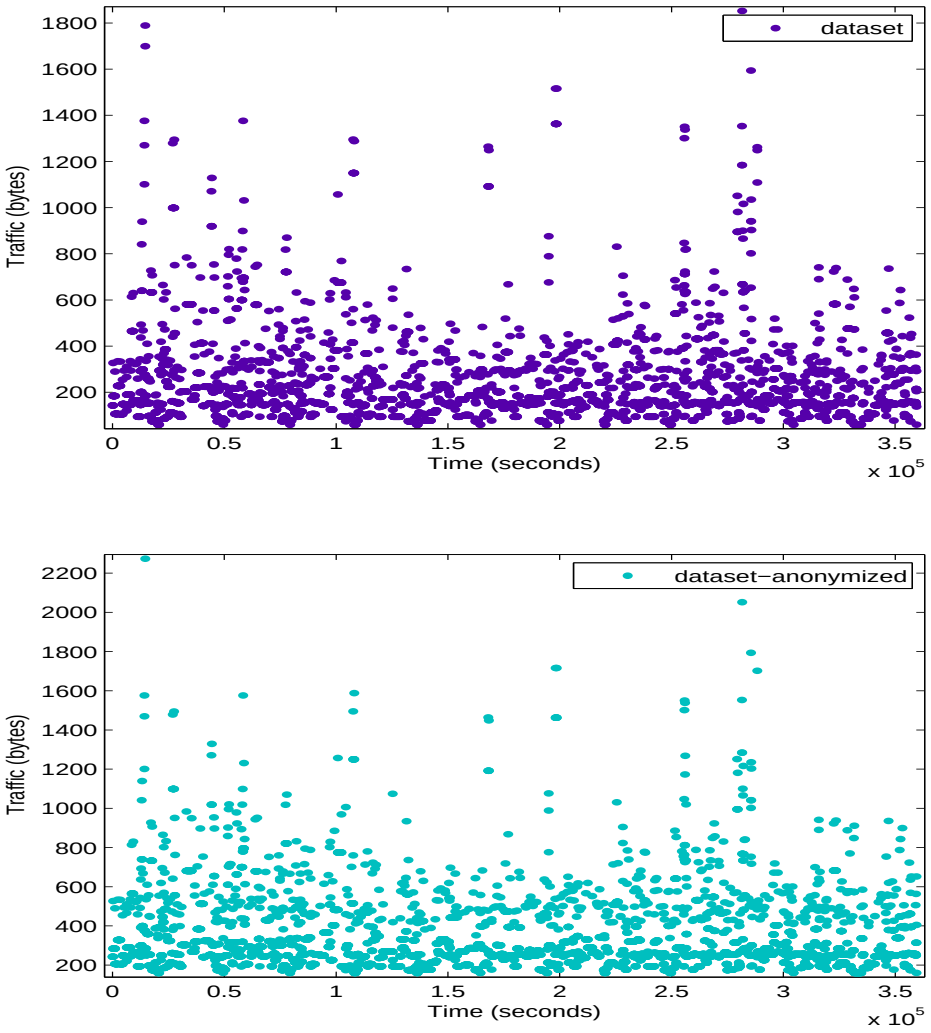


Figure 4.9: Run-sequence: un-anonymized (top) and anonymized (bottom) datasets.

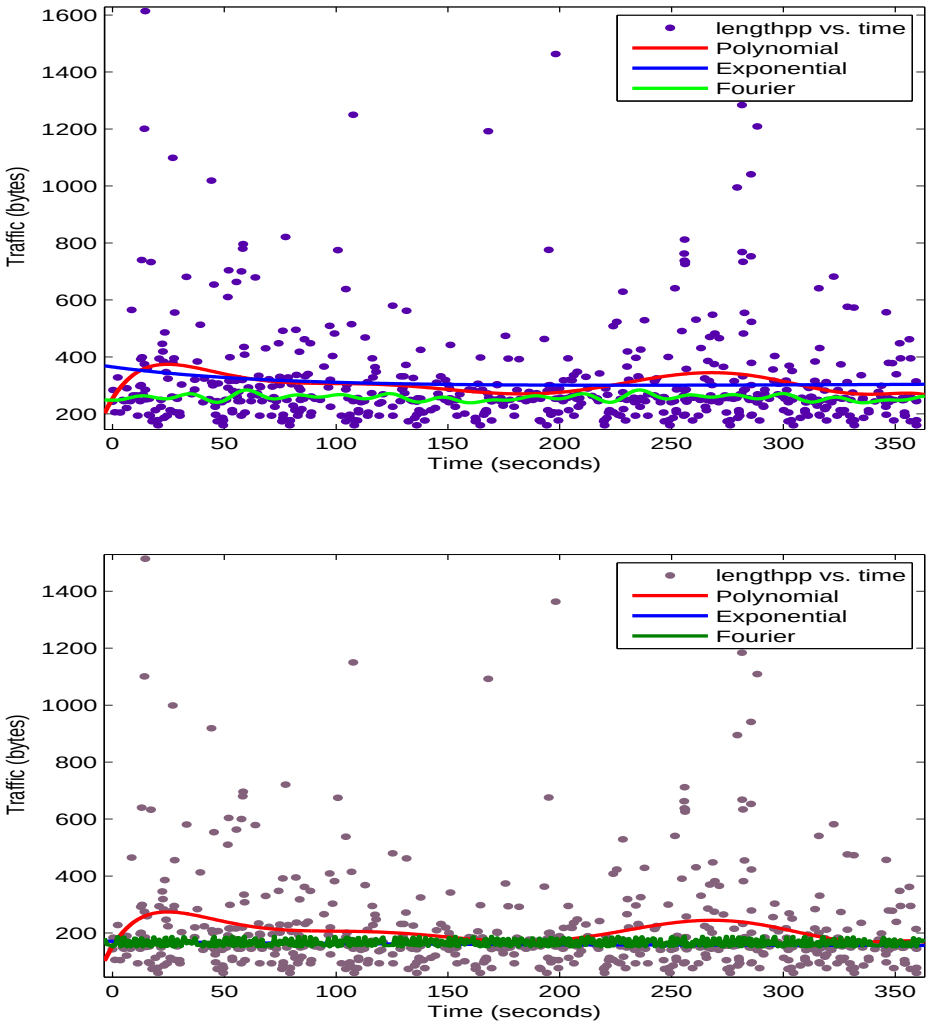


Figure 4.10: Curve fitting to traffic volume: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.3 Volume (packets)

Traffic volume as the number of packet arrived per second is shown in Figure 4.11. In the anonymized dataset, time-stamp field is anonymized using the *Precision degradation* algorithm while the packet length field is anonymized using the *Enumeration* algorithm. The un-anonymized and anonymized datasets show small variations.

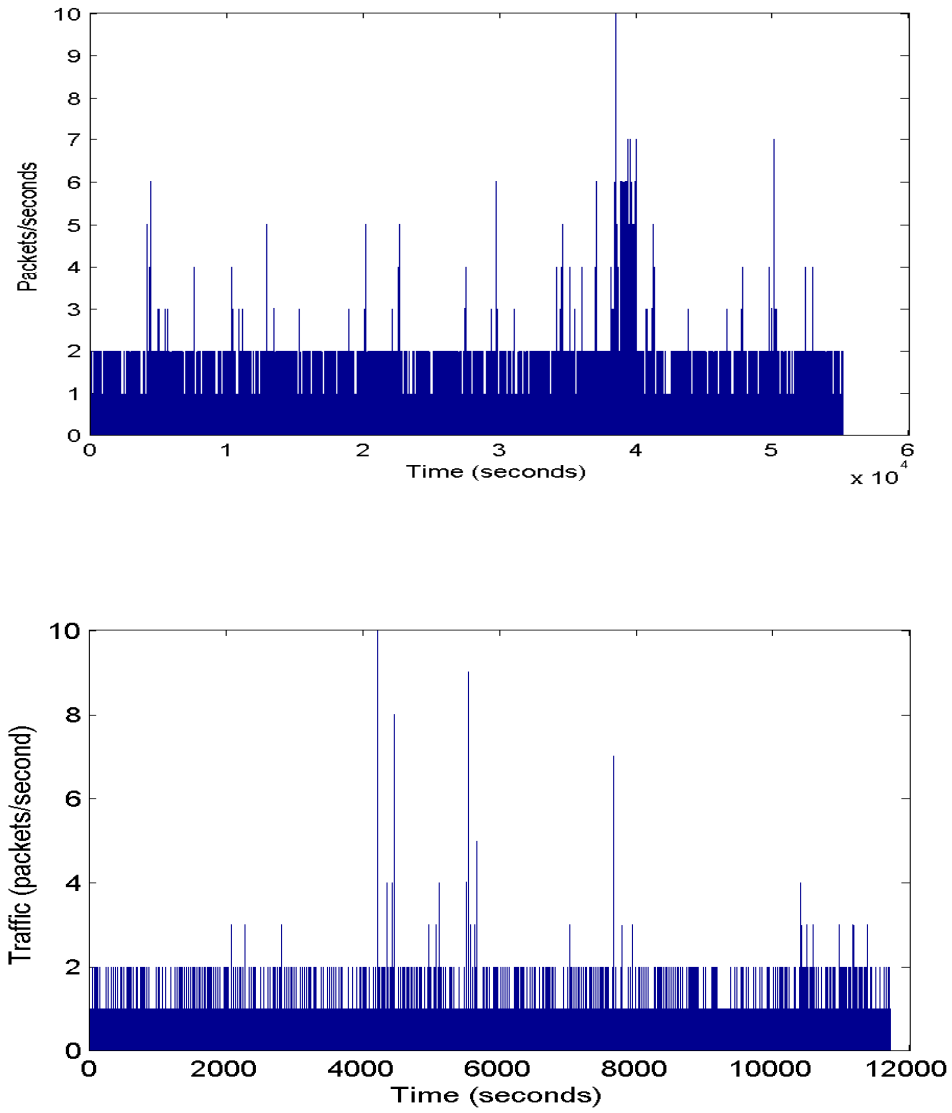


Figure 4.11: Traffic volume in packets: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.4 Throughput

Throughput is the number of application bytes transferred. Throughput statistics provide insight into the implementation of underlying protocols. Incorrect implementation or inefficiency of the protocol may significantly reduce the throughput. The BCNET traffic trace

throughput is shown in Figure 4.12. Anonymization did not change the throughput statistics of the dataset.

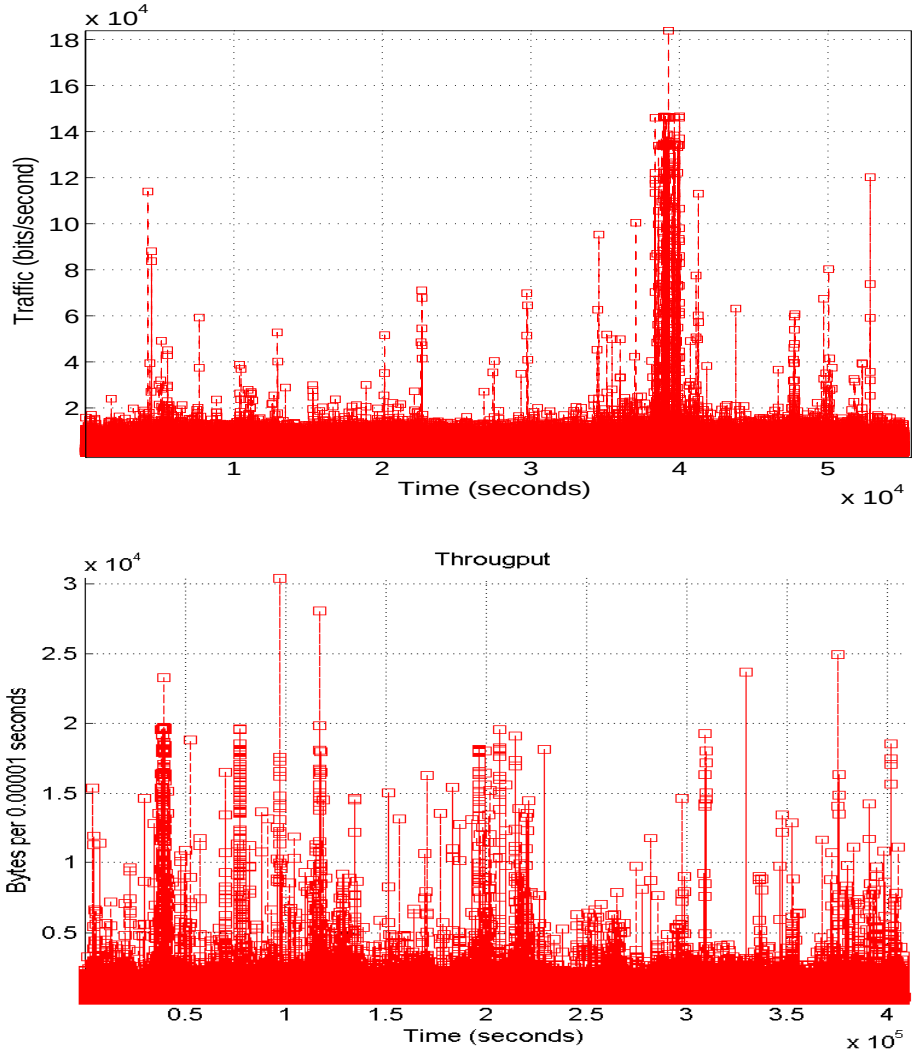


Figure 4.12: Throughput: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.5 Empirical Distribution Option

This option displays the empirical cumulative distribution function (CDF) of the dataset. The empirical distributions of packet length for the un-anonymized and anonymized datasets are shown in Figure 4.13. Packet length data is anonymized by applying the *Enumeration*

algorithm that added 100 byte to each data point, which shifted the emperical CDF graph of the anonymized dataset.

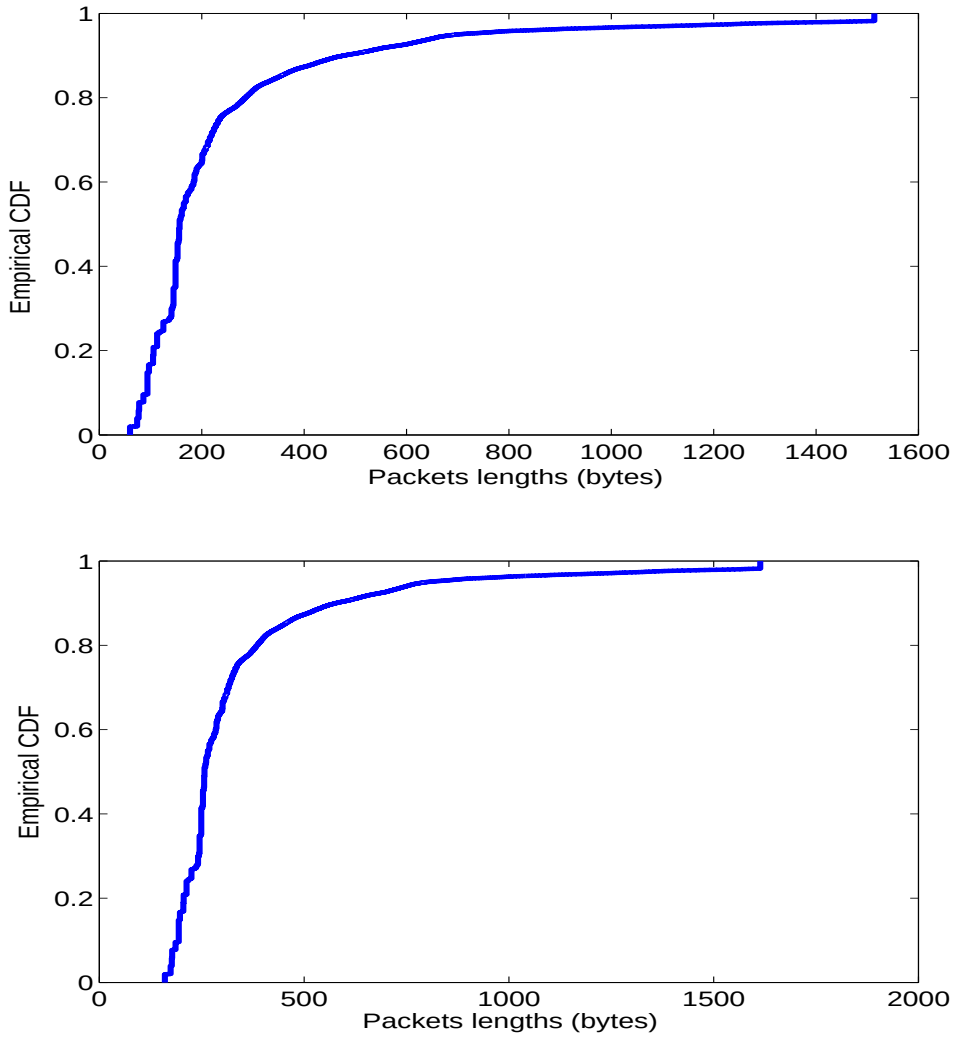
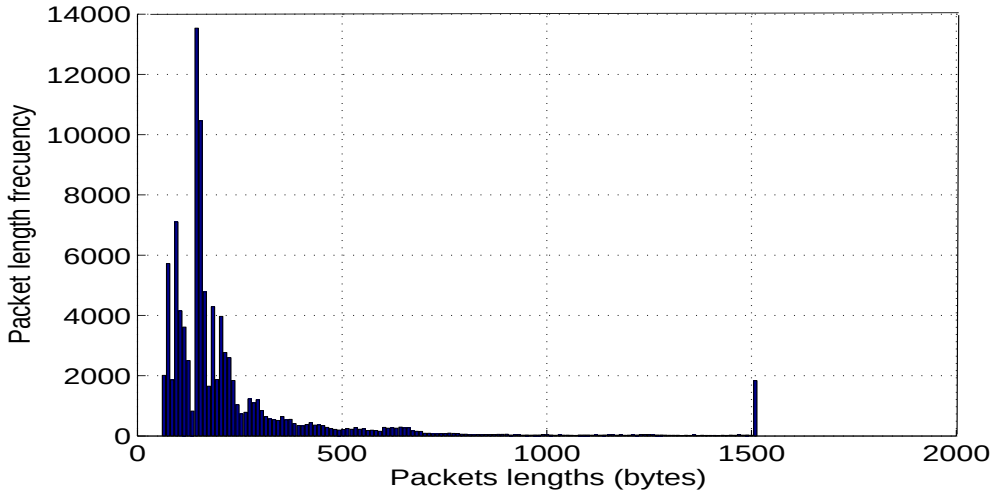


Figure 4.13: Empirical distribution of packet length: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.6 Packet Length Distribution Option

The packet length distribution option displays the histogram plot that indicates the appropriate distribution model of the dataset. Histogram plot reveals center, spread, and outliers

of the dataset. The plot helps determine discrepancies between measured and statistically modeled processes regarding network traffic. It also helps in modeling self-similar network traffic to find the the best approximation of measured traffic in real network [41], [42]. The histogram plots of un-anonymized and anonymized datasets are shown in Figure 4.14. The histogram graph indicates significant percentage of packets with length of 230 bytes for un-anonymized dataset and 250 bytes for anonymized dataset. The packet length data is anonymized using the *Enumeration* algorithm.



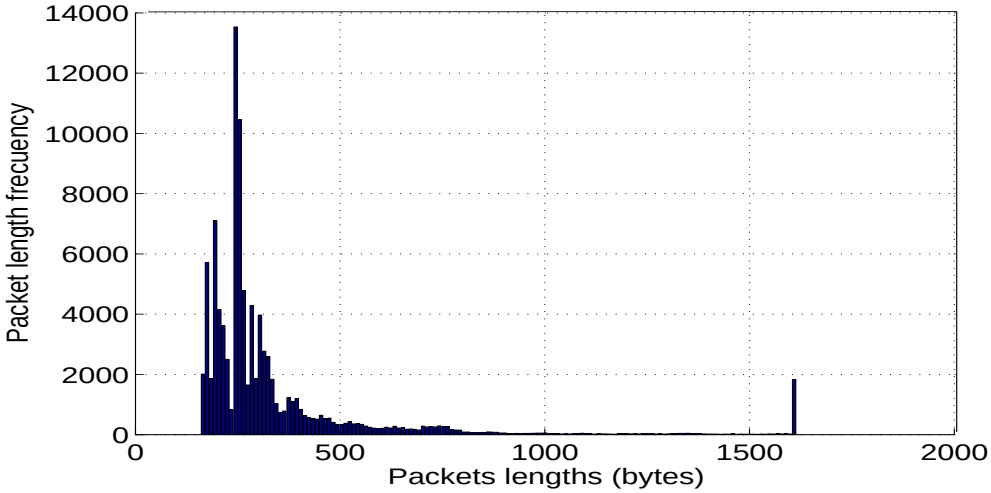


Figure 4.14: Distribution of packet length: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.7 Protocol Distribution

This option classifies the traffic trace according to protocols, as shown in Figure 4.15. The IP, UDP, TCP, ICMP, DNS, and BGP traffic packets are displayed. The BGP packets are the most common in the traffic trace. Protocol distribution provides an overview of the different protocols occupancy in the network. In the anonymized dataset, the time field is anonymized using the *Precision degradation* algorithm.

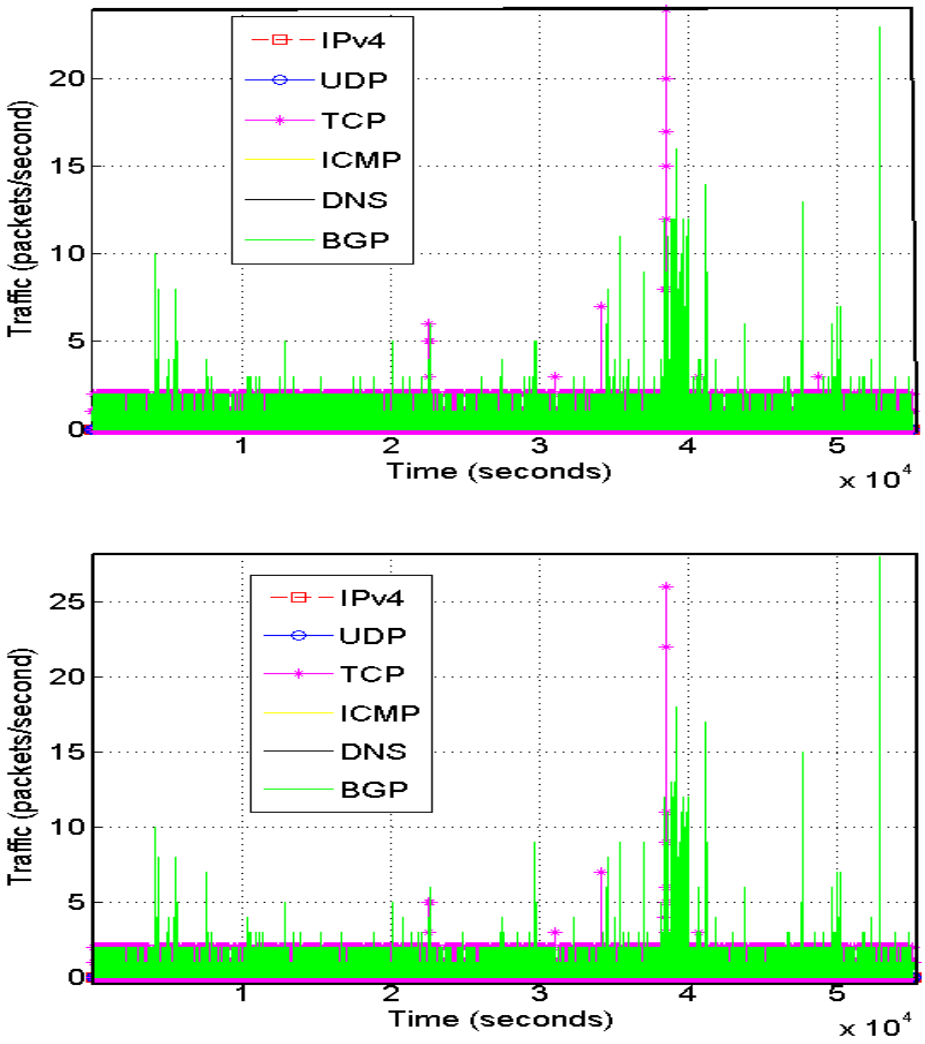


Figure 4.15: Packet distribution by protocol: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.8 Boxplot Option

This option displays the Boxplot of the dataset. Boxplot illustrates position and variation information. It measures the median and the quartiles of a dataset. The median of the dataset is indicated by the red line. The box indicates a range between the lower and the upper quartiles that represents 50% of the dataset. Boxplots of un-anonymized and anonymized datasets are shown in Figure 4.16. 50% of the packet lengths range between

50 bytes and 350 bytes in both datasets. The remaining 50% of the packet lengths in the dataset ranges between 350 and 1,500, indicated by the thin line.

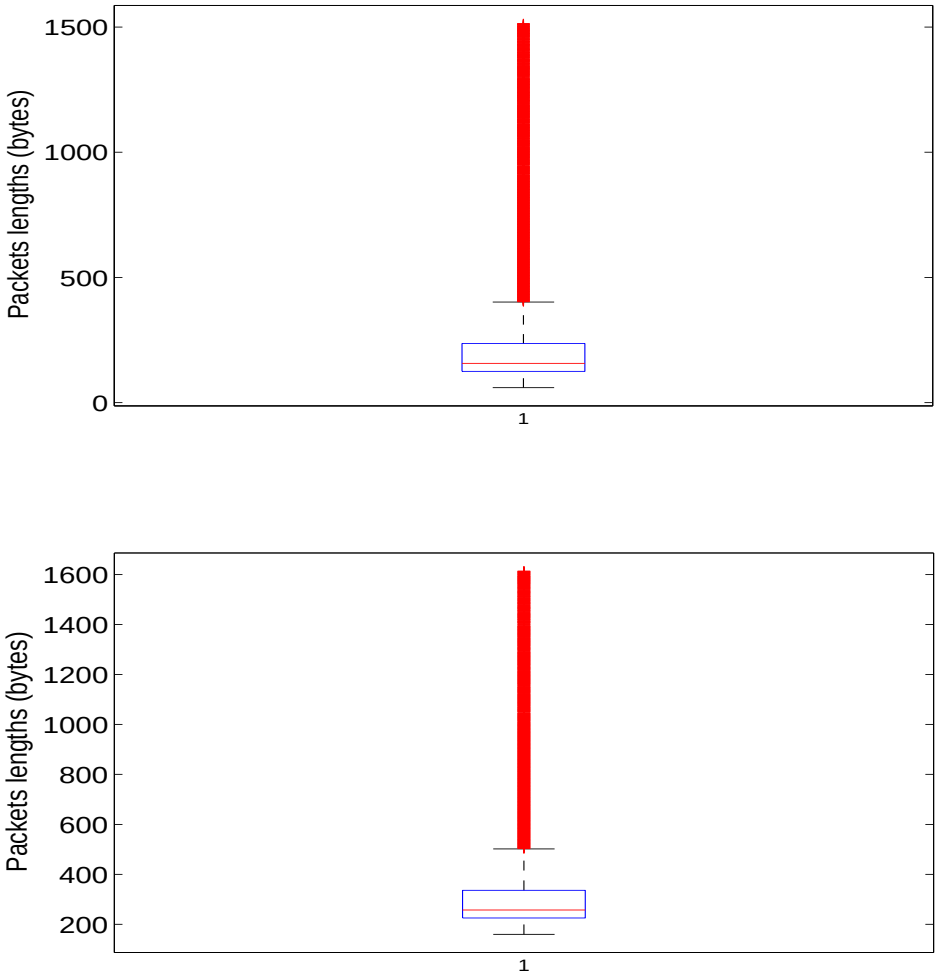
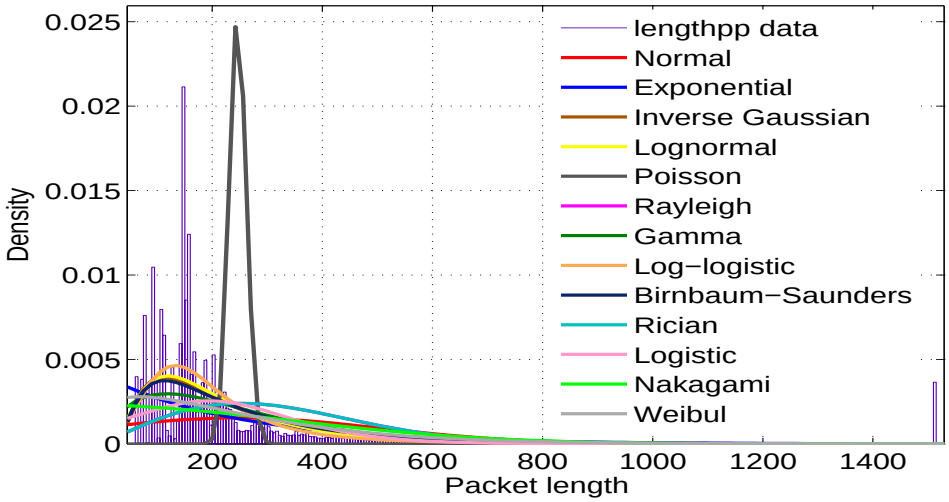


Figure 4.16: Boxpolt of packet length: un-anonymized (top) and anonymized (bottom) datasets.

4.2.6.9 Fitting PDF and CDF Option

Exploring the statistical properties of a traffic trace assists in deriving an appropriate traffic model and employing modeling to determine network performance [43]. The fitting PDF and

CDF options are used to fit the dataset to a theoretical probability density function (PDF), and cumulative distribution function (CDF). The PDF and CDF are characteristics of a probability distribution that characterize the probability that data in a network trace are within a known range [16]. In this analysis, we considered the probability of packet length distribution. The PDF indicates the occurrence of each length of packet in the network trace. The CDF indicates that the probability that the number of occurrence for each packet length will not exceed the calculated value. The un-anonymized and anonymized datasets show no visible variations, as shown in Figure 4.17 and Figure 4.18. The fitting PDF and CDF option also provides list of thirteen distributions to fit to the datasets: Birnbaum-Saunders, Poisson, Normal, Exponential, Gamma, Rayleigh, Lognormal, Weibull, Inverse Gaussian, Rician, Log-Logistic, Logistic, and Nakagami. Fitting the PDF and CDF of the packet length data are shown in Figure 4.17 and Figure 4.18, respectively. The numerical results of Birnbaum-Saunders, Gamma, Rayleigh, Lognormal, and Weibull distributions of the un-anonymized and anonymized datasets are shown in Table 4.7. We used MATLAB default configurations to fit the distributions. Note that none of this distributions is a good fit and selecting different parameters may find a better fit.



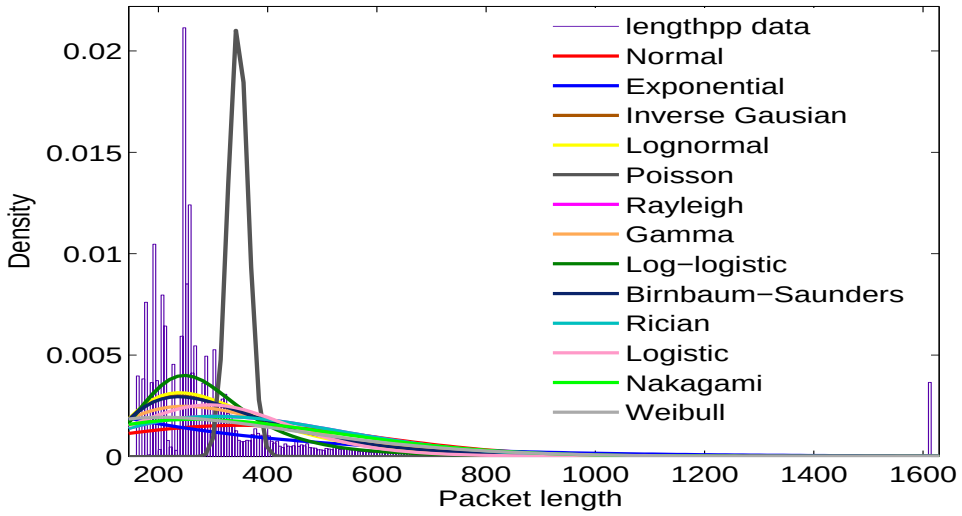
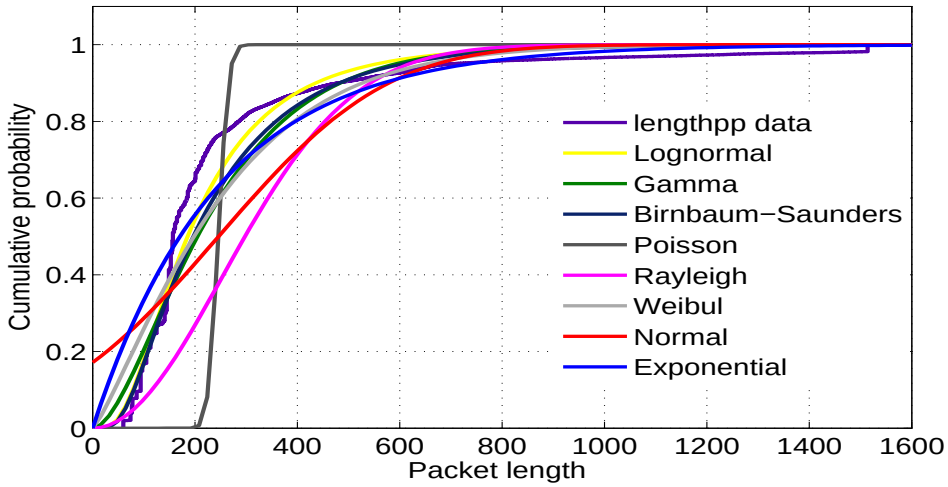


Figure 4.17: PDF of packet length distribution: un-anonymized data (top) and anonymized data (bottom).



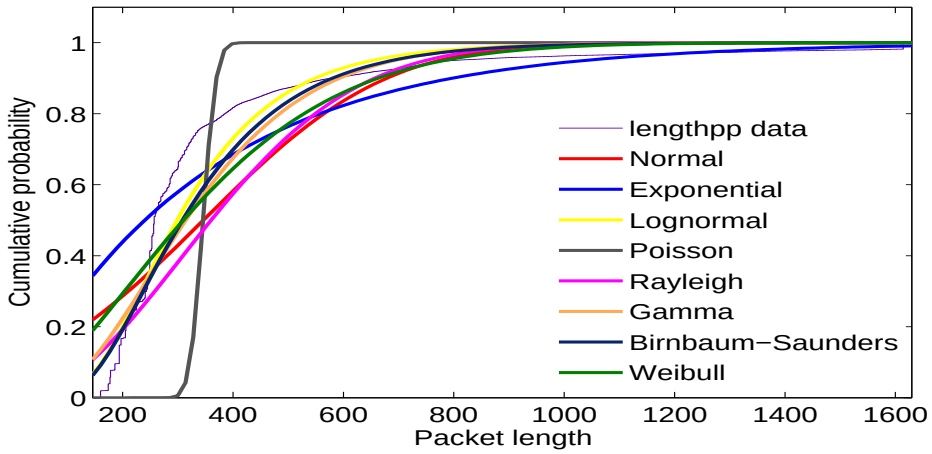


Figure 4.18: CDF of packet length distribution is fitted to Birnbaum-Saunders, Poisson, Normal, Exponential, Gamma, Rayleigh, Lognormal, and Weibull distributions: un-anonymized (top) and anonymized (bottom) datasets.

Table 4.7: Five CDF distributions fitted to the CDF of the un-anonymized and anonymized datasets.

Distribution	parameter	Un-anonymized	Anonymized
Birnbaum-Saunders	Log likelihood	-4.59009e+006	-4.68229e+006
	Mean	231.83	331.143
	Variance	24703.6	23509.9
Gamma	Log likelihood	-4.68286e+006	-4.77124e+006
	Mean	230.646	330.646
	Variance:	23745.4	25467
Rayleigh	Log likelihood	-4.9413e+006	-4.91148e+006
	Mean	282.965	352.354
	Variance	21878.1	33923.5
Lognormal	Log likelihood	-4.57197e+006	-4.66409e+006
	Mean	219.93	321.964
	Variance	22630.8	21531.6
Weibull	Log likelihood	-4.74125e+006	-4.88621e+006
	Mean	234.169	334.072
	Variance	32088.8	40474.6

4.3 Graphical User Interface of the Anonym tool

The Anonym tool is developed with a graphical user interface. The options of the Anonym tool GUI are shown in Figure 4.19, where:

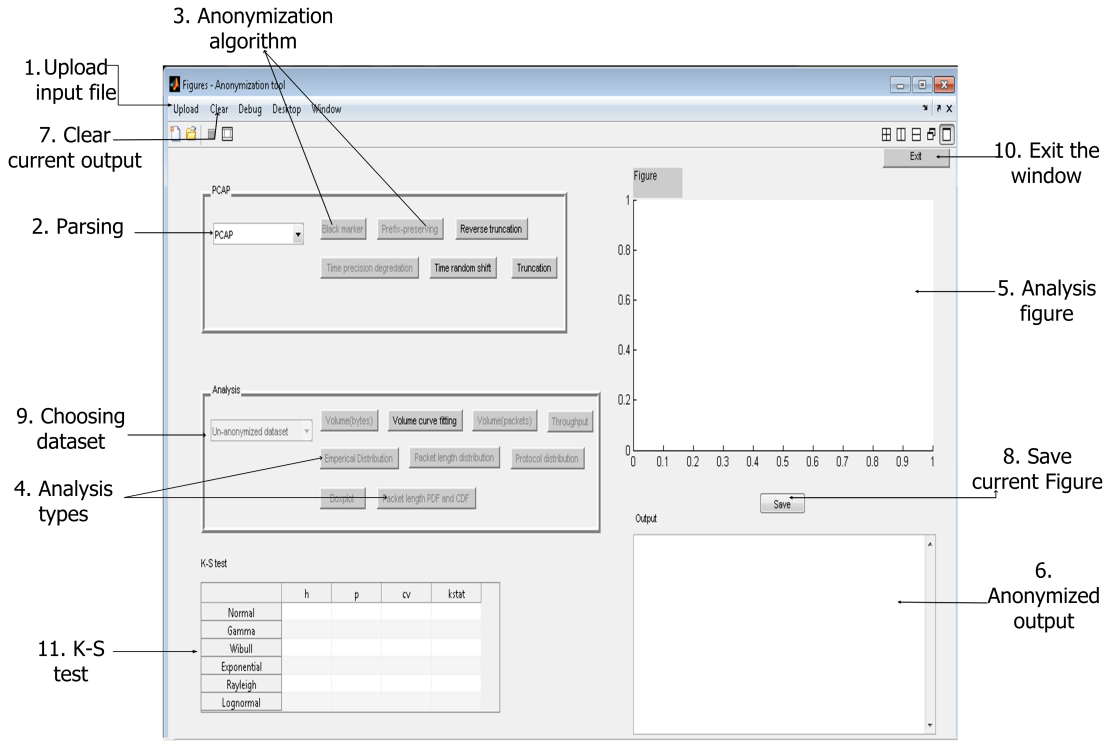


Figure 4.19: Graphical User Interface of the Anonym tool.

1. *Upload* option is used for uploading pcap and mrt format input files.
2. *PCAP and MRT* options are used for parsing the input files for anonymization and analysis.
3. *Anonymization algorithms* that are supported: Black marker, Prefix-preserving, Reverse truncation, Time precision degradation, Time random shift, and Truncation.
4. *Analysis types* that are supported: Volume (bytes), Volume curve fitting, Volume (packets), Empirical distribution, Packet length distribution, Throughput, Protocol distribution, Boxplot, and Packet length PDF and CDF.
5. *Analysis figure* appears in the figure window.

6. *Anonymized output* appears in the output window.
7. *Clear* option removes results from the figure and the output windows.
8. *Save* option saves the figure displayed in the figure window.
9. *Choosing dataset* option selects the dataset for analysis.
10. *Exit* options closes the tool.

4.3.1 Operational Diagram of Prefix-preserving option

The Anonym tool includes thirty options. Each option has separate functionality. The operational diagram of *prefix preserving* option is shown in Figure 4.20. The user first uploads an input file and chooses appropriate parsing function for the file as shown in the upper left corner of Figure 4.20. Once the parsing is done, the function to separate IPv4 and IPv6 flow records is called. This function saves the IPv4 and IPv6 flow records in two separate files. Codes of these functions are listed in Appendix A.

When the prefix preserving option is chosen, two functions for anonymizing IPv4 and IPv6 flow records are called. These functions access the two separate IPv4 and IPv6 flow record files and perform *prefix preserving* anonymization on IP address fields. Two additional functions are called to anonymize time-stamp and packet length fields. The *Time precision degradation* function is called to anonymize time-stamp field while the *Enumeration* function is called to anonymize packet length field. These functions are listed in Appendix A.

After the anonymization of the fields is performed, the anonymized IPv4 and IPv6 flow records are saved in two separate files. A function is called to assemble these flow records in the order as they appeared in the input file. The output function is then called to show the anonymized results in the output window. Another function is called to rewrite the input file with anonymized fields.

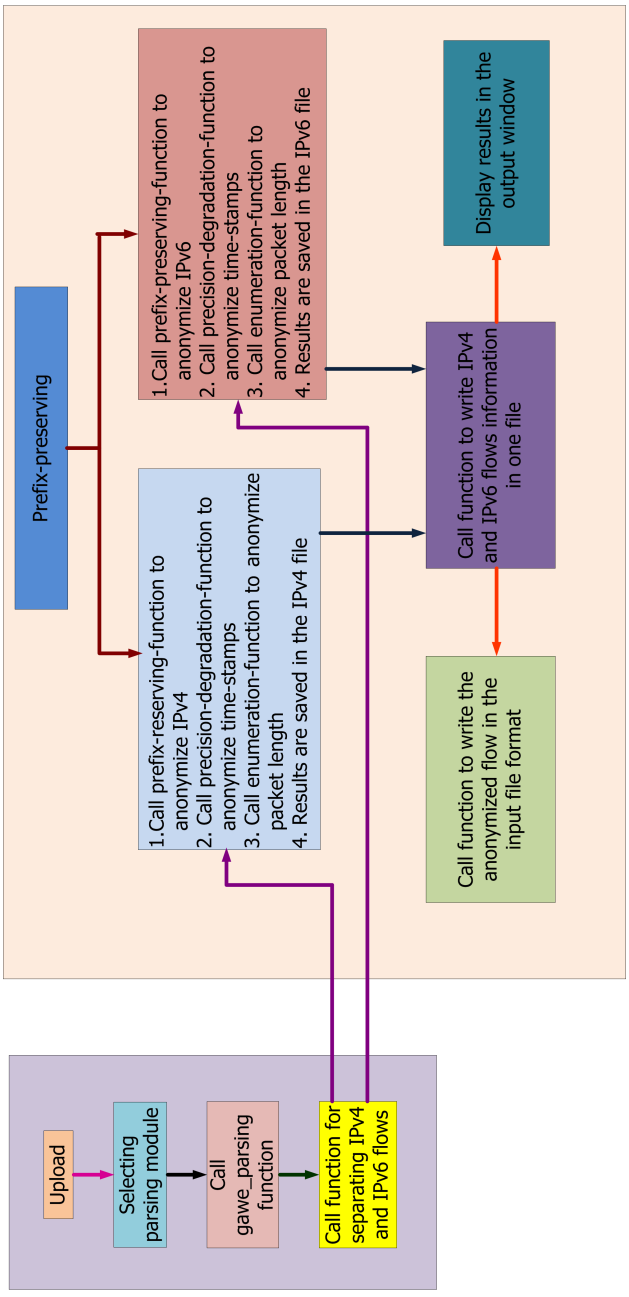


Figure 4.20: Operational diagram of the prefix-preserving option.

Chapter 5

Conclusions

Sharing network trace helps improve network quality and security applications. In this thesis, we propose and implement a network trace anonymization tool called Anonym. Anonym is developed with a GUI. It provides options to anonymize time, IPv4 and IPv6 addresses, MAC addresses, and packet length data. IPv6 anonymization is a new option in the Anonym tool. The Anonym tool also provides various options to anonymize each filed. It supports mrt and pcap format files. For anonymization purpose, the input file is parsed to a human readable format. After anonymization, the results are saved in the input file format. To validate the performance of the Anonym tool, we compare its output with the existing anonymization tools Anontool and FLAIM.

Anonym also provide options to analyze the trace data. Analysis types supported by Anonym are: volume (bytes), volume curve fitting, volume (packets), empirical distribution, packet length distribution, throughput, protocol distribution, boxplot, and packet length PDF and CDF. Traffic data collected from BCNET are used for anonymization and analysis. Analysis results indicate that anonymization does not create significant variations between the un-anonymized and anonymized traces. Anonym provides options to employ the KolmogorovSmirnov (K-S) test on collected dataset, which determines the difference between two datasets based on a null hypothesis. Anonym provides six options to test hypothesis. We applied these tests on un-anonymized and anonymized datasets.

The Anonym tool shows comparable performance to Anontool and FLAIM. It supports Windows and Linux operating systems. Anonym is flexible and extensible and may work with various types of log files if an appropriate parser is written. It supports multilevel anonymization and multiple log formats. Anonym may help network administrators in

sharing anonymized network traces.

5.1 Future work

The study of anonymization depends on many factors. The Anonym tool may be enhanced to support other log formats such as netflow, iptable, and pccat. Other anonymization algorithms may be implemented such as binning, hash, partitioning, permutation, and random noise addition. Anonym currently anonymizes five fields. Since network logs include additional fields such as network protocol, IP ID number, TCP window size, TCP sequence number, anonymization of these fields may also be implemented. Anonym may be enhanced to provide users with options to select fields and anonymization algorithms. The analysis and Kolmogorov-Smirnov test options may be further improved by providing options to choose the sample size from the datasets.

Appendix A

List of Programs

Program A.1: IPv4 address, time, and packet length parsing from pcap format file.

This function parses a pcap format input file and writes a pre-anonymized set file which includes five fields: time-stamp, IPv4 source address, IPv4 destination address, protocol, and packet length.

```
function[time4,IPv4source,IPv4destination,ipv4protocol,ipv4pktlength]
= ipv4decode(name2) IPv4source=[]; IPv4destination=[]; ipv4protocol=[];
ipv4pktlength=[]; time4=[];
fid=fopen(name2);
while 1 line = fgetl(fid); if  isempty(line) spaces = [];
for i=1:size(line,2) if (line(i)==' ') spaces = [spaces i];
end end time4=[time4 str2num(line(1:spaces(1)-1))]; line=line(spaces(1)+1:end);
spaces=spaces-spaces(1); spaces=spaces(2:end);
ipv4text=line(1:spaces(1)-1); dot=[]; for i=1:size(ipv4text,2) if (ipv4text(i)=='.')
dot=[dot i];end end count=size(dot,2);
IPv4sline = zeros(1,4);
IPv4sline(1) = str2num(ipv4text(1:dot(1)-1)); for i=1:count-1 IPv4sline(i+1)
=str2num(ipv4text(dot(i)+1:dot(i+1)-1)); end
IPv4sline(4) = str2num(ipv4text(dot(count)+1:end));
IPv4source = [IPv4source; (IPv4sline)];
ipv4text=line(spaces(1)+1:spaces(2)-1); dot=[]; for i=1:size(ipv4text,2) if
(ipv4text(i)=='.') dot=[dot i];end end count=size(dot,2);
IPv4sline = zeros(1,4);
IPv4sline(1) = str2num(ipv4text(1:dot(1)-1));
for i=1:count-1 IPv4sline(i+1) =str2num(ipv4text(dot(i)+1:dot(i+1)-1)); end
IPv4sline(4) = str2num(ipv4text(dot(count)+1:end));
IPv4destination = [IPv4destination; (IPv4sline)];
ipv4protocol=char(ipv4protocol ,line(spaces(2)+1:spaces(3)-1));
ipv4pktlength=[ipv4pktlength str2num(line(spaces(3)+1:end))];
if (feof(fid)) break; end end end fclose(fid);
```

Program A.2: IPv6 address, time, and packet length parsing from pcap format file.

This function parses a pcap format input file and writes a pre-anonymized set file which includes five fields: time-stamp, IPv6 source address, IPv6 destination address, protocol, and packet length.

```
function [time6,IPv6s,IPv6d,protocol,pktlength]
= ipv6decode(name)

IPv6s=[];
IPv6d=[];
protocol=[];
pktlength=[];
time6=[];
fid=fopen(name);

while 1
line = fgetl(fid);
if  isempty(line) spaces = [];
for i=1:size(line,2) if (line(i)==' ') spaces = [spaces i];
end end time6=[time6 str2num(line(1:spaces(1)-1))];

line=line(spaces(1)+1:end);
spaces=spaces-spaces(1);
spaces=spaces(2:end);
ipv6text=line(1:spaces(1)-1);
colon=[];
for i=1:size(ipv6text,2)
if (ipv6text(i)==':')
colon=[colon i];
end end
count=size(colon,2);
if (count¡7) IPv6sline = zeros(1,8);
IPv6sline(1) = hex2dec(ipv6text(1:colon(1)-1));
fieldcount = 1;
for i=1:count-1 colon(i+1)-colon(i);
if ((colon(i+1)-colon(i))¿1)
fieldcount = fieldcount +1;
IPv6sline(fieldcount) = hex2dec(ipv6text(colon(i)+1:colon(i+1)-1));

else fieldcount = fieldcount + (8-count);
```

```
end end
IPv6sline(8) = hex2dec(ipv6text(colon(count)+1:end));
else IPv6sline = zeros(1,8); IPv6sline(1) = hex2dec(ipv6text(1:colon(1)-1)); for i=1:count-1
IPv6sline(i+1) = hex2dec(ipv6text(colon(i)+1:colon(i+1)-1)); end
IPv6sline(8) = hex2dec(ipv6text(colon(count)+1:end)); end IPv6s = [IPv6s; IPv6sline];
ipv6text=line(spaces(1)+1:spaces(2)-1);
colon=[];
for i=1:size(ipv6text,2)
if (ipv6text(i)==':')
colon=[colon i];
end end
count=size(colon,2);
if (count¡7) IPv6sline = zeros(1,8);
IPv6sline(1) = hex2dec(ipv6text(1:colon(1)-1));
fieldcount = 1;
for i=1:count-1
colon(i+1)-colon(i);
if ((colon(i+1)-colon(i))¿1)

fieldcount = fieldcount +1;
IPv6sline(fieldcount) = hex2dec(ipv6text(colon(i)+1:colon(i+1)-1));

else fieldcount = fieldcount + (8-count);
end end

IPv6sline(8) = hex2dec(ipv6text(colon(count)+1:end));
else IPv6sline = zeros(1,8); IPv6sline(1) = hex2dec(ipv6text(1:colon(1)-1));
for i=1:count-1 IPv6sline(i+1) = hex2dec(ipv6text(colon(i)+1:colon(i+1)-1));
end
IPv6sline(8) = hex2dec(ipv6text(colon(count)+1:end));
end IPv6d = [IPv6d; IPv6sline];
protocol=char(protocol ,line(spaces(2)+1:spaces(3)-1));

pktlength=[pktlength str2num(line(spaces(3)+1:end))];

if (feof(fid))
break;
end
end
end
fclose(fid);
```

Program A.3: Separating IPv4 and IPv6 flow records.

This function separates IPv4 and IPv6 flow records and writes them in two separate files. This function is called after parsing function.

```
clc close all; clear all;
path=[cd '\gawk\'];
name=[path 'chicago-ipv6-2011-parsed.txt'];
name4='ipv4.txt'; name6='ipv6.txt';
iporder=[];
f=fopen(name); f4=fopen(name4,'w');
f6=fopen(name6,'w');
for j=1:40
line = fgetl(f);
if (strfind(line, ':'))
iporder=[iporder 1];
fprintf(f6, 'else iporder=[iporder 0];
fprintf(f4, 'end if (feof(f))
break; end end fclose(f);
fclose(f4); fclose(f6);
```

Program A.4: IPv4 and IPv6 address anonymization using prefix preserving anonymization.

This function applies prefix-preserving anonymization in IPv4 and IPv6 flow records. After anonymization the anonyymized results are written in two separate files for IPv4 and IPv6 flow records.

```
clc close all; clear all;
path=[cd '\gawk\']; name=[path 'chicago-ipv6-2011-parsed.txt']; name4='ipv4.txt';
name6='ipv6.txt';
iporder=[]; f=fopen(name); f4=fopen(name4,'w'); f6=fopen(name6,'w');
for j=1:40
line = fgetl(f); if (strfind(line, ':')) iporder=[iporder 1]; fprintf(f6, 'else iporder=[iporder 0];
fprintf(f4, 'end if (feof(f)) break; end end fclose(f); fclose(f4); fclose(f6);
[time4,IPv4source,IPv4destination,protocol4,ipv4pktlength] = ipv4decode(name4);
v4source= 'v4SColumn.txt'; v4destination='v4DColumn.txt';
inputsize= size(IPv4source,1); ipv4writefile (v4source,v4destination, time4,
ipv4pktlength,IPv4source,IPv4destination,inputsize);
v4Sanonymized='v4sourceanonymized.txt'; v4Danonymized='v4destinationanonymized.txt';
[s,s]=dos(['cryto_ run.exe ' v4source ' >' v4Sanonymized]); [s,s]=dos(['cryto_ run.exe '
v4destination ' >' v4Danonymized]);
[time6,IPv6s,IPv6d,protocol6,pktlength] = ipv6decode(name6);
IPv4s=IPv6s(:,1:4); IPv4d=IPv6d(:,1:4);
lines = size(IPv6s,1);
namev4s='10outv4s.txt'; namev4d='10outv4d.txt';
ipv6toipv4writefile(namev4s,namev4d, time6, pktlength, IPv4s, IPv4d,lines );
namev4sout='10outv4sout.txt'; namev4dout='10outv4dout.txt';
[s,s]=dos(['cryto_ run.exe ' namev4s ' >' namev4sout]); [s,s]=dos(['cryto_ run.exe ' namev4d
' >' namev4dout]);
[IPv4sout,IPv4dout, timeout, pktlengthout]=ipv4toipv6readfile(namev4sout, namev4dout,
lines );
nameano='10outv6ano.txt';
writeipv6anon(time6,IPv6s,IPv6d,IPv4sout,IPv4dout,protocol6,pktlengthout,nameano);
nameano='10outano.txt';
writeipanon(iporder, time6,IPv6s,IPv6d,IPv4sout,IPv4dout,protocol6,pktlengthout,time4,
ipv4pktlength,IPv4source,IPv4destination,protocol4,nameano);
```

Program A.5: precision-degradation-function to anonymize IPv4 flow time-stamps

This function applies precision-degradation anonymization in the time-stamp field of IPv4 flow records. After anonymizing the anonyymized results are written in file with IPv4 flow records.

```
[IPv4source,IPv4destination, time4, ipv4pktlength]
=ipv4readfile(v4Sanonymized, v4Danonymized, inputsize );
precision degradation timeanony4=fix(time4*100)/100;
nameano='10outtimev4ano.txt';
writeipv4anon(timeanony4,IPv4source,IPv4destination,
protocol4,ipv4pktlength,nameano) [IPv4sout,IPv4dout, timeout, pktlengthout]
=ipv4toipv6readfile(namev4sout, namev4dout, lines); precision degradation
```

Program A.6: precision-degradation-function to anonymize IPv6 flow time-stamps

This function applies precision-degradation anonymization in the time-stamp field of IPv6 flow records. After anonymizing the anonyymized results are written in file with IPv6 flow records.

```
[IPv4sout,IPv4dout, timeout,pktlengthout]
=ipv4toipv6readfile(namev4sout, namev4dout, lines); timeanony6=fix(time6*100)/100;
Anonymized IPv6 address output
nameano='10outv6ano.txt';
writeipv6anon(timeanony6,IPv6s,IPv6d,IPv4sout,
IPv4dout,protocol6,pktlengthout,nameano);
```

Program A.7: Output file

This function writes the output file with both IPv4 and IPv6 flow records in the order they appear in the input file.

```
function writeipanon(iporder, time6,IPv6s,IPv6d,IPv4sout,IPv4dout,protocol6,
pktlengthout,time4,ipv4pktlength,IPv4source,
IPv4destination,protocol4,nameano)
fid=fopen(nameano,'w');
lines=length(iporder);
i6=1;
i4=1;
for i=1:lines if (iporder(i)==1)
fprintf(fid, '%f %s:%s:%s:%s:%s:%s:%s:%s ', time6(i6),num2str(dec2hex(IPv4sout(i6,1))),
num2str(dec2hex(IPv4sout(i6,2))),...
num2str(dec2hex(IPv4sout(i6,3))), num2str(dec2hex(IPv4sout(i6,4))),
num2str(dec2hex(IPv6s(i6,5)))... ,num2str(dec2hex(IPv6s(i6,6))),
num2str(dec2hex(IPv6s(i6,7))), num2str(dec2hex(IPv6s(i6,8))));

fprintf(fid, '%s:%s:%s:%s:%s:%s:%s:%s ', num2str(dec2hex(IPv4dout(i6,1))),
num2str(dec2hex(IPv4dout(i6,2))),... num2str(dec2hex(IPv4dout(i6,3))),
num2str(dec2hex(IPv4dout(i6,4))), num2str(dec2hex(IPv6d(i6,5))), ...
num2str(dec2hex(IPv6d(i6,6))), num2str(dec2hex(IPv6d(i6,7))),
num2str(dec2hex(IPv6d(i6,8)))); fprintf(fid,'%s %i\n', protocol6(i6+1,:), pktlength-
out(i6));
i6=i6+1;
else fprintf(fid, '%f %s.%s.%s.%s ', time4(i4),num2str((IPv4source(i4,1))),
num2str((IPv4source(i4,2))),...
num2str((IPv4source(i4,3))), num2str((IPv4source(i4,4))));

fprintf(fid, '%s.%s.%s.%s ', num2str((IPv4destination(i4,1))),
num2str((IPv4destination(i4,2))),...
num2str((IPv4destination(i4,3))), num2str((IPv4destination(i4,4))));
fprintf(fid,'%s %i\n', protocol4(i4+1,:), ipv4pktlength(i4));

i4=i4+1;
end

end
```

Appendix B

Abbreviations

API	Application Programming Interface
AS	Autonomous System
BC	British Columbia
BGP	Border Gateway Protocol
CAIDA	Cooperative Association for Internet Data Analysis
CDF	Cumulative Distribution Function
DAG	Data Acquisition and Generation
DNS	Domain Name Service
ESP	Encap Security Payload
GUI	Graphical User Interface
HMAC	Hash Message Authentication Codes
HTTP	Hyper Text Transfer Protocol
ISDN	Integrated Services Digital Network
ICMP	Internet Control Message Protocol
IP	Internet Protocol
ISP	Internet Service Provider
K-S	KolmogorovSmirnov
MRT	Multi-Threaded Routing Toolkit
MAC	Media Access Control
NCC	Network Coordination Center
OSI	Open Systems Interconnection
OUI	Organizational Unique Identifier

ORAN	Optical Regional Advanced Network
PCAP	Packet Capture
PDF	Probability Density Function
QoS	Quality of Service
SSE	Sum of Squares Due to Error
RMSE	Root Mean Squared Error
RIPE	Reseaux IP Europeens
T1	Transmission System 1
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
XML	Extensible Markup Language
XDSL	x Digital Subscriber Line

Bibliography

[1] (1989-2013) RIPE (Réseaux IP Européens). [Online]. Available: http://www.ripe.net.

[2] (2013) University of Oregon Route Views Project. [Online]. Available: http://www.routeviews.org.

[3] (2013) The Corporative Association for Internet Data Analysis. [Online]. Available: http://www.caida.org/data.

[4] (2013) PRISM state of the art on data protection algorithms for monitoring systems. [Online]. Available: http://fp7-prism.eu/images/upload/Deliverables/fp7-prism-wp3.1-d3.1.1-final.pdf.

[5] J. Xu, J. Fan, M. Ammar, and S. B. Moon, "On the design and performance of prefix-preserving IP traffic trace anonymization," in *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement (IMW' 01)*, San Francisco, CA, USA, Nov. 2001, pp. 263–266.

[6] (2013) Cryptography-based prefix-preserving anonymization. [Online]. Available: http://www.cc.gatech.edu/computing/Telecomm/projects/cryptopan/.

[7] A. Slagell, J. Wang, and W. Yurcik, "Network log anonymization: application of Crypto-PAn to Cisco netflows," in *Proc. NSF/AFRL Workshop on Secure Knowledge Management (SKM' 04)*, Buffalo, NY, USA, Sept. 2004, pp. 223-228.

[8] M. Foukarakis, D. Antoniades, S. Antonatos, and E. P. Markatos, "Flexible and high-performance anonymization of netflow records using anontool," in *Proc. Third International Workshop on the Value of Security Through Collaboration (SECOVAL' 07)*, Nice, France, Sept. 2007, pp. 33–38.

[9] D. Koukis, S. Antonatos, D. Antoniades, E. Markatos, and P. Trimintzios, "A generic anonymization framework for network traffic," in *Proc. IEEE International Conference on Communications (ICC' 06)*, Istanbul, Turkey, June 2006, vol. 5, pp. 2302–2309.

[10] A. Slagell, K. Lakkaraju, and K. Luo, "FLAIM: a multi-level anonymization framework for computer and network logs," in *Proc. 20th conference on Large Installation System Administration (LISA' 06)*, Washington, DC, USA, July 2006, pp. 101–115.

[11] M. Bishop, B. Bhumiratana, R. Crawford, and K. Levitt, "How to sanitize data," in *Proc. 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE' 04)*, Washington, DC, USA, June 2004, pp. 217–222.

[12] P. Porras and V. Shmatikov, "Large-scale collection and senitization of network security data: risks and challenges," in *Proc. Workshop on New Security Paradigms (NSPW '06)*, Germany, Oct. 2007, pp. 57–64.

[13] R. Crawford, M. Bishop, B. Bhumiratana, L. Clark, and K. Levitt, "Senitization models and their limitations," in *Proc. 2006 Workshop on New Security Paradigms (NSPW' 06)*, Germany, Mar. 2007, pp. 41–56.

[14] R. Pang, M. Allman, V. Paxson, and J. Lee, "The devil and packet trace anonymization," *ACM SIGCOMM*, Aug. 2006, vol. 36, pp. 29–38.

[15] H. Kapri, "Network traffic data analysis," M.S. Thesis, Electrical and Communication Engineering, LSU, Baton Rouge, LA, USA, December 2011.

[16] B. Vujicić, C. Hao, and Lj. Trajković, "Prediction of traffic in a public safety network," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS' 06)*, Kos, Greece, May 2006, pp. 2637–2640.

[17] A. Lakhina, "Network-wide traffic analysis: methods and applications," Ph.D. dissertation, BU, MA, USA, 2007.

[18] J. Verdú, J. García, M. Nemirovsky, and M. Valero, "Analysis of traffic traces for stateful applications," in *Proc. Network Processors and Applications (NP3' 04)*, Madrid, Spain, Feb. 2004, pp. 223–238.

[19] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of Internet traffic engineering," RFC 3272 (Informational), Internet Engineering Task Force, may 2002, updated by RFC 5462. [Online]. Available: http://www.ietf.org/rfc/rfc3272.txt

[20] C. A. Thekkath, T. D. Nguyen, E. Moy and E. D. Lazowska, "Implementing network protocols at user level," in *Proc. IEEE/ACM Trans. Network*, Oct. 1993, pp. 554–565.

[21] M. Miyazakii, "Workload Characterization and Performance for a Network Processor," M.S. Thesis, Electrical Engineering, Princeton University, Princeton, NJ, USA, June , 2002.

[22] (2013) Network performance and capacity planning: Techniques for an e-business world. [Online]. Available: http://www-07.ibm.com/services/pdf/nametka.pdf.

[23] L. Subedi and Lj. Trajković, "Spectral analysis of Internet topology graphs," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS' 10)*, Paris, France, June 2010, pp. 1803–1806.

[24] (2013) Wireshark. [Online]. Available: http://www.wireshark.org/.

[25] J. Chen and Lj. Trajković, "Analysis of Internet topology data," in *Proc. International Symposium on Circuits and Systems (ISCAS' 04)*, Vancouver, British Columbia, Canada, May 2004, vol. 4, pp. 629–632.

[26] (2013) BCNET. [Online]. Available: https://wiki.bc.net.

[27] T. Farah, S. Lally, R. Gill, Al-Rousan, R. Paul, D. Xu, N., and Lj. Trajković, "Collection of BCNET BGP traffic," in *Proc. 23rd International Teletraffic Congress (ITC' 11)*, San Francisco, CA, Sept. 2011, pp. 322–323.

[28] S. Lally, T. Farah, R. Gill, R. Paul, N. Al-Rousan, and Lj. Trajković, "Collection and characterization of BCNET BGP traffic," in *Proc. 2011 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PacRim' 11)*, Victoria, BC, Canada, Aug. 2011, pp. 830–835.

[29] T. Viipuri, "Traffic Analysis and Modeling of IP Core Networks," M.S. Thesis, Electrical and Communication Engineering, Aalto University, Espoo, Finland, Dec. 2004.

[30] (2013) ENDACE. [Online]. Available: http://www.endace.com.

[31] E. Bosch and B. Trammell, "IP flow anonymization support," May 2011. [Online]. Available: http://tools.ietf.org/html/rfc6235.

[32] (2013) RIJNDAEL Advanced Encryption Standard. [Online]. Available: http://www.cs.mcgill.ca/~kaleigh/computers/crypto_rijndael.html.

[33] J. Fan, J. Xu, M. H. Ammar, and S. B. Moon, "Prefix-preserving IP address anonymization: measurement-based security evaluation and a new cryptography-based scheme," *Computer Network*, Oct. 2004, vol. 46, pp. 253–272.

[34] (2013) Lucent's extensions to cryptopan. [Online]. Available: http://www.cc.gatech.edu/computing/Networking/projects/cryptopan/lucent.shtml/.

[35] (2013) Anonymization Application Programming Interface (AAPI). [Online]. Available: http://www.ics.forth.gr/dcs/Activities/Projects/anontool.html.

[36] (2013) FLAIM. [Online]. Available: http://en.wikipedia.org/wiki/FLAIM.

[37] (2013) FLAIM (framework for log anonymization and information management). [Online]. Available: http://flaim.ncsa.illinois.edu/.

[38] (2013) Kolmogorov–Smirnov. [Online]. Available: http://www.itl.nist.gov/div898/handbook/eda/section3.

[39] (2013) Kolmogorov–Smirnov test. [Online]. Available: http://www.physics.csbsju.edu/stats/KS-test.html.

[40] (2013) Principles of Statistics. [Online]. Available: http://www.graphpad.com/guides/prism/6/statistics/.

[41] M. Fras, J. Mohorko, and Z. Cucej, "A new goodness of fit test for histograms regarding network traffic packet size process," in *Proc. International Conference on Advanced Technologies for Communications (ATC 2008)*, Hanoi, Vietnam, Oct. 2008, pp.345–348.

[42] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," in *Proc. IEEE/ACM Transactions on Networking*, Feb. 1994, vol.2, no.1, pp.1–15.

[43] M. Fras, J. Mohorko, and Z. Cucej, "Packet size process modeling of measured self-similar network traffic with defragmentation method," in *Proc. 15th International Conference on Systems, Signals and Image Processing (IWSSIP 08)*, Bratislava, Slovakia, June 2008, pp. 253–256.

[44] M. Foukarakis, D. Antoniades, S. Antonatos, and E. P. Markatos, "On the anonymization and deanonymization of netflow traffic," in *Proc. Conference for Network Flow Analysis (FloCon' 08)*, Savannah, GA, USA, Jan. 2008, pp. 272–275.

[45] M. Peuhkuri, "A method to compress and anonymize packet traces," in *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement (IMW' 01)*, San Francisco, CA, USA, Nov. 2001, pp. 257–261.

[46] S. E. Coull, M. P. Collins, C. V. Wright, F. Monrose, and M. K. Reiter, "On web browsing privacy in anonymizaed netflows," in *Proc. 16th USENIX Security Symposium (SS' 07)*, Boston, MA, USA, Aug. 2007, pp. 1–14.

[47] M. Atallah, A. Elmagarmid, M. Ibrahim, E. Bertino, and V. Verykios, "Disclosure limitation of sensitive rules," in *Proc. 1999 Workshop on Knowledge and Data Engineering Exchange*, Washington, DC, USA, Nov. 1999, pp. 45–52.

[48] M. Jiang, M. Nikolić, S. Hardy, and Lj. Trajković, "Impact of self-similarity on wireless network performance," in *Proc. IEEE International Confenence on Communications (ICC' 01)*, Helsinki, Finland, June 2001, pp. 477–481.

[49] Lj. Trajković, "Analysis of Internet topologies," *Circuits and Systems Magazine*, Sept. 2010, vol. 10, no. 3, pp. 48–54.

[50] M. Najiminaini, L. Subedi, and Lj. Trajković, "Analysis of Internet topologies: a historical view," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS' 09)*, Taipei, Taiwan, May 2009, pp. 1697–1700.

[51] S. Lau and Lj. Trajković, "Analysis of traffic data from a hybrid satellite-terrestrial network," in *Proc. Fourth Int. Conf. on Quality of Service in Heterogeneous*

Wired/Wireless Networks (QShine' 2007), Vancouver, BC, Canada, Aug. 2007, pp. 9:1–9:7.

[52] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," in *Proc. 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM' 05)*, NY, USA, Aug. 2005, vol. 35, pp. 217–228.