Simpleweb/University of Twente Traffic Traces Data Repository

Rafael Ramos Regis Barbosa[1], Ramin Sadre[1], Aiko Pras[1], and Remco van de Meent[2]

[1]University of Twente
Design and Analysis of Communication Systems (DACS)
Enschede, The Netherlands
[2]Vodafone NL

April 29, 2010

Abstract

The computer networks research community lacks of shared measurement information. As a consequence, most researchers need to expend a considerable part of their time planning and executing measurements before being able to perform their studies. The lack of shared data also makes it hard to compare and validate results. This report describes our efforts to distribute a portion of our network data through the *Simpleweb/University of Twente* Traffic Traces Data Repository.

1 Introduction

Performing useful network measurements can be a difficult task. Problems can appear in different parts of the procedure, ranging from gaining permission to start the measurements, to power failures causing the monitoring equipment to fail. In addition, even when researchers succeed in their measurements, it might be that they fail to observe the phenomena they want to study, e.g. a SSH brute-force attack when studying intrusion detection or network congestion when validating a new QoS scheme.

The difficulties to perform measurements alone is a valid reason to support the distribution of such data, but there is another, arguably more important, reason. Sharing data used in experiments enable researchers to validate results and compare methods. This is one of the basis of well-grounded scientific work.

The request for sharing data is a constant in the computer networks research community (e.g. [1–3]). This report describe our efforts to share our traffic measurement data through the *Simpleweb/University of Twente* repository. Currently it contains eight datasets consisting of packet headers, Netflow version 5 data and a labeled dataset for flow-based intrusion detection. The repository is available at this location: `http://traces.simpleweb.org/`.

Other databases of network measurement are available. *DatCat* [4], *MOME* [5], *Internet Traffic Archive* [6] and *MAWI Working Group Traffic Archive* [7]

are examples of well-known network traffic repositories. The *UMass Trace Repository* [8] contains other traces such as *CPU* and memory, and storage traces, in addition to network traces. RIPE NCC Data Repository [9] contains measurements collected by RIPE NCC projects, packet trace sets recovered from the defunct NLANR website and, datasets collected and currently hosted by other research institutions. Some of these datasets are described in [10]. Some of these databases have specialised information, for instance, the *CRAWDAD* [11] focuses on wireless and mobility data while *PREDICT* [12] focuses on data for cyber security research.

This report updates the information provided in [13]. The remainder of this report describes the situation of the repository as from April 2010. Section 2 presents the measurement procedure and the subsequent steps taken to protect the network users' privacy. In Section 3 we describe datasets individually, providing some contextual information of each of the data collection scenarios.

2 Measurements

In this section we detail some points of our approach for data collection and *anonymization* for packet headers, Netflow version 5 data and the labeled dataset for intrusion detecton.

2.1 Packet Headers

The measurements are performed by capturing the headers of all packets that are transmitted over the (Ethernet) "uplink" of an access network to the Internet, as outlined in Figure 1. The switch (or router) copies all traffic flowing in to and out of the access network to the measurement PC. Even an ordinary PC should have no problem in handling a load on the uplink up to several hundred Mbit/s [14]. To capture packets the standard tcpdump [15] utility is used.

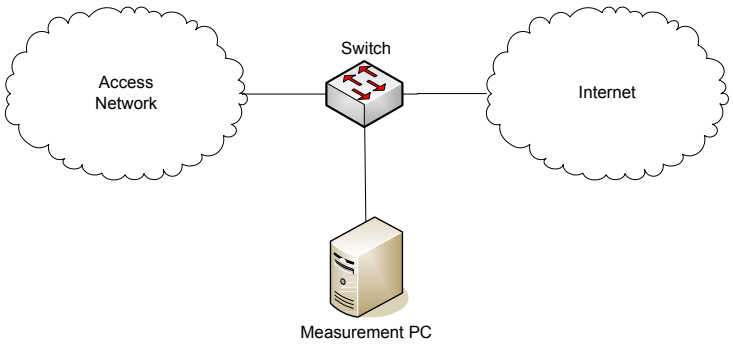

Figure 1: Packet Header measurement setup

The packets' payload contain sensitive information, such as replies from HTTP requests or content of Instant Messaging (IM) conversations, so, to preserve the users' privacy, this information is not included in our traces. We set tcpdump to capture the first 64 octets of each Ethernet frame. Only headers from the link up to the transport layer are captured, whilst the packet's payload

is ignored. This information is dumped to a binary file that is stored on disk. The resulting packet trace is a file of possibly several gigabytes, depending on the load of uplink. In order to save resources, the traces are compressed. This saves on average some 60% in disk space.

The headers in the packet trace include source and destination IP addresses and port numbers. Although the payload of the IP packets is discarded, careful analysis of the packet trace still may reveal possibly sensitive information, such as which websites are visited by who, which threatens users' privacy. On the other hand, the removal of addresses from the packet traces would severely reduce their usefulness. Thus, there is a trade-off to be made between protecting privacy and usability of the traces.

To protect users' privacy, the packet headers are made anonymous by scrambling the source and destination IP addresses. Other information, such as transport port numbers and the timestamps at which packets arrive are left unchanged. To this end we used the tcpdpriv [16] utility with following options:

```
tcpdpriv -A50 -P99 -r original-packet-trace -w anonymized-packet-trace
```

The "-A50" option enables the prefix-preserving anonymization. This means that, within a single packet trace, if two of the original addresses are equal in the most significant n bits, then these two addresses will map to scrambled addresses that are similarly equal in the most significant n bits. For example, if source address $a.b.c.p$ is mapped to $x.y.z.k$, then $a.b.c.q$ is mapped to $x.y.z.l$. A possible drawback of this approach, however, is that some topological information might be revealed, whereas strict random mapping would not.

The effect of the "-P99" option is that transport port numbers are unchanged, e.g., if an original packet was sent from TCP port 1025 to port 80 (i.e., web-browsing), the same port numbers will be stored in the anonymized packet trace.

2.2 Netflow Version 5

The measurement setup for Netflow version 5 data is depicted on Figure 2. The Netflow data is recorded in the access router that connects a university to its Internet Service Provider (ISP). Most of the university's traffic incoming and outgoing traffic is routed through this link. Also some internal traffic is recorded, as this access router also interconnects some of the university's subnets. No sampling was performed in the measurement i.e., all packet are processed in the access router to compute the Netflow data.

The data recorded at the access router is sent to a Netflow collector, that stores the data for further analysis. The data we make available is a tcpdump capture of the packets sent from the access router to the Netflow collector.

Following the same reasoning from Section 2.1, we decide to protect users' privacy by anonymizing the IP addresses present in the Netflow data, while leaving transport port numbers and timestamps unchanged. Remember that the Netflow data is stored in tcpdump/libpcap format, therefore, besides the three IP addresses present in the Netflow each record (source, destination and next hop fields), we have to anonymize the source and destination IP addresses present in each packet.

To this end we use the AnonTool, an open-source implementation of Anonymization API [17]. AnonTool contains two separate applications to

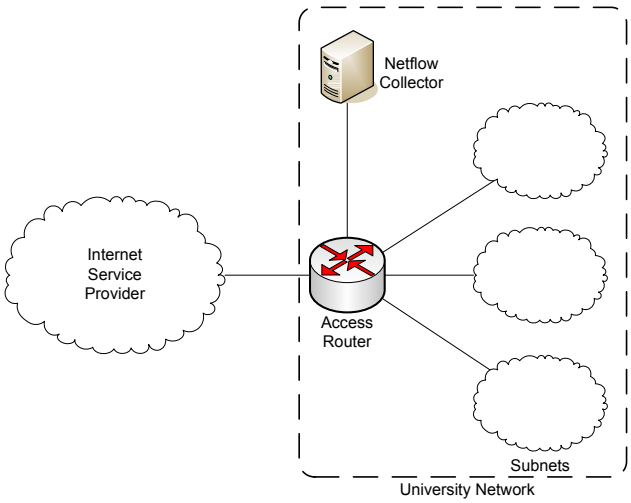

Figure 2: Netflow measurement setup

anonymize information in the packet headers and in the Netflow records. This means that, for anonymizing all IP addresses it would be necessary to process the data twice, one with each application. To avoid that, we decided to code a small application that makes use of the Anonymization API directly, anonymizing all IP addresses in one pass.

The main functions used are showed in the *C* code snippet in Figure 3. The function "PREFIX_PRESERVING", which is based on the Crypto-PAn implementation [18], is used in all IP addresses. This function performs a prefix-preserving anonymization based on a cryptographic key. An advantage of this method over the *tcpdpriv* method, is that the mapping is consistent across multiple traces, as long as the same key is used. As this function changes values in the IP header and payload, the checksum calculation needs to be corrected. This is done using the function "CHECKSUM_ADJUST", showed on the last lines of the code snippet.

2.3 Labeled Dataset for Intrusion Detection

For the construction of this dataset a single host, referred to as honeypot, is monitored. A honeypot can be defined as an "environment where vulnerabilities have been deliberately introduced to observe attacks and intrusions" [19]. The honeypot was installed on a virtual machine running on Citrix XenServer 5 [20], running a Linux distribution (Debian Etch 4.0). In particular it had the following services installed:

- **ssh:** OpenSSH service [21]. Beside the traditional service logs, the OpenSSH service running on Debian has been patched in order to log sessions: for each login, the transcript (user typed commands) and the timing of the session have been recorded. This patch is particularly important to track active hacking activities.

- **Apache web server:** a simple webpage with a log in form has been deployed. We relied on the service logging capabilities for checking the content of the incoming http connections.

- **ftp:** The chosen service was proftp [22]. As for http, we relied on the ftp logs for monitoring attempted and successful connections. proftp uses the auth/ident service running on port 113 for additional authentication information about incoming connections.

As the other scenarios, the privacy sensitive information is also anonymized. Here the Crypto-PAn [18] application was used. It is important to note that this method does not preserve the ordering of the addresses, so the flows of the scans may not appear sequentially in the database.

The labeled dataset is provided as a gzipped SQL script, generated from a MySQL database. Further information on the data collection and labelling can be found in [23].

3 Repository

The Simpleweb/University of Twente Traffic Traces Data Repository is available at this location:

`http://traces.simpleweb.org/`

In the remainder of this section we describe the eight scenarios present in the repository as from April 2010, to give the context of the traces.

```
//anonymize IP addresses
add_function(sd, "ANONYMIZE",
    IP, SRC_IP, PREFIX_PRESERVING);
add_function(sd, "ANONYMIZE",
    IP, DST_IP, PREFIX_PRESERVING);

//anonimize IP addresses in Netflow
add_function(sd,"ANONYMIZE",NETFLOW_V5,
    NF5_SRCADDR,PREFIX_PRESERVING);
add_function(sd,"ANONYMIZE",NETFLOW_V5,
    NF5_DSTADDR,PREFIX_PRESERVING);
add_function(sd,"ANONYMIZE",NETFLOW_V5,
    NF5_NEXTHOP,PREFIX_PRESERVING);

//fix checksums
add_function(sd, "ANONYMIZE",
    IP, CHECKSUM, CHECKSUM_ADJUST);
add_function(sd, "ANONYMIZE",
    UDP, CHECKSUM, CHECKSUM_ADJUST);
```

Figure 3: Anonymization code snippet using Anonymization API

3.1 Trace 1 - Packet Headers

In scenario 1, the 300 Mbit/s (a trunk of 3 x 100 Mbit/s) ethernet link has been measured, which connects a residential network of a university to the core network of this university. On the residential network, about 2000 students are connected, each having a 100 Mbit/s ethernet access link. The residential network itself consists of 100 and 300 Mbit/s links to the various switches, depending on the aggregation level. The measured link has an average load of about 60%. Measurements have taken place in July 2002.

3.2 Trace 2 - Packet Headers

In the second scenario, the 1 Gbit/s ethernet link connecting a research institute to the Dutch academic and research network has been measured. There are about 200 researchers and support staff working at this institute. They all have a 100 Mbit/s access link, and the core network of the institute consists of 1 Gbit/s links. The measured link is only mildly loaded, usually around 1%. The measurements are from May - August 2003.

3.3 Trace 3 - Packet Headers

This dataset was collected in a large college. Their 1 Gbit/s link (i.e., the link that has been measured) to the Dutch academic and research network carries traffic for over 1000 students and staff concurrently, during busy hours. The access link speed on this network is, in general, 100 Mbit/s. The average load on the 1 Gbit/s link usually is around 10-15%. These measurements have been done from September - December 2003.

3.4 Trace 4 - Packet Headers

In scenario 4, the 1 Gbit/s aggregated uplink of an ADSL access network has been monitored. A couple of hundred ADSL customers, mostly student dorms, are connected to this access network. Access link speeds vary from 256 kbit/s (down and up) to 8 Mbit/s (down) and 1 Mbit/s (up). The average load on the aggregated uplink is around 150 Mbit/s. These measurements are from February - July 2004.

3.5 Trace 5 - Packet Headers

The dataset *Packet Headers 5* was collected in a hosting-provider, i.e. a commercial party that offers floor- and rack-space to clients who want to connect, for example, their WWW-servers to the Internet. At this hosting-provider, these servers are connected at (in most cases) 100 Mbit/s to the core network of the provider. The bandwidth capacity level of this hosting-provider's uplink (that we have measured) is around 50 Mbit/s. These measurements are from December 2003 - February 2004.

3.6 Trace 6 - Packet Headers

In scenario 6, a 100 Mbit/s Ethernet link connecting an educational organization to the internet has been measured. This is a relatively small organization

with around 35 employees and a ittle over 100 students working and studying at this site (the headquarter location of this organization). All workstations at this location (100 in total) have a 100Mbit/s Lan connection. The core network consists of a 1 Gbit/s connection. The recordings took place between the external optical fiber modem and the first firewall. The measured link was only mildly loaded during this period. These measurements are from May - June 2007.

3.7 Trace 7 - Netflow Data

The Netflow version 5 data was recorded in the access router connecting a university to its ISP. It contains flow information about most of the incoming and outgoing university's traffic and some internal traffic as well. The traces cover a period of time of two working days, namely between Wednesday August 1st 2007, 00:00 and Thursday August 2nd 2007, 23:59. The university has /16 network providing connectivity to the employees and the students on its buildings and the campus. The university is connected to its ISP through a 10 Gbps optical link with an average load of 650 Mbps and peaks up to 1.0 Gbps.

3.8 Trace 8 - Labeled Dataset for Intrusion Detection

In this scenario, a honeypot (running in a virtual machine) was monitored for 6 days, from Tuesday 23 September 2008 12:40:00 GMT to Monday 29 September 2008 22:40:00 GMT. The honeypot was hosted in the University of Twente network and directly connected to the Internet. The monitoring window is comprehensive of both working days and weekend days. The data collection resulted in a 24 GB dump file containing 155.2 million packets. The processing of the dumped data and logs, collected over a period of 6 days, resulted in 14.2M flows and 7.6M alerts.

Acknowledgements

This research work has been supported by the EC IST-EMANICS Network of Excellence (#26854) and by the Istrice research program (http://www.ctit.utwente.nl/research/sro/istrice/).

References

[1] M. Alllman and V. Paxson. Issues and Etiquette Concerning Use of Shared Measurement Data. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, page 140. ACM, 2007.

[2] V. Paxson. Strategies for Sound Internet Measurement. In *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, pages 263–271. ACM New York, NY, USA, 2004.

[3] KC Claffy, M. Crovella, T. Friedman, C. Shannon, and N. Spring. Community-Oriented Network Measurement Infrastructure (CONMI) Workshop Report. *ACM Computer Communication Review*, 32(2):41–48, April 2006.

[4] C. Shannon, D. Moore, K. Keys, M. Fomenkov, and B. Huffaker. The Internet Measurement Data Catalog. *ACM SIGCOMM Computer Communication Review*, 35(5):100, 2005.

[5] Cluster of European Projects aimed at Monitoring and Measurent (MOME). http://www.ist-mome.org/.

[6] V. Paxson. Internet Traffic Archive. http://ita.ee.lbl.gov/.

[7] MAWI Working Group Traffic Archive. http://tracer.csl.sony.co.jp/mawi/.

[8] UMass Trace Repository. http://traces.cs.umass.edu/.

[9] RIPE NCC Data Repository. https://data-repository.ripe.net/.

[10] Tony McGregor, Shane Alcock, and Daniel Karrenberg. The RIPE NCC Internet Measurement Data Repository. *Passive and Active Measurement*, pages 111–120, 2010///.

[11] Community Resource for Archiving Wireless Data At Dartmouth (CRAWDAD). `http://crawdad.cs.dartmouth.edu/`.

[12] Protected Repository for the Defense of Infrastructure against Cyber Threats (PREDICT). http://www.predict.org/.

[13] R. van de Meent. M2C Measurement Data Repository. *University of Twente, Enschede, The Netherlands, M2C Deliverable D15*, 1, 2003.

[14] R. Poortinga, R. van de Meent, and A. Pras. Analysing Campus Traffic Using the Meter-MIB. In *Proceedings of Passive and Active Measurement Workshop 2002*, pages 192–201. Citeseer, 2002.

[15] tcpdump/libpcap. `http://www.tcpdump.org/`, .

[16] tcpdpriv. `http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html`, .

[17] D. Koukis, S. Antonatos, D. Antoniades, E. Markatos, and P. Trimintzios. A Generic Anonymization Framework for Network Traffic. In *Proceedings of the IEEE International Conference on Communications (ICC 2006)*. Citeseer, 2006.

[18] 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 Networks*, 46(2):253–272, 2004.

[19] F. Pouget and M. Dacier. Honeypot-based Forensics. In *AusCERT Asia Pacific Information Technology Security Conference*, 2004.

[20] Citrix. http://www.citrix.com/.

[21] OpenSSH. http://www.openssh.com/.

[22] proftp. http://www.proftpd.org/.

[23] A. Sperotto, R. Sadre, D. F. van Vliet, and A. Pras. A Labeled Data Set For Flow-based Intrusion Detection. In *Proceedings of the 9th IEEE International Workshop on IP Operations and Management, IPOM 2009, Venice, Italy*, volume 5843 of *Lecture Notes in Computer Science*, pages 39–50. Springer Verlag, October 2009.