

Traffic Measurements for Link Dimensioning

A Case Study

Remco van de Meent, Aiko Pras, Michel Mandjes,
Hans van den Berg, and Lambert Nieuwenhuis

University of Twente
PO Box 217
7500 AE Enschede
The Netherlands
`r.vandemeent@utwente.nl`

Abstract. Traditional traffic measurements meter throughput on time scales in the order of 5 minutes, e.g., using the Multi Router Traffic Grapher (MRTG) tool. The time scale on which users and machines perceive Quality of Service (QoS) is, obviously, orders of magnitudes smaller. One of many possible reasons for degradation of the perceived quality, is congestion on links along the path network packets traverse. In order to prevent quality degradation due to congestion, network links have to be dimensioned in such a way that they appropriately cater for traffic bursts on time scales similarly small to the time scale that determines perceived QoS. It is well-known that variability of link load on small time scales (e.g., 10 milliseconds) is larger than on large time scales (e.g., 5 minutes). Few quantitative figures are known, however, about the magnitude of the differences between fine and coarse-grained measurements. *The novel aspect of this paper is that it quantifies the differences in measured link load on small and large time scales.* The paper describes two case studies. One of the surprising results is that, even for a network with 2000 users, the difference between short-term and long-term average load can be more than 100%. This leads to the conclusion that, in order to prevent congestion, it may not be sufficient to use the 5 minute MRTG maximum and add a small safety margin.

1 Introduction

Some believe that QoS in networks will be delivered by the use of technologies such as DiffServ and IntServ, which ensure the “right” allocation of available resources among different requests. Concerns about deployment, operational complexity and, in the case of IntServ, scalability, give rise to alternative approaches of providing QoS. One of these alternatives is overprovisioning [1]. The idea behind overprovisioning is to allocate so many resources that users no longer experience a QoS improvement in case additional resources get allocated. This paper focuses on the bandwidth overprovisioning of individual network links. An example of such link is the access line between an organization’s internal network and its Internet Service Provider.

Since bandwidth may be expensive, managers who rely on overprovisioning as mechanism for delivering QoS need to know the amount of traffic that users of a specific link may generate at peak times. To find this figure, managers usually use tools like MRTG [2]. Such tools are able to measure the average load of network links by reading the Interface Group MIB counters every 5 minutes (by default), and plot the results in a graph. The peaks in these graphs, plus a certain safety margin, or often used to dimension the specific link.

In cases where overprovisioning is used as mechanism to provide QoS, load averages of 5-minutes may not be adequate to properly dimension network links. With web browsing, for example, traffic is exchanged in bursts which last between parts of a second and several seconds. If, within these seconds, the link gets congested, the user will not perceive an acceptable QoS. Also distributed computer programs, which interact without human intervention, “perceive” QoS on time scales smaller than seconds. The traditional 5 minute figures of MRTG do not give any insight in what happens on these small time scales. It is therefore important to increase of the *time-granularity* of the measurements, i.e., to decrease the size of the time-window that is used to determine the link load. To overprovision, load figures are needed on the basis of seconds, or even less.

1.1 Contribution

The goal of this paper is to quantify the differences in measured link load at various time scales. For this purpose, two case studies have been performed. In the first study traffic was measured on the external link of a university’s residential network; this link was used by thousands of students. Because of this high number, we expected that the differences between the long- and short-term averages would be relatively small. To get some stronger differences, the second study was performed on the access line of a small hosting provider, which served some tens of customers.

The outcome of the measurements came as a surprise. On the university’s link, with thousands of users, the differences between long- and short-term averages could be more than hundred percent. In the case of the hosting provider, the differences could even be thousands of percents.

1.2 Organization

The remainder of this paper is organized as follows. Section 2 describes the two networks on which the measurements were performed, the measurement equipment that was used, as well as the way this equipment was connected. Section 3 discusses some factors that influence the achievable granularity of our measurements. Section 4 discusses the tools that were used for processing, analyzing and visualizing the gathered data. Section 5 presents the analysis of the measurements and determines the ratio of peak versus average throughput on the links of our case studies. The conclusions are provided in Section 6.

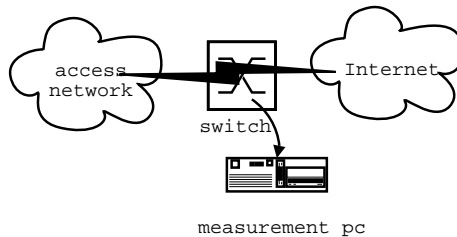


Fig. 1. Measurement Setup

2 Measurement Setup

In this study we have concentrated on network technologies that are common in both production and academic environments. We assume such networks to consist of (normal, Fast and Gigabit) Ethernet links, which are connected by hubs, switches and routers. In general it is important that normal network operation should not be disturbed by the measurements. For example, it may not be acceptable to interrupt network operation to install optical splitters that copy all network traffic to a measuring device. It will often be better to rely on the “mirror” facilities provided by current middle- to high-end switches, and copy all traffic that should be monitored to one of the free interfaces of that switch. The measurement device can then be connected to that interface, and all traffic can be analyzed without disturbing the network. See Figure 1 for a schematic representation of the measurement setup.

To validate whether it is practically feasible to measure throughput on small time scales, and to compare traditional **MRTG** measurements to throughput measurements with fine granularity, two case studies have been performed.

The first case study measures throughput on a backbone link that is used by thousands of users. With this large number of users it is expected that the traffic generated by a single user will have limited impact on the traffic aggregate; it is therefore expected that the difference in “long-term” and “short-term” throughput averages will be relatively low. The second case study measures throughput on a link that is used by some tens of users. With this small number of users it is anticipated that the results will differ much more.

2.1 Campus Network

The Campusnet [3], a residential network of the University of Twente, connects about 2000 students to the Internet. Each student has a 100 Mbit/s full duplex connection to the network. The network is hierarchically structured, and has a full-duplex 300 Mbit/s backbone link to the rest of the world. This link, with a 5-minute average load of about 50%, has been monitored in the first study. The question whether or not this link is overprovisioned, is hard to answer beforehand, given the expected variability of the throughput on smaller time scales.

2.2 Hosting Provider Network

The second case study involves the network of a small hosting company in The Netherlands. This network connects some tens of customers over a switched 100 Mbit/s network to the Internet. The link to the Internet, with a 5-minute average load of 5–10%, has been monitored in the second study. As this link is only mildly loaded, we anticipate that it can be regarded as being overprovisioned.

2.3 Measurement Device

An important requirement for this study is to use as much as possible common hardware and software. For the measurement device it was decided to take an off-the-shelf PC; the software was based on Linux. The details of the measurement device are shown in Table I.

Table 1. Measurement PC Configuration

Component	Specification
CPU	Pentium-III 1 GHz
Mainboard	Asus CUR-DLS (64 bit 66 MHz PCI)
Hard disk	60 + 160 Gigabyte, UDMA/66
Operating system	Debian Linux, 2.4.19-rc1 kernel
Network interface	1 x Gbit/s Intel Pro/1000T
Main memory	512 MB reg. SDRAM

The advantage of using off-the-shelf hardware and commodity software, is the low price of the measurement device and the simplicity of the software installation and maintenance. A potential drawback is the possible poor performance and scalability. In particular problems may occur because of limited CPU and I/O (e.g., network, disk) speed. To avoid such problems, we selected a Gigabit Ethernet card and a motherboard with a 64 bit bus. It turned out that this PC could easily capture hundreds of Mbit/s.

If, in future measurements, performance and scalability becomes problematic, it should be possible to migrate to some alternative measurement setup, such as TICKET [4], NG-MON [5] or IPMon [6].

There are also hardware based solutions for very fine-grained measurements, such as the DAG cards [7], but these are very expensive. One could also argue that measurements on time scales smaller than 10 milliseconds, as offered by alternative solutions, do not add to the notion of perceived QoS.

3 Time-Granularity

The choice for a specific measurement device and setup imposes certain constraints on the achievable time scales for metering. In this section we will discuss

four factors that determine which time granularity is practical for our specific measurement setup.

A first factor is that the switch should copy all traffic from the ports to be monitored to the port to which the measurement device is connected. This switch, like all network devices, contains buffers to temporarily store packets that cannot be transmitted immediately. The switch that was used for our case studies was able to buffer frames for some tens of milliseconds. Since the capacity of the link that connected the switch to our measurement device could easily handle all monitored traffic, delays remained short and buffer overflow did not occur.

A second factor plays a role whenever a full-duplex link is monitored. On such links packets flow in two directions. If both directions of traffic should be copied to a single outgoing link, delays are introduced since only one packet can be copied at a time. As long as the link to the measurement device supports at least twice the speed of the full-duplex link, the delay will be less than the time it takes to transmit a single packet. For Gigabit Ethernet links, this delay will be a fraction of a millisecond.

A third factor is that it takes some time between forwarding a packet within the switch, and timestamping the packet within the measurement device. Without special equipment, it is hard to exactly determine this time, but it is at most a fraction of a millisecond, and the same for all packets.

The fourth factor, which is the most important one, is the resolution of the timestamp itself. This resolution depends on the software that runs on the measurement device. With standard Linux/x86, which was used in our case studies, this resolution is 10 milliseconds (100 Hz). For each packet that the network card delivers to the Linux kernel, a timestamp is added by the network driver in the `netif_rx()` routine. The resolution can be improved by the use of the Time Stamp Counter (TSC) feature that is present on modern x86 CPUs. It is also expected that the granularity can be further improved by using the `mmtimer` feature that will probably be around in future chipsets. This feature is designed for multimedia purposes, but can be used for measurement purposes as well. Both TSC and `mmtimer` give nanosecond time precision, but are not fully used in current Linux kernels.

From the above considerations it can be concluded that a safe value for the reasonably achievable time-granularity is 10 ms (100 Hz). In the next sections we will use this value as the minimum measurement interval.

4 Measurement Tools

The measurement process can be divided into three stages: 1) capturing, 2) processing and analyzing, and 3) visualizing network traffic. These steps are described in more detail below.

For our case studies we chose to process, analyze and visualize the gathered data “off-line”, i.e., network data is stored to disk, and further processing is done afterwards. The reason to do this “off-line”, is that we want to experiment with

analyzing and visualizing the data. In principle, however, throughput-evaluation can be done “on-line”, e.g., by using simple counters.

4.1 Capturing

The measurement PC receives all traffic from the monitored network link via a Gigabit Ethernet interface. The traffic is captured using `tcpdump` [8] and the associated `libpcap` library. For the throughput analysis performed in this study, it is not necessary to store the complete contents of all frames; for our purpose it is sufficient to store only the first 66 octets of each frame. These octets contain all header information up to the transport layer (e.g., TCP port numbers, if available); this is sufficient to perform, for example, flow arrival analysis. To improve performance and to avoid potential privacy problems, the rest of the payload is ignored. For each frame that is captured, `libpcap` also adds to the capture file: 1) a timestamp (as provided by the Linux kernel), 2) the size of the captured fraction of the frame (i.e., 66 in this study), and 3) the total size of the original frame. Note that capturing results in a huge amount of data, particularly on high-speed networks; a measurement period of 15 minutes can involve multiple gigabytes of data. To handle capture files bigger than 2 gigabytes, it is important to compile `tcpdump` and `libpcap` with Large File Support enabled.

4.2 Processing and Analysis

The second step involves preparation for, and the actual analysis of, the gathered data. In studies that focus on throughput, this step is relatively simple: anonymization of the data (if required) and grouping of the captured packets according to the required granularity.

Although from a technical point not a required part of the measurement process, anonymization should be done for privacy reasons. Various tools, implementing different anonymization schemes, are around. In this study the `tcpdpriv` [9] utility has been used, which can be configured for different levels of protection (scrambling of only parts of the IP address, scrambling of transport port numbers, etc.).

For throughput analysis, all packets in the capture file should be grouped. Every group consist of all the packets captured within a certain time interval. In accordance with the discussion in Section III, we chose the minimum time interval to be 10 milliseconds. Depending on the network load, a single group can consist of hundreds of packets. Obviously, the throughput of each interval can be calculated by summing up the sizes of all packets within the associated group, and dividing the resulting number by the length of the time interval (10 ms).

4.3 Visualization

From the interim-results of the analysis step, graphs can be plotted. In our case studies, the `GD` [10] library has been used to create images, using `Perl` scripts.

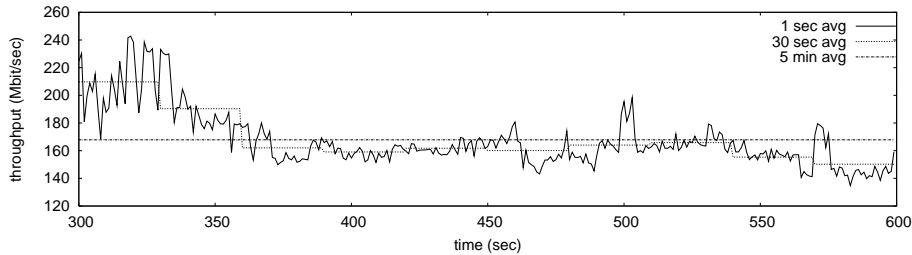


Fig. 2. Campusnet - Time-Granularity < 5 min

A problem with visualization is that the amount of information may be too large to properly display in a single graph. Therefore a reduction may be required, e.g., by plotting only the highest 10 ms average throughput value of a longer, for example 10 seconds time interval. The analysis and visualization tools that have been developed as part of this research, are available online from [11].

5 Measurement Results

The main goal of this study is to get quantitative figures showing the difference between the traditional 5 minutes traffic measurements of MRTG, and finer-grained measurements with time scales up to 10 ms. This section presents the results.

Figure 2 shows, for the Campusnet scenario, the difference between common MRTG statistics and measurements on smaller time scales. The time-granularity is increased from 5 minute throughput averages, to 30 second averages and finally 1 second averages. Note that the 5 minute average is around 170 Mbit/s. From the picture it is clear that, within that interval, the average throughput in the first minute is considerably higher than the 5 minute average. This is true for both the 30 seconds as well as the 1 second averages. Some of the measured 1 second average throughput values are even 40% higher than the traditional 5 minute average value. It should be noted that all measurements span 15 minutes; for visualization reasons, the graphs show only part of that interval.

Figure 3 zooms in on the first half second of the measurement of Figure 2. Time-granularity is further increased from 1 second, to 100 ms and finally 10 ms. It should be noted that each 10 millisecond interval still contains hundreds of packets. The graph shows that the 100 ms averages are relatively close to the 1 second average throughput. This is not a general rule, however; other measurements on the same network have shown differences of up to tens of percents. It is interesting to see spikes of over 300 Mbit/s for the 10 ms averages – almost twice the value of the 5-minute average. Note that the figure shows the aggregation of traffic flowing in and out of the Campusnet, hence the possibility of values higher than 300 Mbit/s.

Figure 4 shows throughput statistics for the hosting provider’s network. The utilization of this network is relatively low, with some tens of concurrent users

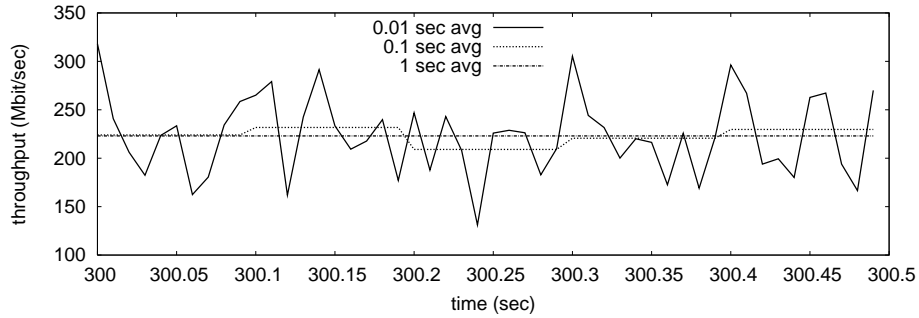


Fig. 3. Campusnet - Time-Granularity: < 1 sec

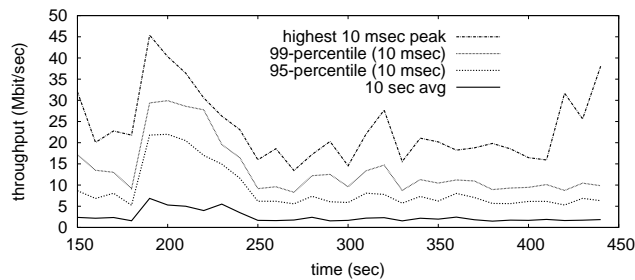


Fig. 4. Hosting Provider's Network: Average, Peak and Percentiles

at the most. The 5 minute average throughput (not in the graph) for this interval is around 2 Mbit/s. The lowest line shows the average throughput with a time-granularity of 10 seconds. The figure shows that for the 190th till 250th second, the 10 second average throughput values are a multiple of the traditional 5 minute average. The top line in Figure 4 shows the highest 10 ms average within each 10 second interval – thousands of percents higher than the 5 minute average. The other two lines are the 95th and 99th percentile of all 10 ms averages within each 10 second interval. These percentiles are regarded to be a better performance measure than the absolute peak value, as they better describe user-perceived QoS. Also these values are considerably higher than the longer term average throughput. An explanation for the huge differences in “short-term” and “long-term” averages, is the combination of a small number of users, and the high speed at which a single user can send – a modern server can easily saturate a 100Mbit/s LAN. Hence a single user can have, when he sends traffic, a big impact on the traffic aggregate.

To compare between the hosting provider's network and the Campusnet, Figure 5 shows for the Campusnet scenario the same kind of information as Figure 4. Note that the 95/99-percentiles of the 10 ms measurements are also well above the “long-term” averages. The (relative) fluctuations, compared to the longer time averages are less, however, than for the hosting provider's network. This is

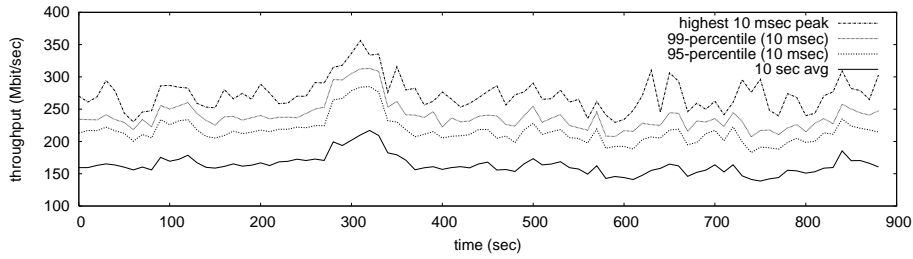


Fig. 5. Campusnet: Average, Peak and Percentiles

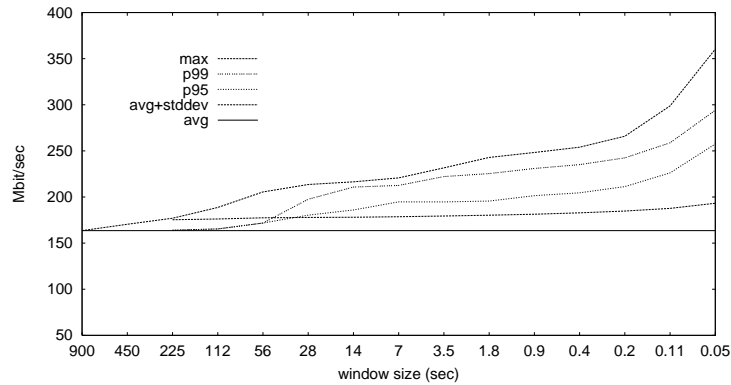


Fig. 6. Campusnet: Throughput vs. Time Window

probably caused by the fact that an increase in the number of concurrent users and related link load, in general leads to a decrease in burstiness of the traffic aggregate [12]. But still, the percentiles of the 10 millisecond measurements are tens of percents higher than the 10 second averages. It is important to take this into account while dimensioning a network.

In order to get a better idea on how the (peak) throughput averages increase with a decreasing time window size, we split the measurement data (900 seconds) in partitions, see Figure 6 for the Campusnet scenario, and Figure 7 for the hosting provider scenario. After 0 splits, the average throughput of the entire 900 seconds time window is plotted. After 1 split, we look at the first 450 seconds and the second 450 seconds time window, after 2 splits, we have 4 windows of 225 seconds, etc. After 14 splits, we have time windows of approximately 50 milliseconds. For each different window size, we can now determine the maximum throughput of all windows of a certain size, the standard deviation, and the 95/99-percentiles.

It is obvious that the maximum (average) throughput increases when the time-window size is split in half. It can also be seen, by looking at the deviation plots, that the fluctuations increase when the time window size becomes smaller.

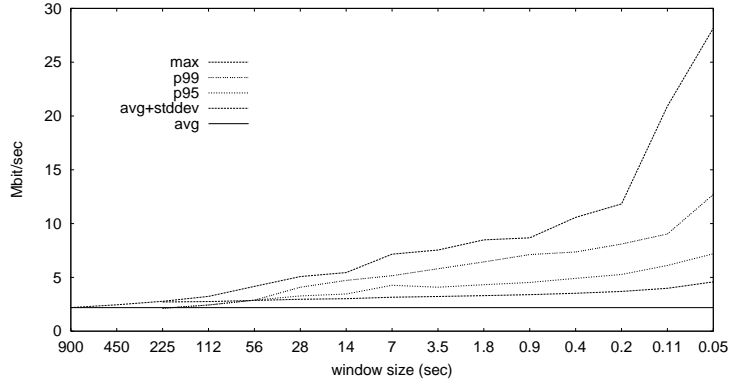


Fig. 7. Hosting Provider’s Network: Throughput vs. Time Window

It is interesting to see that the maximum, as well as the 95/99 percentiles grow steadily, up to approximately 12 splits, i.e., a window size of about 200 milliseconds; from that point on, the “growth rate” increases. Other measurements on the same networks also show an increasing “growth rate”, but the window size at which this change happens varies per measurement.

6 Conclusions

In this paper we have quantified the difference between the traditional 5 minutes traffic measurements of MRTG, and finer-grained measurements with time scales up to 10 ms. We have performed two case studies, one on the external link of a university’s residential network, and one on the access line of a small hosting provider.

The access line of this hosting provider served some tens of customers. With such small number of customers, we expected major differences between the 10 ms load figures and the traditional 5 minutes figures. Our measurements showed that these differences could be thousands of percents.

Because the university link served thousands of students, we expected that the differences between the long- and short-term averages would be relatively small. The outcome of our measurements came as a surprise, however. It turned out that, even with this number of users, an overdimensioning of about 100% is required to cater for 99% percent of the “peaks”.

Our graphs show the limitations of using MRTG figures to overprovision network links. In case bandwidth overprovisioning is the approach to ensure QoS, it may be better to base decisions on finer-grained measurements, which correspond to the time scale that determines perceived QoS.

We assume that the increasing variability of the throughput on small time scales is influenced by a number of parameters, e.g., the number of concurrent users, the users’ access rates, and other traffic characteristics such as file size

distributions. In future work we expect to quantitatively investigate the influence of each of these factors, and to derive simple dimensioning rules related to intelligent overprovisioning.

As a side-result, this paper showed that, without any special hardware or software, it is possible to perform traffic measurements with a granularity of up to 10 milliseconds, on network links carrying hundreds of Mbit/s. The tools that we've used to perform these measurements, as well as the tools that were used to analyze the results, can be downloaded from the web.

Acknowledgements

The authors would like to thank the ITBE, the university's network managers, for the opportunity to meter traffic on the network of the University of Twente and for providing assistance whenever necessary.

The authors would also like to thank Virtu Secure Webservices BV for providing access to its networking infrastructure and support in the measurement efforts.

The research presented in this paper is sponsored by the Telematica Instituut, as part of the Internet Next Generation (ING) project and the Measurement, Modelling and Cost Allocation (M2C) project.

References

1. C. Fraleigh, F. Tobagi and C. Diot, *Provisioning IP backbone networks to support latency sensitive traffic*. In *Proceedings IEEE Infocom 2003*, San Francisco CA, USA, 2003.
2. T. Oetiker. *MRTG: Multi Router Traffic Grapher*. <http://people.ee.ethz.ch/~oetiker/webtools/mrtg/>.
3. 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*, pp. 192–201, March 2002.
4. E. Weigle and W. Feng. *TICKETing High-Speed Traffic with Commodity Hardware and Software*. In *Proceedings of Passive and Active Measurement Workshop 2002*, pp. 156–166, March 2002.
5. S. Han, M. Kim and J.W. Hong. *The Architecture of NG-MON: A Passive Network Monitoring System for High-Speed IP Networks*. In *Proceeding of 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management*, pp. 16–27, October 2002.
6. Sprint ATL, IP Monitoring Project, <http://www.sprintlabs.com/Department/IP-Interworking/Monitor/>.
7. Endace Measurement Systems. <http://www.endace.com>.
8. Lawrence Berkeley National Laboratory Network Research. TCPDump: the Protocol Packet Capture and Dumper Program. <http://www.tcpdump.org/>.
9. Ipsilon Networks. tcpdpriv. <http://ita.ee.lbl.gov/html/contrib/tcpdpriv.html>.
10. Boutell.com, Inc. GD Graphics Library. <http://www.boutell.com/gd/>.
11. R. van de Meent. Homepage. <http://wwwhome.cs.utwente.nl/~meentr/>.

12. J. Cao, W. Cleveland, D. Lin and D. Sun, *Internet traffic tends to Poisson and independent as the load increases*, Bell Labs Technical Report, 2001.