Integrating Active Methods and Flow Meters
- An Implementation Using NeTraMet

Thomas Lindh[1], Nevil Brownlee[2]

[1] KTH, Royal Institute of Technology, IMIT.
Address: KTH Syd, Marinens vag 30, 136 40 Haninge, Sweden. Email: Thomas.Lindh@syd.kth.se
[2] CAIDA, SDSC, UC San Diego and The University of Auckland, New Zealand. Email: nevil@caida.org.

***Abstract*—The purpose of this paper is to present an implementation of a method that combines active and passive methods to measure and estimate the main performance parameters in IP networks. More specifically the method integrates performance and flow measurements. The main features of the method and an implementation using the flow meter NeTraMet are described. Test results for packet delays and delay variations, losses and throughput are presented. Applications to monitoring of traffic from server-based networks and an adaptive monitoring system are also considered.**

***Index Terms*--Performance monitoring, flow meter, NeTraMet, active and passive methods, sampling.**

I. Introduction

There are a number of good reasons for operators of IP networks to measure and verify the actual level of performance in their networks. In addition to traditional areas such as daily operations, usage information for billing purposes, customer reports, planning and dimensioning, the ambition to offer services with different quality assurances to customers makes measurement even more important.

We present an implementation of a method that combines the traditional ingredients in passive and active methods into a synthesis with new characteristics and possibilities. This combined approach is realised as an enhancement of the network traffic flow meter NeTraMet.

The paper is organised as follows. After a short background in section II, the monitoring method and the NeTraMet implementation are briefly presented in sections III and IV. Test results are evaluated in section V. Applications to virtual private networks and server networks, and NeTraMet as component in an adaptive monitoring system are considered in section VI. Tables and figures are assembled in two appendices.

II. Background and Related Work

In essence this work derives from two main sources. One origin is the existing and commonly used flow meter NeTraMet [1], which is an implementation of the guidelines in Real-time Traffic Flow Measurements [2]. NeTraMet can be classified as a traffic meter that utilises passive methods to collect flow-based traffic information from networks. Its ability to handle flows with different levels of granularity and to cope with high-speed networks has been proven in several tests and installations.

The second origin of this work is a method for performance monitoring that has been presented at two previous PAM conferences [3], [4]. Results from simulation based on traffic data from RIPE-NCC traffic measurement boxes [5] have been published [3], and an embedded version of the method has been implemented in Linux-based routers [4]. These projects were part of a co-operation between KTH (IMIT), Telia and Ki Consulting in Sweden.

There are several examples in recent years that flow meters and passive methods for performance monitoring attract a growing interest. The IETF working group IPFIX (IP Flow Information Export) [6] aims at standardising flow measurement output, today implemented differently in e.g. NetFlow [7], sFlow [8] and NeTraMet [1]. The initiative PSAMP is focused on the important sampling aspects of passive measurements [9].

The main contribution in this paper is to integrate active methods and performance metrics into flow meters, and thereby obtain new and richer functionality.

III. The Monitoring Method

A. The Main Characteristics of the Method

The monitoring method has the following goals and main characteristics.

- The goal is to use a single procedure to measure and estimate the main performance parameters: packet losses, packet delays, packet delay variations and throughput.
- The method is based on an in-service technique that reflects the performance of the actual user traffic.
- It is a direct method that does not depend on the accuracy of certain traffic models.
- It takes advantage of a flow meter's ability to tune packet filter parameters, such as network addresses, protocols, port numbers and service classes, in order to obtain traffic flows of different granularity.

- The method introduces the monitoring block as a fundamental concept to achieve a higher resolution than merely long-term averages for the entire measurement period. The size of the monitoring block, expressed in number of packets or time units, determines the level of resolution and accuracy of the results. This parameter and the filtering capability of the flow meter allow an operator to adjust the measurements to its goals and policy.
- A combination of traffic meters and dedicated monitoring packets is used to obtain these goals. The method could be implemented either as an integral part of existing network nodes (embedded monitoring) or as a stand-alone monitoring system, which is the focus of this paper.

B. *Traffic Meters and Monitoring Packets*

The basic idea is to use traffic meters in conjunction with monitoring packets, which are inserted into the user traffic (Figure 1). The traffic meters act as packet filters designed to count packets (and bytes) belonging to the traffic flow subjected to monitoring. These functions could either be implemented in the existing (edge) routers or as stand-alone monitoring systems. The monitoring block could either consist of a number of packets (fixed or varying), or a time interval (fixed or varying) between these dedicated monitoring packets. In the former case the generation of monitoring packets needs to be exactly co-ordinated with the counter functions. This is however not required in the latter case. This paper is focused on a stand-alone solution that uses time-based monitoring packets.

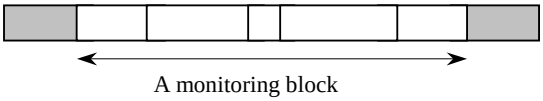

Figure 1: Two monitoring packets (shaded in the figure) enclose a monitoring block that consists of the number of user data packets being sent during the time interval between the monitoring packets.

C. *Stand-Alone Monitoring Systems*

Figure 2 shows two monitoring systems, M1 and M2, which consist of passive as well as active components: traffic meters that identify the traffic flows and keep up counters, and monitoring packets sent periodically or according to a statistical distribution. This packet generator could be implemented in the monitoring nodes or in separate hosts. The traffic meters in M1 and M2 maintain packet filters that count the number of user packets and bytes for the traffic flows subjected to monitoring. When the traffic meters in M1 and M2 detect a monitoring packet, the intermediate cumulative number of data packets and bytes, an ID of the monitoring packet and a timestamp are stored by the respective system. One could say that a monitoring packet acts as a probe that triggers a local storage of measurement data in each measurement system. After processing the measurement data recorded by M1 and M2 the results summarised in the next subsection (III.D) can be obtained.

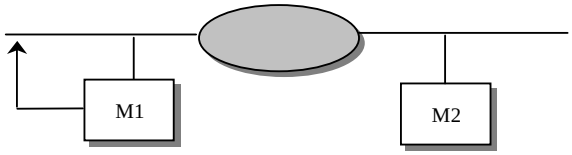

Figure 2: Two stand-alone monitoring systems, M1 and M2, which consist of traffic meters, counters and other monitoring functions. The generation of monitoring packet can be implemented in M1 and M2 or in separate hosts.

D. *The Measurement Results*

The resolution and accuracy of the measurements depend mainly on two factors, the distance between the monitoring packets and the granularity of flow definition that is applied. The main results are the following.

- The number of lost packets and the packet loss ratio during the measurement period.
- The length of the loss-free periods and the loss periods expressed in terms of the number of consecutive monitoring blocks that contains lost packets and the number of monitoring blocks without losses. Alternatively this could be expressed in terms of “loss-free seconds” and “loss seconds”.
- An estimate of packet delays and delay variations based on samples (the monitoring packets).
- The utilised capacity (throughput) between the measurement points.

IV. Implementation in NeTraMet

As distributed, NeTraMet meters traffic flows, where a flow is defined as a set of packets having specified values of their address attributes. To configure NeTraMet one writes a ruleset specifying which flows to meter, and which attribute values are required for each flow. We modified our meter to maintain a table of measurement groups. Each group may have a list of flows belonging to it. We also implemented two new flow attributes:

- GroupIdent, which is the ID of the measurement group the traffic flow belongs to;
- MeasurementData, which indicates that dedicated monitoring packets (belonging to a specific measurement group) are associated with a traffic flow. These packets cause the meter to create measurement data records (as described in section V) for each traffic flow in the measurement group.

When flow data is read from our NeTraMet meter, it includes all the measurement data records produced for a flow since the previous meter reading.

This system is simple to understand and use, and it allows one to collect measurement data for any required flow. For example, using a single ruleset, one could measure the behav-

iour of flows to several different destinations (IP addresses), carrying different protocols (TCP, UDP), and different application traffic (IP port numbers). Our measurement modifications to NeTraMet are included in the NeTraMet distribution, from version 4.5b8.

V. Test Configuration and Results

The test configuration in Figure 3 consists of two NeTraMet systems (M1 and M2) synchronised to a NTP time-server [10]. Packet losses and delays in the core network are emulated using Nistnet [11]. User traffic is generated between C1 and C2 and the monitoring packets are sent either between M1 and M2 or between C1 and C2. Except for C1 and C2 all nodes are running Linux Red Hat 7.3. The NeTraMet systems are configured to count the number of bytes and packets for the user data flows, and to detect monitoring packets associated with those flows.

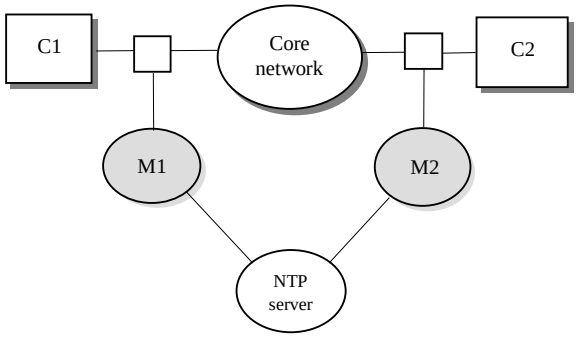

Figure 3: The test network configuration.

Each time a monitoring packet is detected a record of the following format is stored for each flow that belongs to the specific monitoring group defined using the new attributes in the NeTraMet OAM extensions:

{*Id Ts ToOctets ToPDUs FromOctets FromPDUs*}.

- *Id* is the identification of the monitoring packet. In this case the identification field in the IP header.
- *Ts* is the timestamp when the monitoring packet is transmitted (received) by the NeTraMet system. In this case the unix system clocks were synchronised to an external NTP server.
- *ToOctets* is the current intermediate cumulative value of byte counter for the traffic flow (flows) associated with the monitoring packet in the direction from source to destination.
- *ToPDUs* is the current intermediate cumulative value of packet counter for the traffic flow (flows) associated with the monitoring packet in the direction from source to destination.
- *FromOctets* and *FromPDUs* are the corresponding counter values as above for the direction destination to source.

To and *From* are specified by the NeTraMet ruleset; in this case, *To* means from C1 to C2. The tests were carried out with different characteristics and levels of losses and delays, and with different time intervals between the monitoring packets. Tcpdump [12] is used on hosts M1 and M2 to store a timestamp, the packet ID, and the number of bytes for both the data packets and monitoring packet. These recordings enable a comparison between the actual true values of the parameters and the estimates based on the NeTraMet measurements.

The data traffic consists of UDP packets that are generated with time intervals according to a negative exponential distribution. The packet size is randomly chosen between 64 and 1460 bytes. The monitoring packets are transmitted periodically. The ratio (R) between the monitoring packets and the data packets was 1/100, 1/200, 1/400, 1/800, 1/1000 and 1/2000. The data traffic and the monitoring packets were subjected to the same loss and delay characteristics. In order to maintain the same loss and delay conditions during each test with different values of R, the recorded data with R=1/100 was re-sampled off-line for R=1/200, 1/400 etc. The measurement periods varied between 5 and 15 minutes in the different tests. Some of the results are presented below. Tables and figures are assembled in Appendix I and II.

The rules for NeTraMet are programmed using SRL (simple ruleset language) [14]. The code used in our tests can be found in Appendix III. The flow files delivered by NeMaC (NeTraMet Manager/Collector) [15] at each NeTraMet system have been parsed with an awk program. The resulting measurement results were then processed and presented using Matlab.

A. One-Way Delays and Delay Variations

Figure 6 illustrates how the one-way delay varies during the measurement period. Table I summarises the actual and estimated values of the delay and delay variation for different values of the ratio between the number of monitoring packets and user data packets (R). The relative error in the estimating the average delay spans from 1.8% for R=1/100 up to 13.8% for R=1/2000. The estimates of the maximum delay are of course less accurate, with relative errors from 14.5% (R=1/100) up to 72.5% (R=1/2000). However, the estimates of the average delay for R down to 1/1000 are well within the 99th percentile. The relative errors in measuring the maximum delay variation are quite similar to those for the maximum delay. The histogram in Figure 7 and 8 show the distribution of one-way delays for R=1/100 and R=1/1000 respectively during the measurement period.

However, for several applications the maximum delay is not a useful metric, since delays above a certain limit in a playback (jitter) buffer are considered lost anyhow. In these cases it can be more appropriate to define a P% minimum delay window and compute the fraction of the delays that fall within this window [13].

Figures 9 and 10 show histograms of the delay variations based on the monitoring packets for R=1/100 and R=1/1000.

There are several different schemes for sending monitoring packets that can be applied. Besides sending the measurement packets periodically, either with certain time intervals or certain number of data packets between the monitoring packets, we have also implemented more random techniques. An algorithm with alternating active on-periods and passive off-periods has been tested. First, periodic bursts of monitoring packets (e.g. five packets during a one second long on-period followed by a four second off-period) was compared to the previous scheme of sending one measurement packet periodically every second. Secondly, the length of the on-periods and the off-periods were generated according to the negative exponential distribution. In neither of these schemes any significant improvement of the relative error were found compared to the results in Appendix II. However, the tests have not yet been performed systematically for different distributions of traffic and losses.

B. Throughput

In Table II the average, maximum and minimum of the throughput per monitoring block are presented for different values of the ratio between monitoring packets and user data traffic. The average throughput is approximately the same for all values of R. The estimate of the maximum throughput increases when the size of the monitoring block decreases as expected. In this case the maximum throughput increases by a factor 1.6 when the block size is decreased from R=1/2000 to R=1/100. The distribution of the throughput per monitoring block is illustrated in Figure 11 (R=1/100) and Figure 12 (R=1/1000). The desired resolution of the estimates will depend on the operators' requirements.

C. Loss Periods and Loss-Free Periods

In addition to the loss ratio for the entire measurement period this version of NeTraMet is capable of providing intermediate metrics of losses for each monitoring block. The resolution of these results is determined by the ratio between the monitoring packets and the data packets. This feature makes it possible to define periods that contain lost packets and those without losses. A loss-free period is here defined as the number of consecutive monitoring blocks without packet losses, and consequently a loss period as the number of consecutive monitoring blocks with one or more lost packet.

Table III and IV summarises the length of the respective periods expressed in time units or the number of packets (bytes). The actual loss process during the measurement period is illustrated in Figure 13, and the subsequent Figures 14-27 illustrate some metrics that are possible to obtain.

The distribution of the loss ratio in the monitoring blocks is shown in Figure 14 (R=1/100) and Figure 15 (R=1/1000). A more powerful metric can be obtained using the defined loss periods, illustrated in Figures 16-23, and loss-free periods characterised in Figures 24-27. The length of the loss periods (in seconds) versus time during the measurement period is depicted in Figures 16 and 17 for R=1/100 and R=1/1000. The periods are measured in seconds and also in octets. The distribution of the length of the loss periods in seconds and octets are illustrated in the histograms in Figures 18-21, and for the loss ratio in Figures 22-23 for different ratios of monitoring packets. The distribution of the length in seconds and octets of the loss-free periods can be seen in Figures 24-27.

The ratio between the loss-free time and the total measurement period is 71% for R=1/100, and 18% for R=1/2000. Hence, the loss period is 29% (R=1/100) and 82% (R=1/2000) of the total time respectively in this case. The total loss-free periods expressed in bytes can of course be quite different from those in time units, due to the nature of the traffic process. However, in these tests the results are similar for the two cases.

D. Evaluation of Test Results

The laboratory tests have clearly shown that NeTraMet measures one-way delays, packet losses and throughput accurately for the entire measurements period, and specifically using monitoring blocks of different sizes. Our conclusion is that NeTraMet is very well suited for the monitoring purposes described previously in the paper. One limitation has been the constraints on the frequency of the monitoring packets, approximately 5 packets/second due to limitations set by the SNMP implementation. The maximum UDP segment size in NeTraMet/NeMaC is 1500 bytes.

The extension of NeTraMet with the monitoring method described in [3] and [4] adds performance measurement capabilities to the flow meter (or flow measurement capabilities to a performance meter). The tests show that a single measurement provides a wide range of performance metrics.

One central property of the method is the dual role of the monitoring packets. On one hand to obtain samples for delay estimates and on the other hand to delimit the size of the monitoring block, which determines the level of resolution of the loss and throughput results. If the periodic pattern of the monitoring packets would undermine the accuracy of the delay measurements this dual function had to be reconsidered. However, the tests have so far not rejected the use of a periodic pattern for the monitoring packets.

VI. Applications and System Aspects

The method is directly applicable to cases where the ingress and egress nodes for packets are known, such as IP tunnels in virtual private networks and measurements in server-based networks.

IP-based virtual private networks represent a case where more elaborate performance monitoring is motivated. The goal is often to measure and estimate losses, delays and throughput between provider edge nodes that surround a core network or between customer edge nodes. This has been discussed previously in [3] and [4].

A. Server-Based Networks

Monitoring of traffic flows and performance parameters in a server-based network (Figure 4) is an interesting application of

the measurement method presented in this paper. Assume that clients connected to the edge nodes retrieve real time services from the server network. Processing of data collected by the systems M and N_i result in measurements of packet losses (ratio and distribution), one-way delays and delay variations and throughput. Besides these performance metrics it is also feasible to determine how the total traffic leaving the server network is distributed between the respective edge nodes. In addition, the routing paths can of course be traced if some of the monitoring packets also record the route.

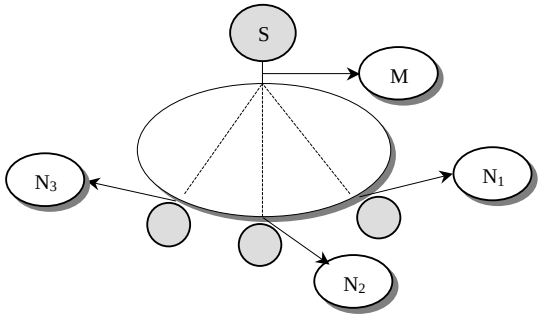

Figure 4: NeTraMet system M monitors flows between the server network (S) and all the edge nodes. N_1, N_2 and N_3 monitor the flow between the server network and the respective edge node. Monitoring packets are sent by the NeTraMet systems or by separate nodes.

One objection to this scenario may be hesitations as to whether it is scalable enough to handle high-speed data rates and high-resolution measurements. Several variables determine the amount of storage capacity needed in this case, such as the number of monitoring packets per time unit (the size of the monitoring blocks) and the number of traffic flows that are monitored. However, the number of server nodes is small in this case, which means that the number of flows that have to be monitored are limited. The amount of measurement data being produced is therefore determined by the number of network destination addresses (behind the edge nodes), the granularity of the traffic flows, the data packet rates and the desired accuracy and resolution of the measurement results.

Provided that a monitoring packet is sent once a second between each pair of M and N_i during a busy hour each N_i needs to store 3600 records and the NeTraMet node at the server must to keep i·3600 records. If the average data rate is 100 Mb/s and the average packet size is 250 bytes, the average size of the monitoring block will be 50000 packets. In order to obtain a ratio R=1/1000 of the monitoring packets to the user data packets, 50 monitoring packets per second is required. In this case each N_i has to maintain 180000 records and correspondingly i times that for measurement node M. However, if multicast is used to transmit the monitoring packets the requirements on the M node will be the same as on the N_i nodes.

One possible scenario is that the flows in the monitoring group are completely defined, e.g. that the source IP addresses (of the servers) as well as the destination network addresses of the flows are known in advance. However, NeTraMet can also be used to find the finer-detailed sub-flows (e.g. different network destination addresses) the main flow from the server network is composed of. In addition, the OAM extension to NeTraMet provides facilities for performance monitoring of these flows.

B. An Adaptive Feedback Monitoring System

In Figure 5 the NeTraMet performance and flow meter is a component of a monitoring system. The output from the NeTraMet meters (N_1 and N_2) are parsed and processed by the analysis component, which delivers reports to the operators and input data to the control component. Based on input data from the ongoing measurements, other network information (alarms etc) and requirements from the operator, the control part generates an output signal to the monitoring packet generator. The analysis and control components could of course support several distributed NeTraMet nodes.

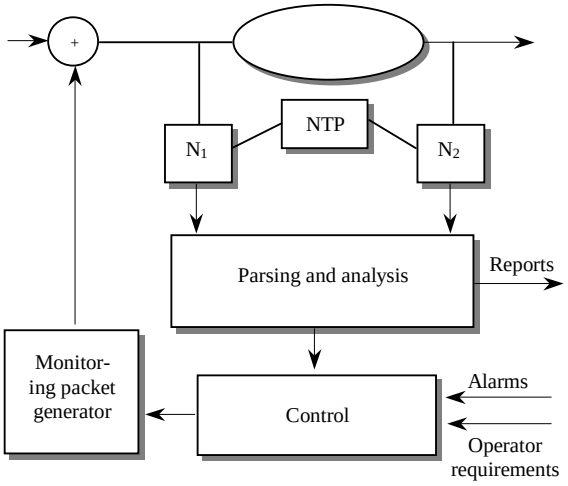

Figure 5: A feedback monitoring system based on NeTraMet nodes (N). The monitoring packets are generated based on information from the ongoing measurements, operator requirements and other network event.

Several different schemes and policies for measuring performance of traffic flows are feasible. The simplest case would be to generate monitoring packets with a constant interval despite changes in traffic and the network. Another alternative would be to maintain a constant ratio between the monitoring packets and the user data traffic. Still more related control rules could be specified for performance monitoring of certain service classes, e.g. low loss and (or) low delay requirements. In these cases an adaptation to feedback information from the ongoing measurements is helpful. We restrict the further discussion to two examples, first to keep the share of the monitoring packets of the total user data traffic to a constant level and secondly, to find the appropriate frequency of monitoring packets with respect to the measurement accuracy.

1) Maintaining a Constant Ratio between Monitoring Packets and Data Packets

Assume that a set of NeTraMet flow meters is deployed at appropriate points in a network.

i) Let monitoring packets be transmitted rather sparsely between the measurement nodes, e.g. once a minute, during a relatively long period.
ii) Determine, based on these observations, thereafter the location of the busy period, the average bit rate (B) and packet frequency P_{freq}.
iii) Let the reference value R be the desired the ratio of monitoring packets to data user packets.
iv) The frequency of the monitoring packets is then given by $M_{freq} = R \cdot P_{freq}$, and correspondingly the interval between the monitoring packets by $M_T = 1/M_{freq}$. Hence, the average packet size during the measurement period is B/P_{freq}.
v) Repeat this procedure for the busy period or for subintervals of the busy period.

2) Optimising the Size of the Monitoring Blocks

An operator may want to keep the measurement results within certain levels of accuracy expressed in terms of confidence levels and confidence intervals. Since the loss and throughput measurements are more or less exact this issue mainly concerns the delay estimates, which are based on sampling using monitoring packets. The operator might also want to obtain certain levels of resolution or granularity for the loss and throughput measurements, which is determined by the size of the monitoring block.

i) Determine the busy period as outlined in the previous paragraph, and estimate the standard deviation s.
ii) The number of samples n needed to obtain desired confidence level confidence level α and interval d is given by $n = z^2 s^2/(d/2)^2$, where s is an estimate of the standard deviation, and z can be retrieved from statistical tables of the normal distribution (given α) [16].
iii) The number of required samples n divided by the measurement period is the frequency of monitoring packets to be applied. Hence, the concentration of monitoring packets is $R = P_{freq}/M_{freq}$.

Another approach would be to use a rather high frequency of monitoring packets, e.g. R=1/100, during a busy period as a kind of reference test sequence. These measurement records can be re-sampled off-line for lower values of the parameter R. The frequency of the monitoring packets can thus be adjusted to the preferred levels of resolution and accuracy. A comparison can be made to the results of the reference probe test, with respect to loss and loss-free periods, throughput, one-way delays and delay variations. There are probably several other ways to formulate, implement and optimise rules for monitoring of these parameters in various network scenarios.

The overall goal for the system in Figure 5 is to dynamically adapt the transmission of monitoring packets to: *i)* the operator's requirements and policy; *ii)* traffic conditions reported by network management systems; and *iii)* the NeTraMet measurement feedback results. Such a system can be used to as a tool to continuously learn the loss and delay process in network traffic, to determine the preferred resolution of measurements, and of course to monitor and verify service level agreements.

C. Open Issues and Remaining Work

Next step in this project is to implement the measurement method in an operating commercial network. The purpose is partly to test capacity, performance and scalability, and partly to use the system as a tool for traffic analysis, especially of losses and delays.

A long-term goal is to implement a prototype of the system outlined in Figure 5, and also to explore how measurements data can serve as input for traffic control functions.

Two more specific issues deserve to be mentioned. To avoid influence on delay measurements from periodic patterns in network traffic a randomised generation of probe packets is sometimes recommended. A way to detect periodic properties would be to perform spectral and time series analysis, e.g. to estimate the autocorrelation function based on the retrieved measurement data.

Disordering of packets due to re-routing sometimes has the effect that packets that belonged to one specific monitoring block when they pass the entry-monitoring node can appear in a different monitoring block when they arrive at the exit-monitoring node. Since the byte and packet counters are cumulative and therefore the total number of losses is known, this can be solved by aggregating several monitoring blocks. However, if a packet is lost in block x and at the same time a packet in the neighbouring block (x-1) is delayed and therefore arrives in block x, it appear as though the loss occurred in block (x-1). Apparently one cannot tell which of the two neighbouring blocks the packet loss took place in. These re-routing events can be detected and delimited in time by periodically sending probe packets that record the packet trajectory between the measurement points.

VII. SUMMARY

The purpose of this paper has been to present an implementation of a traffic flow performance meter. The extensions to the well-known traffic meter NeTraMet are described and some laboratory test results are published. This combination of passive and active approaches has the advantage of adding performance measurement capabilities to the flow meter (or flow measurement capabilities to a performance meter). The tests show that a single measurement provides a wide range of performance metrics. Laboratory results for packet delays and delay variations, losses and throughput are presented. Applications to monitoring of traffic from server-based networks and an adaptive monitoring system are also considered.

ACKNOWLEDGEMENTS

Professor Gunnar Karlsson at KTH/IMIT has supported this work and been very helpful with constructive ideas and criticism. We also thank Ibrahim Orhan at KTH Syd, and the students Daniel Lindqvist, Shelton Videau and Leif Larsson for their valuable contributions to the laboratory configuration and the tests.

VIII. REFERENCES

[1] NeTraMet home page: http://www.auckland.ac.nz/net/NeTraMet/
[2] RFC 2720-2724
[3] Lindh T.: “An Architecture for Embedded Monitoring of QoS Parameters in IP-Based Virtual Private Networks”, Proceedings of Passive and Active Measurements (PAM) 2001, Amsterdam, 23-24 April 2001.
[4] Lindh T.: “A New Approach to Performance Monitoring in IP Networks –combining active and passive methods”, Proceedings of Passive and Active Measurements (PAM) 2002, Fort Collins, 25-26 March 2002.
[5] Georgatsos F., Gruber F., Karrenberg D., Santcroos D., Susanj A., Uijterwaal H., Wilhelm R.: “Providing Active Measurements as a Regular Service for ISP's”, Proceedings of Passive and Active Measurements (PAM) 2001, Amsterdam, 23-24 April, 2001.
[6] IPFIX IETF working group home page: http://www.ietf.org/html.charters/ipfix-charter.html
[7] White Paper - NetFlow Services and Applications: http://www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm
[8] SFlow home page: http://www.sflow.org/
[9] PSAMP IETF working group home page: http://www.ietf.org/html.charters/psamp-charter.html
[10] D. Mills NTP home page: http://www.eecis.udel.edu/~ntp/
[11] Nistnet home page: http://www.antd.nist.gov/itg/nistnet/
[12] Tcpdump.org home page: http://www.tcpdump.org
[13] Mandeville B., Pullin D., Sargood S.: “Optimising IP Network Performance Through Active Measurement”, FITCE Congress, 2002.
[14] “SRL: A Language for Describing Traffic Flows and Specifying Actions for Flow Groups”, RFC 2723, 1999.
[15] Brownlee N.: “Reference manual for NeTraMet and NeMaC version 4.3”, 1999.
[16] Thompson S.K.: “Sampling”, Wiley & Sons, 1992.

APPENDIX I

TABLE I
MEASUREMENTS OF DELAY AND DELAY VARIATIONS FOR DIFFERENT VALUES OF THE PARAMETER P, WHICH IS THE RATIO BETWEEN THE NUMBER OF MONITORING PACKETS AND THE NUMBER OF USER DATA PACKETS.

(unit: ms)	Exact values	P= 1/100	P= 1/400	P= 1/800	P= 1/1000	P= 1/2000
Mean delay	52.46	53.40	50.56	47.59	47.42	45.24
Relative error		-1.8%	3.6%	9.3%	9.6%	13.8%
Max delay	726.1	621.03	596.0	305.7	294.1	199.4
Relative error		14.5%	17.9%	57.9%	59.5%	72.5%
Min delay	6.07	11.00	11.68	11.68	11.69	11.69
99^{th} perctile delay	310					
Max. delay variation	666.6	561.3	548.8	280.2	266.8	168.0
Relative error		15.8%	19.0%	61.7%	63.4%	77.1%
99^{th} perctile delay var.	260					

TABLE II
MEASUREMENTS OF THROUGHPUT FOR DIFFERENT VALUES OF THE PARAMETER P, WHICH IS THE RATIO BETWEEN THE NUMBER OF MONITORING PACKETS AND THE NUMBER OF USER DATA PACKETS

(unit: b/s)	P= 1/100	P= 1/400	P= 1/800	P= 1/1000	P= 1/2000
Mean throughput	$6.4 \cdot 10^5$	$6.4 \cdot 10^5$	$6.4 \cdot 10^5$	$6.4 \cdot 10^5$	$6.4 \cdot 10^5$
Min throughput	$1.66 \cdot 10^5$	$4.3 \cdot 10^5$	$5.4 \cdot 10^5$	$5.4 \cdot 10^5$	$5.6 \cdot 10^5$
Max throughput	$1.1 \cdot 10^6$	$8.1 \cdot 10^5$	$7.5 \cdot 10^5$	$7.2 \cdot 10^5$	$7.0 \cdot 10^5$

TABLE III
MEASUREMENTS RESULTS FOR LOSS PERIODS

(units: s, packets)	P= 1/100	P= 1/400	P= 1/800	P= 1/1000	P= 1/2000
Mean time	0.68	4.86	9.03	12.55	28.64
Max time	6.50	31.20	34.34	56.53	59.02
Min time	0.25	0.99	1.99	2.49	4.98
Total time loss periods	73.00	126.44	153.51	163.11	200.46
(% of time)	29%	51%	62%	66%	82%
Loss ratio in loss periods	2.27%	1.61%	1.32%	1.31%	0.81%
Mean # pkts in loss periods	9	37	56	73	136
Total # packets in loss periods	30177	50906	61646	64969	80181
(% of pkts)	30%	51%	62%	65%	81%

TABLE IV
MEASUREMENT RESULTS FOR LOSS-FREE PERIODS

(units: s, packets)	P= 1/100	P= 1/400	P= 1/800	P= 1/1000	P= 1/2000
Mean time	1.65	4.57	5.20	6.05	7.47
Max time	18.00	17.98	15.92	14.94	14.94
Min time	0.25	1.00	1.99	2.49	4.98
Total time loss-free periods	178.5	123.36	93.53	84.66	44.82
(% of time)	71%	49%	38%	34%	18%
Total # pkts in loss-free periods	70344	49175	37613	34487	18297
(% of pkts)	70%	49%	38%	35%	18%

APPENDIX II

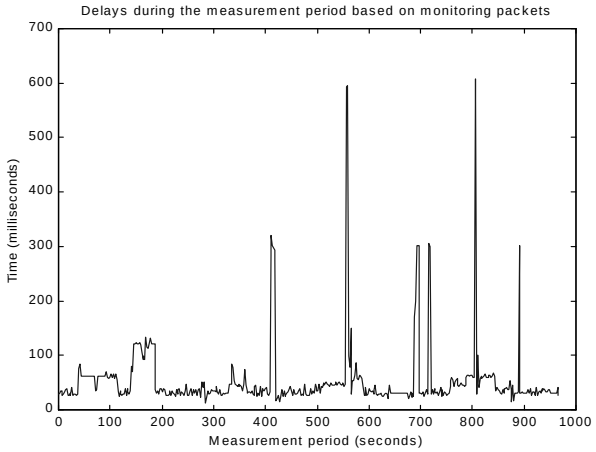

Figure 6: One-way delays versus time during the measurement period, which Figures 7-11 are based on.

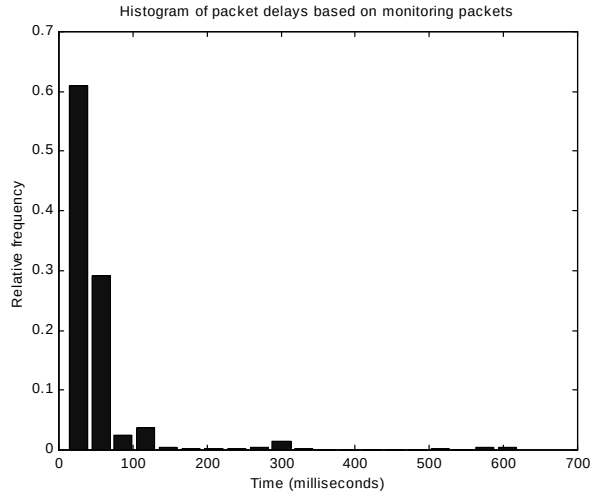

Figure 7: Histogram of one-way delays based on monitoring packets with *R*=1/100.

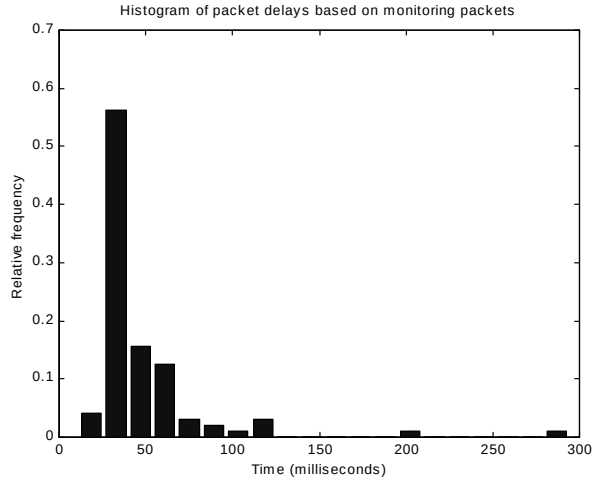

Figure 8: Histogram of one-way delays based on monitoring packets with *R*=1/1000.

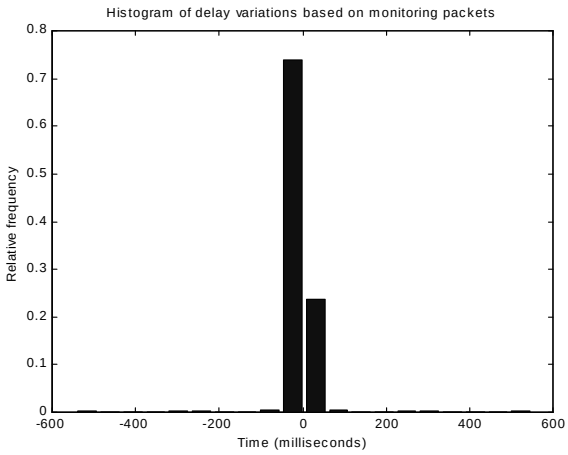

Figure 9: Histogram of delay variations based on monitoring packets with *R*=1/100.

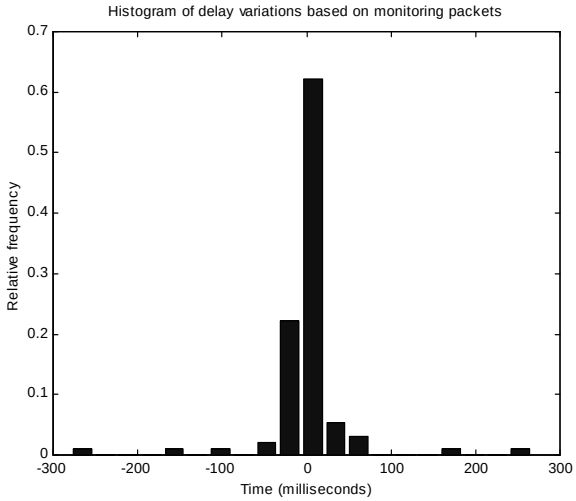

Figure 10: Histogram of delay variations based on monitoring packets with *R*=1/1000

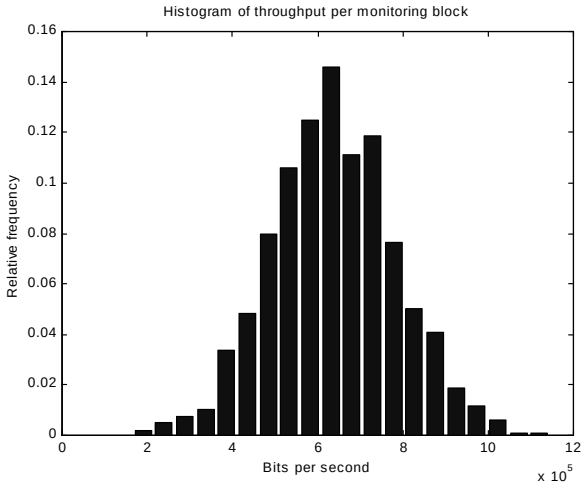

Figure 11: Histogram of the throughput based on monitoring packets with *R*=1/100.

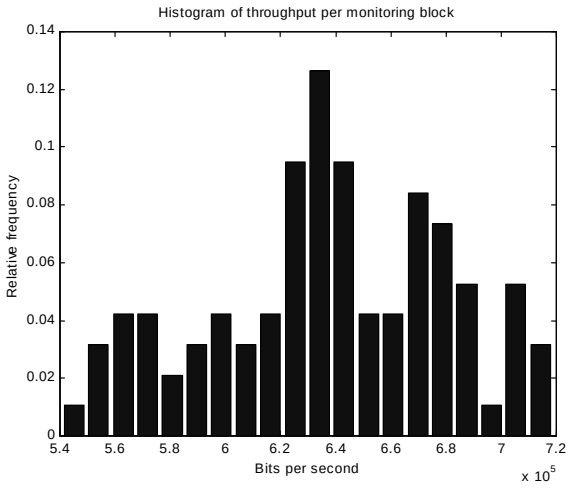

Figure 12: Histogram of the throughput based on monitoring packets with *R*=1/1000.

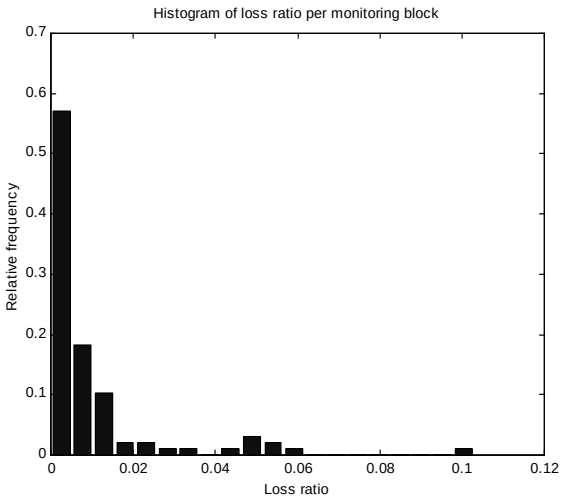

Figure 15: Histogram of the loss ratio per monitoring block with *R*=1/1000.

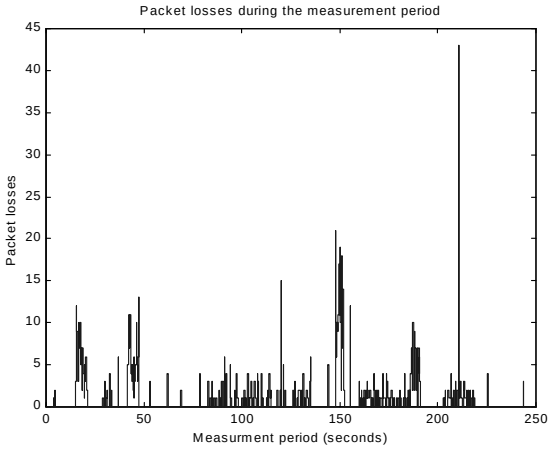

Figure 13: The packet losses versus time during the measurement period, which Figures 13-26 are based on.

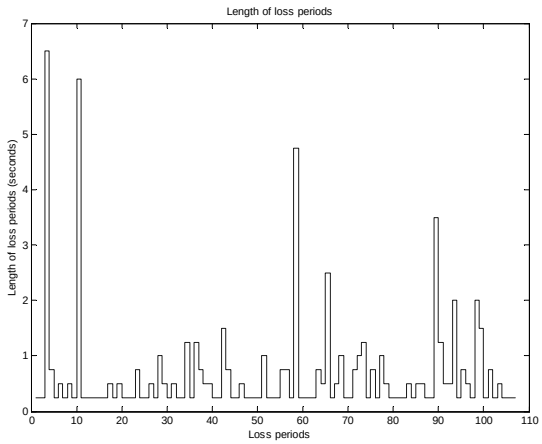

Figure 16: The length of the loss periods versus time during the measurement *R*=1/100.

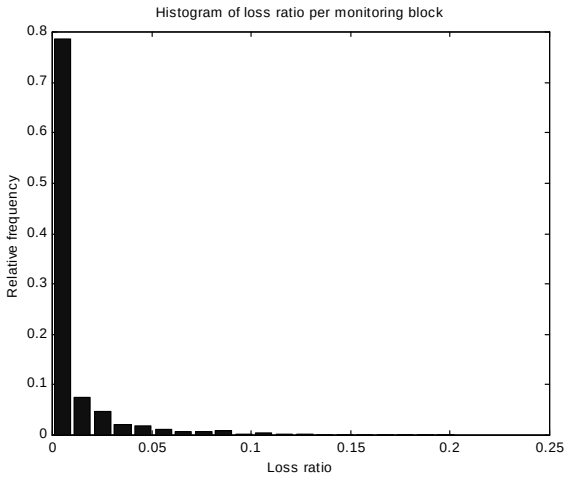

Figure 14: Histogram of the loss ratio per monitoring block with *R*=1/100.

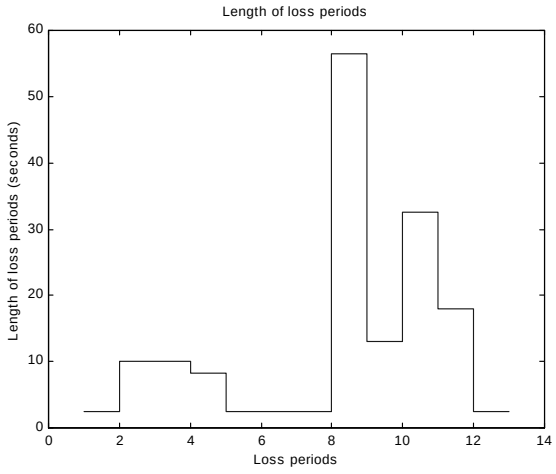

Figure 17: The length of the loss periods versus time during the measurement *R*=1/1000.

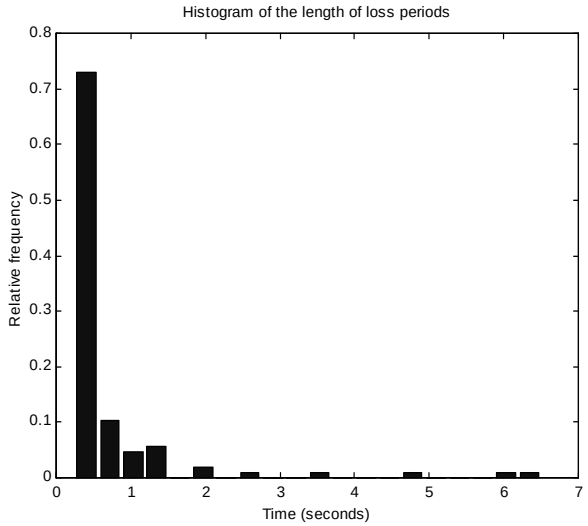

Figure: 18: Histogram of the length in seconds loss periods with *R*=1/100.

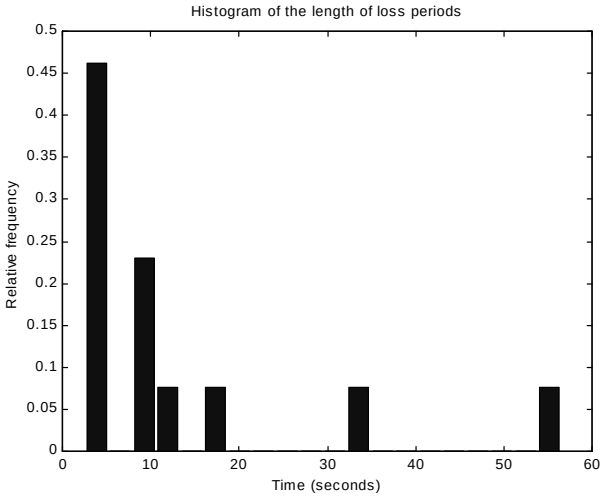

Figure 19: Histogram of the length in seconds of loss periods based on monitoring packets with *R*=1/1000.

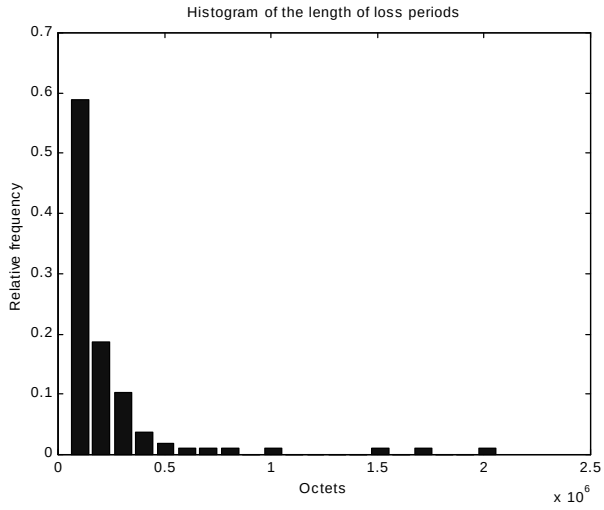

Figure 20: Histogram of the length in octets of the loss periods based on monitoring packets with *R*=1/100.

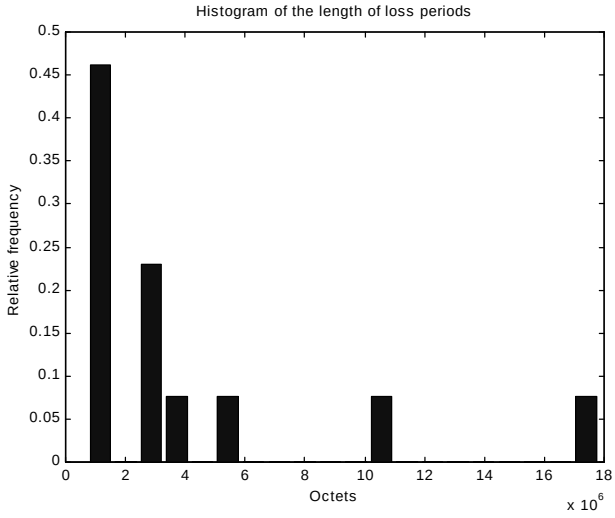

Figure 21: Histogram of the length in octets of the loss periods based on monitoring packets with *R*=1/1000.

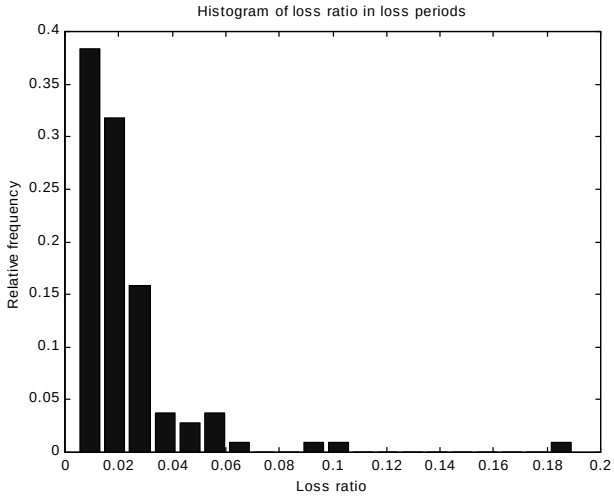

Figure 22: Histogram of the loss ratio in the loss periods based on monitoring packets with *R*=1/100.

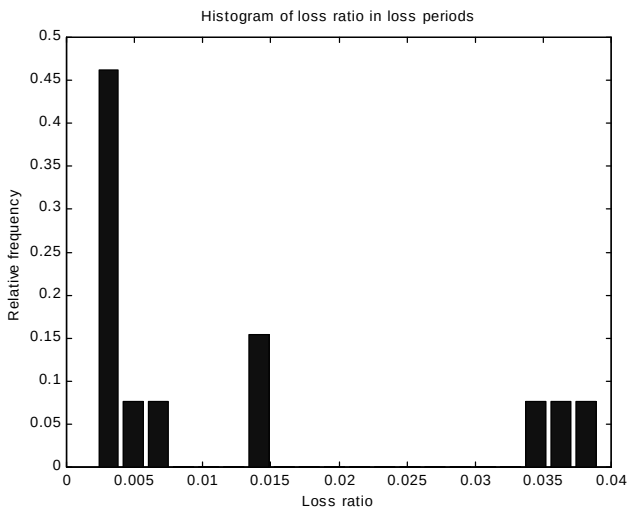

Figure 23: Histogram of the loss ratio in the loss periods based on monitoring packets with *R*=1/1000.

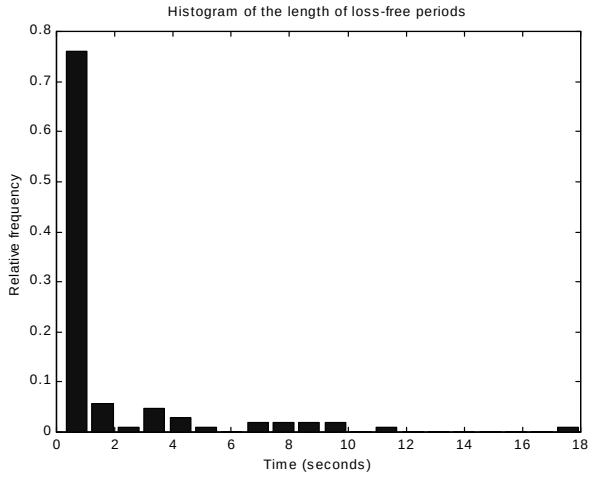

Figure 24: Histogram of the length in seconds of the loss-free periods based on monitoring packets with *R*=1/100.

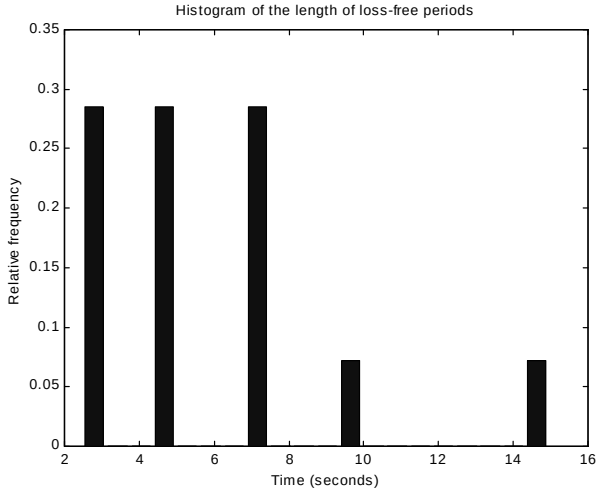

Figure 25: Histogram of the length in seconds of the loss-free periods based on monitoring packets with *R*=1/1000.

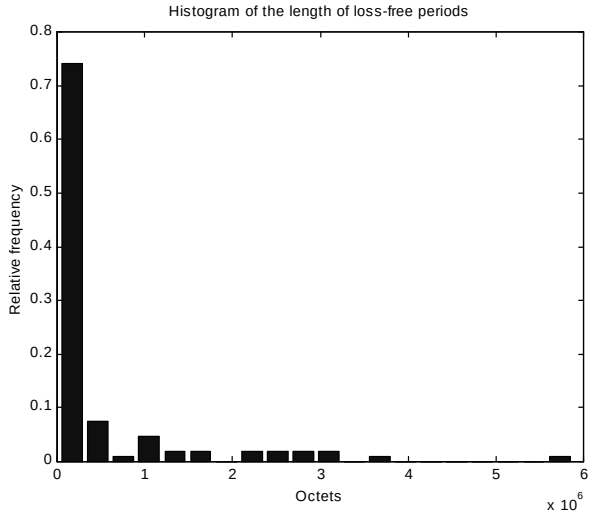

Figure 26: Histogram of the length in octets of the loss-free periods based on monitoring packets with *R*=1/100.

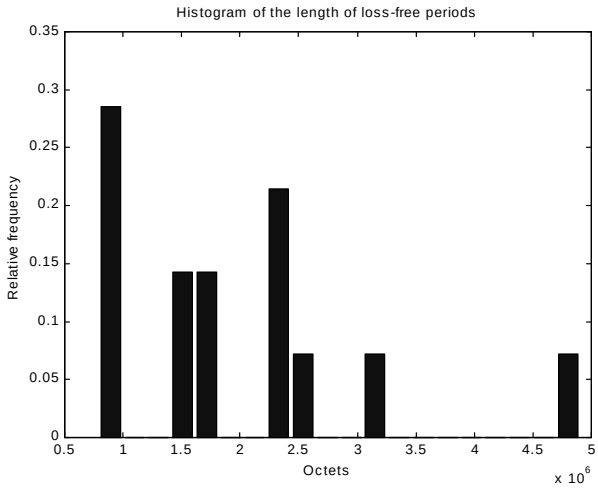

Figure 27: Histogram of the length in octets of the loss-free periods based on monitoring packets with *R*=1/1000.

APPENDIX III

```
# Ruleset to test OAM attributes – srl (simple ruleset language) code

define OAM_SENDER = 40.0.0.4;  #OAM sender
define OAM_RECEIVER = 30.0.0.1; #OAM receiver
define OAM_PROTOCOL = udp;  #UDP monitoring packet
define DATA_SENDER = 40.0.0.1;  #data sender to be monitored
define DATA_RECEIVER = 30.0.0.2; #data receiver to be monitored
if SourcePeerType == IPv4 save;
else ignore;
if SourcePeerAddress == OAM_SENDER &&
    DestPeerAddress == OAM_RECEIVER &&
    SourceTransType == OAM_PROTOCOL #OAM trigger packet UDP
    && MatchingStoD == 1 {
        # Only trigger on S->D, not D->S as well
        save SourcePeerAddress = 1.0/8;
        save OAMdata = 1.0.0!0 & 0.0.0!0; # Save OAM data for all flows
                                          # with OAMident 1
        count;
    }
else if SourcePeerAddress == DATA_SENDER &&
        DestPeerAddress == DATA_RECEIVER save, {
        if DestPeerAddress == 224.0/8  save; # Multicast
        else
            save SourceTransType;
        store OAMident := 1; # All these (fine-detail) flows have OAMident 1
        count;
    }
else if MatchingStoD == 2 {  # other flows
        save SourcePeerAddress;  Save DestPeerAddress;
        save SourceTransType;
        save SourceTransAddress;  Save DestTransAddress;
        count;
    }
set test_oam4;
format
FlowRuleSet FlowIndex FirstTime
”  ” OAMident
”  ” SourcePeerAddress DestPeerAddress
SourceTransType SourceTransAddress DestTransAddress
” {” OAMdata ”}”;
```