A New Approach to Performance Monitoring in IP Networks
– combining active and passive methods

Thomas Lindh

KTH IMIT and Ki Consulting
Address: KTH Syd, Marinens vag 30, 136 40 Haninge, Sweden.
Email: Thomas.Lindh@syd.kth.se

***Abstract*--This paper presents a simple prototype implementation of a performance monitoring system for IP networks that applies both active and passive methods. The method is based on an appropriate combination of traffic meters and dedicated monitoring packets. These monitoring functions could be an integral part of ordinary network elements, or stand-alone systems. The prototype is implemented in Linux-based routers and the main application is monitoring in IP-based virtual private networks.**

***Index Terms*--Performance monitoring, performance parameters, packet losses, delays, throughput, service level agreements.**

I. Introduction

Measurements and estimation of performance parameters in IP networks are becoming increasingly important for today's operators. There are several reasons to develop efficient and reliable methods in this field. Deployment of differentiated services in IP networks will require effective but also simple methods for measurement of relevant performance parameters to support and verify service level agreements with customers. A monitoring system must also support the daily operation, traffic control and planning of an operator's network with relevant and timely measurements and estimates. Accurate measurements of usage are also the foundation for accounting and billing. Measurements are undoubtedly important for a network operator, albeit they are not a goal but means to achieve certain objectives.

In this paper we describe a prototype implementation of a method for performance monitoring of QoS parameters in IP networks, especially virtual private networks. Our approach is to use an appropriate combination of active and passive methods in order to measure and estimate performance parameters in the user traffic. The work is a part of a joint project between the Department of Microelectronics and Information Technology (IMIT) at KTH and Ki Consulting (formerly Telia ProSoft).

The paper is organized as follows. After a short background in section II, the monitoring method is briefly presented in section III. A prototype implementation is described in section IV followed by a discussion about the size of the monitoring block in section V and comments on future work in section VI.

II. Background and Related Work

Methods for measuring and monitoring network performance parameters are traditionally divided into passive and active techniques. The idea behind passive methods is to capture packets in order to store and collect information from various fields within the packet header. Unlike active methods, passive monitoring systems are non-intrusive and do not load the network. Although all passive methods share that basic characteristic, there exist quite different variants. While traffic meters usually minimize the amount of stored data by e.g. merely incrementing counters, other passive tools are designed to record some header fields from each packet.

The method described in this paper can be seen as an integration of passive components (traffic meters) and active components (monitoring packets). Previous work in the project has been presented in [1] and [2]. Some relevant related work in this field are RIPE NCC Test Traffic Measurements [5], NIMI [12], Surveyor [13], AMP [14], NetFlow [15] and Service Assurance Agent from Cisco [20], NeTraMet [10] and several tools based on the RMON management information base. As in the RMON case, SNMP (Simple Network Management Protocol) is a frequently used standard to organize and manage measurement data. Besides these well-known frameworks and tools, a test project at IBM [3] and the EURESCOM project QUASIMODO [4] are also related to our work.

Measurements may also be classified in terms of the level of communications at which they are performed. Besides the network level, measurements of services and applications, e.g. end-to-end between servers and clients, are often equally important. This paper is focused on performance monitoring at the network level (IP) between edge nodes that surround a core network. Measures of response times and availability for servers are often valuable complements to measures of packet

losses, delays and throughput between network edge nodes.

The rapidly growing interest in management of service level agreements and support for traffic engineering in IP networks is reflected in a growing number of products and tools in the area. This development emphasizes the importance of the standardization efforts from IETF (IPPM Working Group), ITU, ETSI and other international bodies. The SLA Handbook from TeleManagement Forum [16] is meant as a common framework to be used by operators and their customers in the entire process of creation and management of service level agreements.

III. The Monitoring Method

A. The Context

A monitoring method does of course not live a life of its own. It has to be seen in a broader context and be subordinated a network or service operator's policy and objectives regarding performance monitoring. There is no meaning to carry out extensive measurements and monitoring schemes for their own sake. An operator has to consider its entire service production and business process, also in relation to the customers' needs and requirements.

The goal is to achieve a proper balance between performance measurements between edge nodes at the network level, and between end nodes at the application level (a vertical view). Today's measurement systems and tools are used for different purposes such as monitoring of service level agreements, network operations and traffic control, and accounting (a horizontal view). Although this paper is focused on measurements on the IP level between edge nodes, the broader context should not be forgotten. There are several advantages to gain from a vertical as well as a horizontal integration approach.

B. The Main Characteristics of the Method

The implemented method, described in section D, has been presented at PAM 2001 [1]. It has the following goals and main characteristics.

- The goal is to measure and estimate losses, delays, delay variations and throughput using a single procedure instead of several different methods.
- The method is based on an in-service technique that reflects the performance of the actual user traffic.
- It should be possible to differentiate the results with respect to different customers, types of traffic, service classes and virtual private networks.
- It should also be possible to obtain a higher resolution than merely long-term averages for the entire measurement periods, e.g. the distribution in time of losses and throughput.
- A combination of traffic meters and dedicated monitoring packets are used to obtain these goals. The method could either be implemented as an integral part of existing network nodes (embedded monitoring) or as stand-alone systems.
- The distance between the monitoring packets, i.e. the size of the monitoring block, determines the level of resolution and the accuracy of the measurement results. This parameter can be used to adjust the measurements to an operator's monitoring policy.

C. Virtual Private Networks

This study is focused on IP-based virtual private networks since they represent a case where more elaborate performance monitoring is clearly motivated. The goal is to measure and estimate losses, delays and throughput for IP traffic in virtual private networks between provider edge nodes or customer edge nodes, based on the general topology in Figure 1. This is further explained in [1].

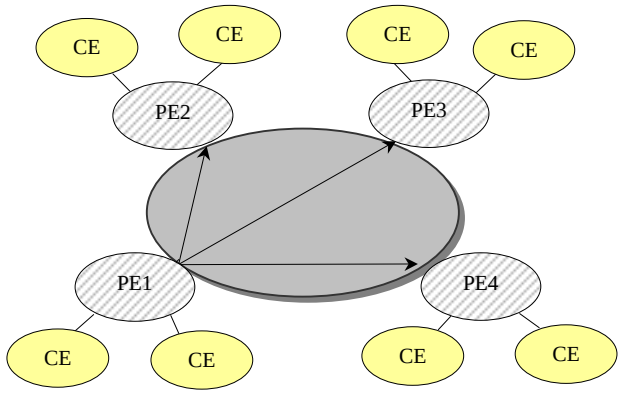

Figure 1: A topology for virtual private networks. The provider edge node 1 (PE1) acts an entry-monitoring node for the outgoing traffic and the receiving nodes act as exit-monitoring nodes for the incoming traffic. Corresponding monitoring functions may also be implemented in customer edge nodes (CE) as well as in individual hosts.

D. Requirements for Monitoring in IP Networks

IP's connectionless nature is a major problem in designing monitoring systems. To use monitoring packets in IP networks requires that an entry-monitoring (sending) node can find out which of the exit-monitoring (receiving) nodes the packet will traverse, based solely on its destination IP address. Similarly, a receiving node has to determine from which sending node an arriving packet was sent. In addition, the receiving node must be able to measure or estimate the packet size for the monitoring blocks, in order to compute throughput and utilization.

The solution for a general IP network requires efficient look-up functions in order to determine the correct exit (and entry) monitoring nodes. However, it turns out that in several important cases the complexity of the problem is considerably reduced. In virtual private networks, implemented by means of point-to-point tunnels, the address of the exit node is actually carried in the additional IP header. In a more general case a look-up function that returns the correct exit (entry) monitoring node given an IP destination (source) address is needed.

Besides IP-based VPNs in fixed or mobile networks there are several other obvious cases where the method is applicable such as other tunnel-based services and the virtual leased line service (expedited service in DiffServ) [18]. Even if the paper is focused on monitoring between edge nodes that surround a core network, the method is not limited to this case. Measurements in access networks and server networks are two other important applications for the method.

E. 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. These functions could either be implemented in the existing edge routers or as stand-alone systems. Traffic meters act as packet filters designed to count packets (and bytes) belonging to the kind of traffic subjected to monitoring by the operator. These filtering and counting functions are coordinated with the generation of dedicated monitoring packets, inserted between blocks of ordinary data packets as shown in Figure 2. Alternatively the distance between the monitoring packets could be implemented as time intervals, which is further discussed in section V.

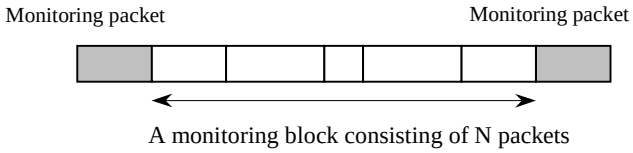

Figure 2: Two monitoring packets enclose a monitoring block that consists of N user packets on the average.

The sending monitoring edge router (or stand-alone system) generates monitoring packets, for example between every N user packet on average or within specific time intervals. These monitoring packets can be used in two different ways: either to carry monitoring information (counter values and timestamps), or to synchronize storage of these values in the corresponding monitoring edge routers (or stand-alone systems). In the latter case the critical synchronization process between the counter functions and the generation of monitoring packets is no longer necessary.

The sending monitoring system inserts the number of packets (and bytes) sent so far and a timestamp into the monitoring packet (or stores these values locally). The receiving monitoring system detects the monitoring packets, inserts (or stores locally) the number of packets (and bytes) received so far and a timestamp.

The size and character of the monitoring block will be discussed further in sections E and F.

F. Measurement Results

It is possible to obtain the following results using the method.

- The number of lost packets between sending and receiving nodes 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 contain lost packets and the number of monitoring blocks that are loss-free. Alternatively this could be expressed as loss-free and loss seconds.
- Samples of the packet delays, and delay variations, between sending and receiving nodes.
- An estimate of the utilized capacity (throughput) between each pair of sending-receiving edge nodes. This requires that the packet length can be estimated.

The results could be based on continuous 24-hour measurements or limited to busy periods.

G. Simulations Based on Traffic Data

The described method has also been simulated in Matlab® based on traffic from the RIPE-NCC test traffic boxes [5]. The traffic data are recordings of traffic between test boxes in Copenhagen, Stockholm and Helsinki during 10 days in November 2000. The simulations show that the method provides acceptable estimates of the average delay and delay variations, packet losses and throughput [1].

IV. IMPLEMENTATION

A. A Prototype for Embedded Monitoring

In this section we describe an implementation of embedded monitoring where these functions constitute an integral part of a router.

The main idea is to implement counters related to packet filters that keep track of the number of incoming and outgoing packets, and insert dedicated monitoring packets between blocks of N data packets on the average. These functions could be realized either on the network interface cards or in the kernel.

The test environment in Figure 3 consists of two hosts that send and receive UDP packets, two edge routers and a computer that emulates losses and delays in a core network. The edge routers communicate by means of an IP tunnel. PC-based routers running Linux operating system have been used. The work is documented in [6].

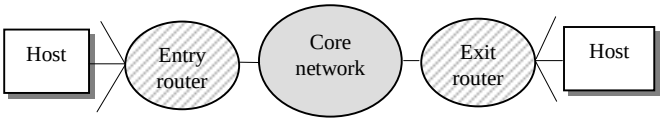

Figure 3: The figure shows the test environment for the prototype implementation. Losses and delays in a core network are emulated using Nistnet.

The embedded monitoring functions within the edge routers consist of a packet filter, counting functions, a process that sends (receives) monitoring packets and facilities the storage and processing of monitoring data. Iptables in Linux [7] has been configured as a filter that selects the relevant packets to be monitored. Netfilter [8] is then used to forward the packets from the kernel to a queue in user space, where the monitoring

functions are implemented as a C program. The counters keep track of the cumulative number of sent (received) packets that pass through the filter. The monitoring packets are inserted between blocks of N user data packets. These dedicated packets carry timestamps, the number of sent and received packets so far, and a sequence number. Packet losses and delays in the core network are emulated using Nistnet [9].

B. The Monitoring Functions

Figure 4 shows a brief flow diagram of the monitoring procedure in an entry-monitoring node. The main functions in respective modules are explained below.

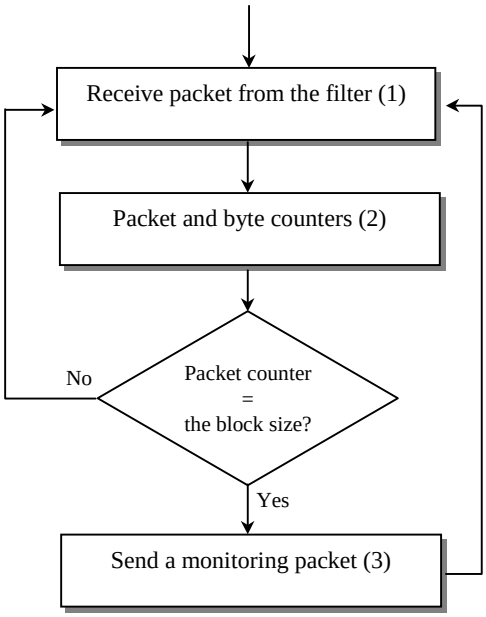

Figure 4: A flow diagram that shows the structure of the monitoring functions in an entry-monitoring node. The numbers within brackets refer to the comments below.

1. Packets that match the filter conditions in the entry router, implemented in Iptables, are sent from the kernel to a program in user space that handles the monitoring functions.
2. The cumulative values of the packet counter and the byte counter are incremented with respect to the incoming packet.
3. If the packet counter has reached N (the size of the monitoring block) a monitoring packet is generated. A timestamp, the current values of the packet counter, the byte counter and a sequence number are inserted into the monitoring packet. The data packet and the monitoring packet are returned to the kernel, and forwarded according to the ordinary routing functions.

The flow diagram in Figure 5 illustrates the monitoring functions in an exit-monitoring node.

1. Packets that match the filter conditions in the exit router, implemented in Iptables, are sent from the kernel to a program in user space that handles the monitoring functions.
2. The cumulative values of the packet counter and the byte counter are incremented according to the incoming packet.
3. If the exit router receives a monitoring packet a timestamp and the current value of the counters are inserted, and the packet is sent back to the entry router.

If the clocks in the entry and exit nodes are synchronized the one-way delay can be calculated. Otherwise, as in current version of the prototype, the timestamp at the exit node registers the arrival times for the monitoring packets. When the monitoring packet has returned to the entry node a timestamp is inserted, in order to compute the round-trip time.

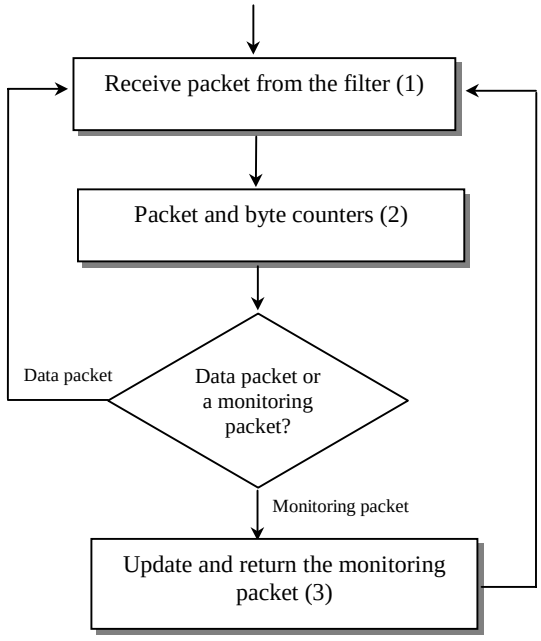

Figure 5: A flow diagram of the monitoring functions in an exit node. The numbers in brackets refer to the comments above.

C. The Monitoring Packets

The monitoring packets used in the prototype contain the following information fields.

- A sequence number of the monitoring packet.
- The cumulative number of packets sent from the entry router.
- The cumulative number of packets received by the exit router.
- The cumulative number of bytes received by the entry router.
- The cumulative number of bytes sent from the entry router.
- A timestamp at the entry router.
- A timestamp at the exit router.
- A timestamp at the entry router when the monitoring

packet returns.

D. The Estimated Parameters

The following parameters are measured and estimated.

- The total number of lost packets and the packet loss ratio during the measurement period.
- The number of lost packets and the loss ratio for each monitoring block.
- The length of the loss-free periods and the loss periods, and also the number of lost packets and the loss ratio for the loss periods.
- The mean length of the packets in each monitoring block.
- The round-trip times for the monitoring packets.
- The arrival times at the exit-monitoring node for the monitoring packets.
- Throughput (in b/s and packets/s) for the entire measurement period and for each monitoring block.

The packet rate is limited to around 8 Mb/s (4000 packets/s and a mean packet length of 250 bytes) in the current tests. It should be emphasized that the goal has not been to optimize the speed, but rather to demonstrate the basic concepts of the method.

E. Results from Measurements

The performance of the monitoring functions has been evaluated for different levels and distributions of packet losses and delays, and various sizes of the monitoring block. The test results can be summarized as follows.

- Packet losses are measured accurately. The number of losses inserted by Nistnet agrees with the number of lost packets detected by the prototype.
- Besides the overall packet loss ratio the distribution of losses in terms of the length of the loss-free periods and the loss periods can be obtained.
- The accuracy of the delay measurements depends to some extent on the operating system. The effect on the average values is however limited.
- The results using different distances between the monitoring packets (i.e. the length of the monitoring block) agree with what was found in the simulations presented in [1]. The block size can be increased with preserved accuracy for the estimates of the average delay provided that the measurement periods (the number of samples) are increased correspondingly.
- The throughput has also been measured accurately for the entire measurement period as well as for the individual monitoring blocks.

The relation between the block size and estimates of delays, throughput and losses will be discussed further in section V. An alternative approach where the distance between the monitoring packets is defined in terms of time intervals will also be considered.

The Figures 6-12 illustrate some examples of measurement results using the prototype. The size of the monitoring block is 1000 packets in all referred cases. Figure 6 shows the distribution of the round-trip times and Figure 7 the round-trip time variation. The distribution of the throughput can be seen in Figure 8. The maximum throughput is around 310 kb/s and the mean value is 260 kb/s in that example. Figure 9 and 10 show the distribution of the length of the loss-free and loss periods respectively. The distribution of the number of lost packets in the loss periods can be seen in Figure 11. Finally Figure 12 illustrates the distribution of the packet loss ratio per loss period.

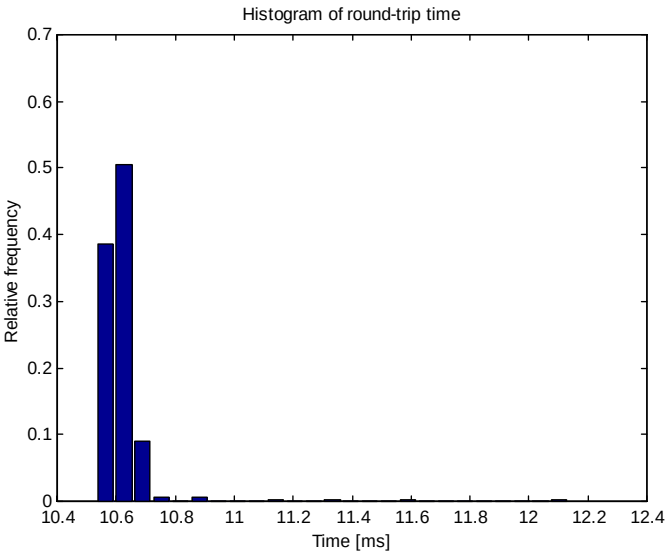

Figure 6: Distribution of the round-trip times with a mean value of 10.6 ms and a maximum of 12.1 ms.

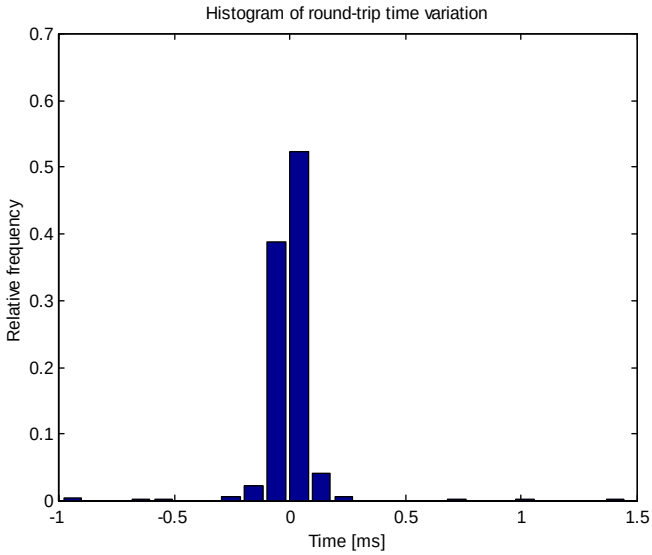

Figure 7: Distribution of the round-trip time variation.

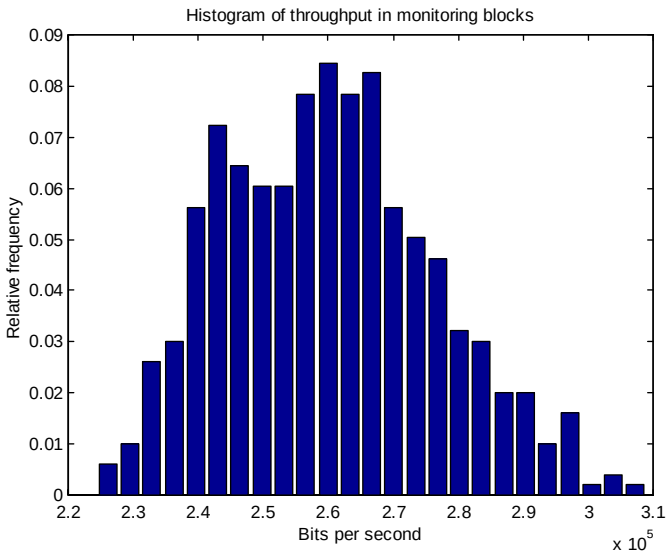

Figure 8: Distribution of the throughput based on the monitoring blocks.

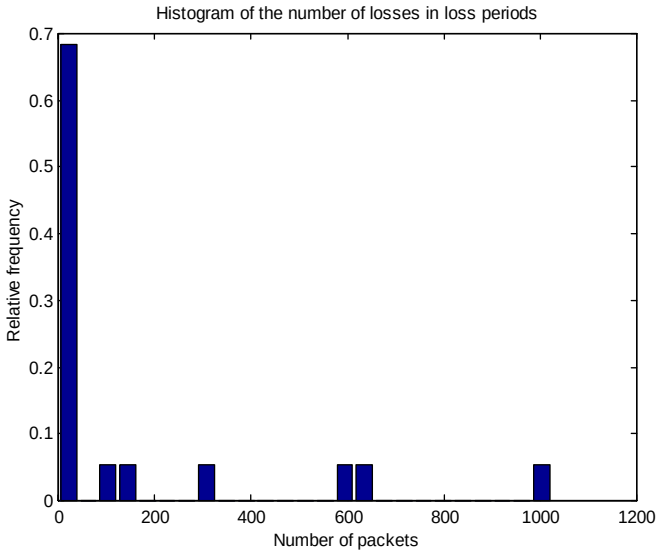

Figure 11: Distribution of losses in the loss periods.

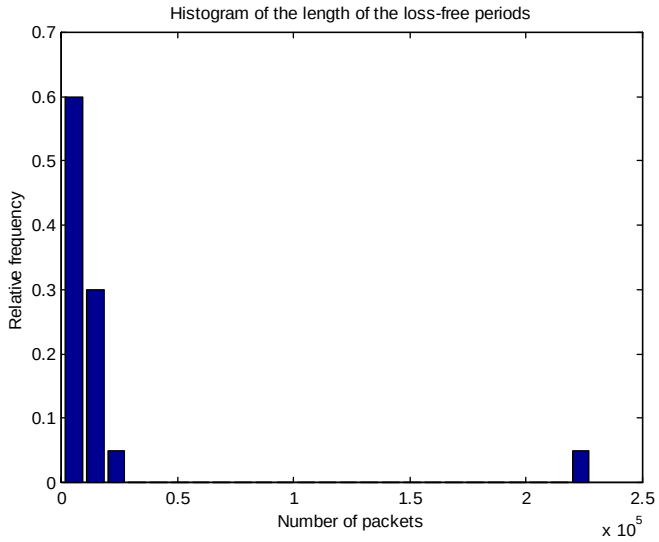

Figure 9: Distribution of the length of the loss-free periods.

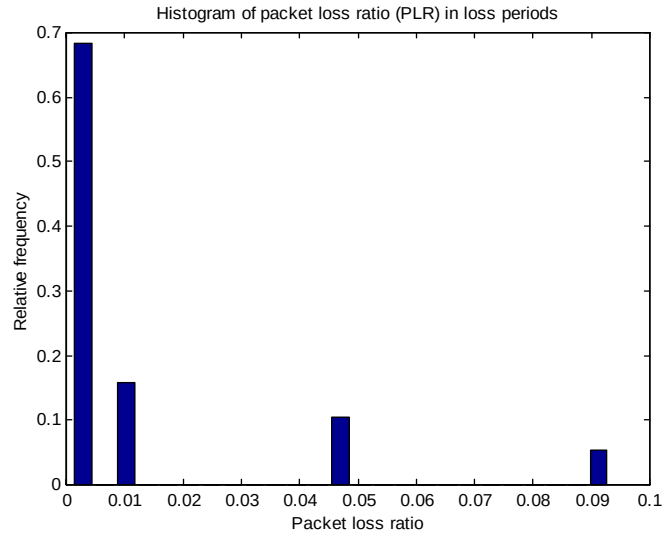

Figure 12: Distribution of the packet loss ratio in the loss periods.

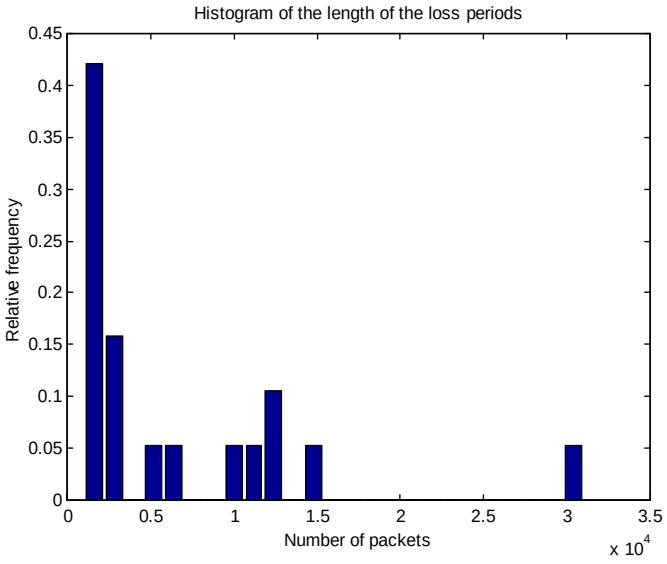

Figure 10: Distribution of the length of the loss periods.

V. The Size of the Monitoring Block

There are several important issues related to the nature and size of the monitoring block. To maintain a distance of N data packets between the monitoring packets requires an effective synchronization between the counter functions and the creation and transmission of the monitoring packets. During periods of heavy traffic this could also be difficult to accomplish. One solution would be to let the block size vary around a mean value, thereby allowing larger block sizes during heavy traffic load and other conditions. With this procedure unwanted synchronization with periodic traffic patterns could also be avoided.

Another possibility, mentioned in a previous section (III.E), is to simplify the creation of the monitoring packets and reduce their role merely to trigger local storage of counter values and timestamps in the respective monitoring nodes. This solution, discussed further in the next section (VI.A), also has the ad-

vantage to facilitate a stand-alone implementation as an alternative to the embedded monitoring system described so far.

Instead of expressing the size of the monitoring block in terms of the number of packets, it might be favorable to determine the distance between monitoring packets as fixed time periods, possibly varying around an average value.

A. The Block Size as a Time Period

The time distance between monitoring blocks can be determined based on the average throughput, or the peak value of the throughput, for the monitored connection.

Table 1 shows the frequency of monitoring packets (col. 3) and the corresponding block size (col. 2) for some given levels of throughput (col. 1). A mean packet size of 250 bytes has been used in the calculations.

TABLE 1: FREQUENCY OF MONITORING PACKETS FOR SOME EXAMPLES OF BLOCK SIZES AND THROUGHPUT

Throughput (Mb/s)	*Block size (packets)*	*Monitoring packets/s*
1	2500	0.2
10	2500	2
100	2500	20
1000	2500	200
1	1000	0.5
10	1000	5
100	1000	50
1000	1000	500
1	500	1
10	500	10
100	500	100
1000	500	1000

B. A Procedure to Determine the Frequency of the Monitoring Packets

If a time-dependent scheme for the generation of monitoring packets is preferred the following procedure is suggested.

- Send monitoring packets, e.g. every second during a time period, in order to determine the average (or peak) throughput B.
- Determine the distance between the monitoring packets based on the requirements for estimation of the average delay.
  1. Determine the number of samples n needed to obtain the desired confidence level α and interval d. The number of samples is then given by $n=z^2s^2/(d/2)^2$, where s is an estimate of the standard deviation and z can be retrieved from statistical tables of the normal distribution (given α) [17].
  2. Decide the minimum time period T for which the estimates should be valid.
  3. $D=T/n$ is then the largest acceptable time interval between the monitoring packets. $(BT)/(nl)$, where l is the average packet length, gives the corresponding size of the monitoring block expressed in the number of packets.
- Determine the distance between monitoring packets based on the requirements for estimation of the loss-free and loss periods.
  1. Determine the desired minimum length L_t of the loss (loss-free) periods in terms of a time interval. Alternatively, let K be the maximum number of packets that are accepted between two consecutive losses in a loss period. $K+2$ is then the length of the monitoring block expressed in terms of the number of packets [1]. The corresponding L_t can be computed using B.
  2. If $L_t<D$ then $D:=L_c$.
- Determine the distance between monitoring packets based on the requirements for estimation of the peak value of the throughput.
  1. Determine the desired time base, L_c, for the estimation of peak throughput.
  2. If $L_c<D$ then $D:=L_c$.

1) An Example

Let the time period T=5 minutes, the confidence level α=0.95 and the confidence interval d=0.4. The standard deviation s is not known and has to be estimated. If the range R can be approximated, the standard deviation s is $R/4$, according to a rule of thumb [21]. The number of samples becomes $n=z^2s^2/(d/2)^2$=1536, with z=1.96 (for α=0.95) and s=4. For T=300 seconds the time distance between the monitoring packets will be approximately 200 ms, i.e. five monitoring packets per second. With a maximum utilized capacity of 100 Mb/s the maximum length of the monitoring blocks would be 10000 packets, assuming an average packet length of 250 bytes. This means that the resolution of the loss and throughput metrics have a lower bound at 10000 packets (or 200 ms). Consequently, a frequency of e.g. 100 monitoring packets per second reduces the maximum size of the monitoring block to 500 packets on the average.

For T=1 hour the time distance between the monitoring packets will be 2.34 seconds, and the corresponding block size approximately 117200 packets (250 bytes on the average). The choice of T depends on the time period for which the process is stationary, but also on the response time required by the management system. Considering previous experiences from network operation and monitoring systems T=5 minutes seems to be an appropriate choice.

C. Controlling the Size of the Monitoring Blocks

The block size is a crucial parameter that determines the accuracy and granularity of the estimates, and can be used to adapt the measurements to an operator's objectives. The goal could be to maintain a constant block size, or to adapt the block size to certain parameters. The procedure described in the previous subsection can be seen as a feedback loop to control and adapt the block size, which is illustrated in Figure 13.

The input P consists of a set of parameters such as the con-

fidence interval and the confidence level (d and α), the time period T, the loss parameter K and the time base L_t for estimation of the throughput. The control signal E, the difference between the reference value Y_{ref} and the actual value Y, is the input value to the function that generates monitoring packets. The detailed design of each block in this model is however for further study.

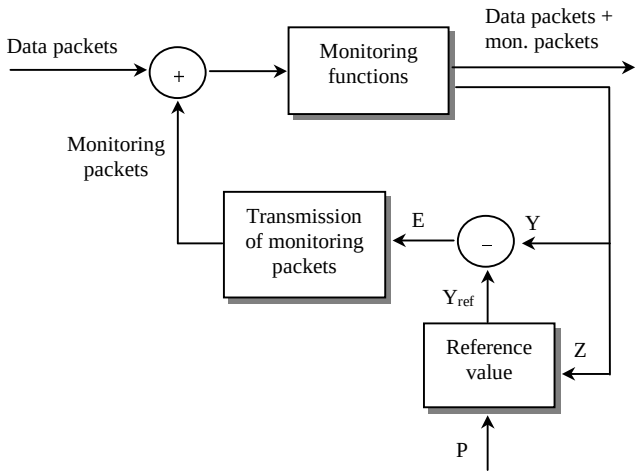

Figure 13: A feedback loop for control of the size of the monitoring block.

The monitoring functions in Figure 13 are mainly packet filters and counters. The output Y is the actual measured value of the block size and Z is the actual measured statistics for throughput, packet losses and delays. The reference Y_{ref} is computed according to the algorithm outlined in the previous subsection, based on the operator's set of parameters P and the actual measured values of the performance parameters Z.

VI. Discussion and Future Work

The implemented prototype is quite simple and has therefore a number of limitations and shortcomings that can be improved.

- The performance can be increased considerably by implementing the monitoring functions as a kernel module, an Iptables target. This would presumably result in speed limits of the same magnitude as firewalls implemented by means of Iptables.
- The accuracy of the delay measurements can be improved if real-time Linux is used instead of Redhat, and if the timestamping is implemented on the interface cards and not in the kernel or user space [11].
- Synchronization of the clocks in the monitoring nodes, using e.g. GPS, would enable one-way delay measurements.
- Scalability in terms of the number of concurrent filters and monitored VPNs is an important implementation issue, albeit not studied yet.
- One important future task is to apply the method to IP-VPNs based on MPLS (Multi-Protocol Label Switching). A major problem is how to handle the merging of labels in MPLS networks. Although MPLS makes IP more connection-oriented, it sometimes tends to hide the source edge node that a packet is sent from and thereby complicate the monitoring task.
- Another issue worth studying is whether measurements during the busy hour (or period) are sufficient or if continuous 24-hours monitoring is to prefer.

Several other open issues are also considered in [1]. However, in this section we focus on four other issues. Some examples of service level agreements (SLAs) that can be monitored using the method (subsection A) are described. Secondly, an implementation of the method that is not an embedded part of an edge router but rather a stand-alone monitoring system is outlined (B). Next, the need for a common measurement infrastructure (C) is considered, and finally some comments about the method in relation to other approaches (D).

A. Monitoring of Service Level Agreements

Performance monitoring is of special concern when the traffic is divided into different service classes with certain constraints and guarantees regarding packet losses, delays and throughput. The monitoring functions (packet filters and meters) can be configured to distinguish between different types of traffic and service classes. The method can support management of service level agreements regarding various levels of losses, delays and throughput. Some examples of guarantees for quality of service in SLAs that are feasible to verify are given below.

- A service class with guarantees for low levels of packet losses, e.g. the packet loss ratio < x % and the loss-free periods (or blocks) > y seconds (or # packets) on the average. Alternatively a parameter K that determines the length of the loss-free and loss periods can be specified.
- A service class with guarantees for low levels of packet delays, e.g. a mean delay < z ms (with specified confidence level and interval), a maximum delay below u ms (below the 95 % percentile), and a mean delay variation for the monitoring packets less than v ms (below the 95 % percentile).
- The utilized capacity (throughput) can be measured for all service classes, with different degrees of resolution.
- The measurements can be configured per service class and per VPN.

B. Stand-Alone Monitoring Systems

In the embedded solution, described in the previous sections, packet filters and other monitoring functions are integral parts of the routers that also act as monitoring nodes. This design is also advantageous if the measurements will be used as input and feedback information for traffic control. However, a drawback is that an operator has to rely on router vendors in deploying the monitoring systems.

Stand-alone implementations would give an operator a larger degree of freedom and independence when designing

and deploying the monitoring functions. Figure 14 shows two monitoring systems, A and B, that consist of passive as well as active components. Traffic meters identify the relevant flows and keep track of the number of packets and bytes passing through. The active part of the system generates monitoring packets, for example according to the scheme outlined in the previous section.

System A and system B filter packets and count the number of packets and bytes for the type of traffic subjected to monitoring. When the traffic meters in A and B receives monitoring packets, the intermediate cumulative number of user packets and bytes and a timestamp are stored locally along with the monitoring packet identification. One could say that a monitoring packet acts as a probe that triggers a local storage of measurement data. For one-way delay measurements the clocks in the systems A and B have to be synchronized.

In the case described above the distances between the monitoring packets are time periods, where no exact synchronization between the counters and the transmission of monitoring packets is required. The time distances between the monitoring packets could be controlled in a way depicted in Figure 13 in section V.

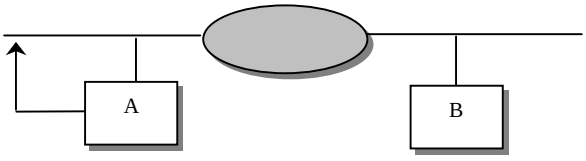

Figure 14: Two stand-alone monitoring systems, A and B, that consist of traffic meters, counters and other and monitoring functions.

C. A Common Measurement Infrastructure for Multi-Purpose Management Applications

Performance measurements and monitoring systems are necessary for a whole range of an operator's management functions. Besides verifying service level agreements, measurements and feedback information are indispensable for traffic control, operation and maintenance, dimensioning and planning, as well as usage and billing systems. To design a common measurement infrastructure is a challenging problem. One example is a control-based approach that combines performance monitoring and traffic control of capacity in IP-based VPNs. The difference between the actual measured values (of throughput, packet losses and delays) and the expected reference values will be the input value for the traffic control algorithm that allocates resources for each VPN.

D. Relationship to Other Approaches

1) The Network Level

The presented method is a combination of the traditional active and passive methods in traffic measurements. Consequently it can benefit from the advantages of from both techniques, but of course also share some of their shortcomings. The delay measurements rely on a sampling procedure carried out by means of the monitoring packets, while packet losses and throughput are measured passively via traffic meters. Passive methods often require massive storage and processing capacity, especially for measurements in high-speed networks. Traffic meters, however, reduce this problem considerably. Some of the advantages of the presented method are.

- The three main categories of performance parameters (packet delays, packet losses and throughput) can be obtained at the same time using a single method.
- The monitoring blocks (expressed in terms of time periods or number of packets) make it possible to describe the loss process and throughput in more detail than overall long-term metrics.
- The concept of monitoring blocks also enables an adaptation of the measurements to an operator's requirements regarding accuracy and resolution.
- Packet filters and traffic meters are flexible tools to carry out in-service monitoring adapted to e.g. different types of traffic, QoS classes, protocols, port numbers and applications.

2) Combining End-to-End and Edge-to-Edge Measurements

This paper is focused on monitoring at the network (IP) level between edge nodes. However, monitoring of services and applications are becoming increasingly important. In recent years management of service level agreements has attracted a lot of attention. SLA management often requires measurements of performance parameters such as response times and availability at the application level. Similar techniques as in network measurements have also been implemented in this field. Active as well as passive approaches are used for monitoring of servers and applications e.g. between clients and servers in an end-to-end fashion. One initiative for an industry standard is Application Response Measurement (ARM) from Hewlett Packard and IBM, nowadays further developed by Open Group and Computer Measurement Group [19].

Since the method in this paper is based on a combination of packet filters and traffic meters on one hand and monitoring packets on the other, it seems feasible to extend its scope to higher protocol layers. The basic idea is to find an appropriate balance between an edge-to-edge approach and an end-to-end approach. A combination of these techniques makes it easier for an operator to discern whether performance deterioration is mainly caused by the network or by applications on servers and hosts.

A media streaming server could for example handle the monitoring functions, i.e. maintain packet and byte counters and insert monitoring packets after a certain block size or time period. Provider edge routers, customers' edge routers or stand-alone systems may accordingly implement the exit monitoring functions. Another way to monitor real-time communication end-to-end is to capture the control packets used by RTCP (Real-Time Control Protocol) for RTP/UDP streams.

However, since an operator's responsibility often ceases beyond the provider edge nodes, end-to-end measurements are often insufficient. A synthesis of end-to-end and edge-to-edge measurements is therefore preferable.

VII. Conclusions

In this paper we have presented a simple prototype implementation of a method for performance monitoring, especially in IP-based virtual networks, that can be seen as a combination of the traditional passive and active approaches. The results using this method are estimates of losses (the overall packet loss ratio as well as the dispersion of losses), delays and delay variations, and throughput between edge nodes. The size of the monitoring block is identified as an important parameter that determines the accuracy and degree of resolution of the measurements. Embedded monitoring functions in existing edge routers as well as stand-alone implementations have been considered. An extension of the measurements to higher protocol layers, as well as a combination of edge-to-edge network measurements and end-to-end application measurements have also been recognized as an important issue.

VIII. Acknowledgements

I would like to thank my supervisor Professor Gunnar Karlsson (KTH IMIT) for his encouraging support and constructive criticism, and Per Nordén at Ki Consulting for his fine work with the prototype.

IX. References

[1] 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.

[2] Lindh T.: "Embedded Monitoring of QoS Parameters in IP-Based Virtual Private Networks", Proceedings of Integrated Network Management VII (IM2001), Seattle, 14-18 May 2001.

[3] Beige M., Jennings R., Verma D.: "Low Overhead Continuous Monitoring of IP Network Performance", Proceedings of Symposium on Performance Evaluation of Computer and Telecommunication Systems, SPECTS'99, Chicago, July 1999.

[4] Quality of Service Methodologies and solutions within the service framework: Measuring, Managing and Charging QoS, EURESCOM Project P906, January 2001.

[5] Georgatos 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] Nordén P.: "Monitoring of Performance Parameters in IP Networks", Degree project, KTH, 2001.

[7] Linux 2.4 Packet Filtering HOWTO, http://netfilter.filewatcher.org/unreliable-guides/packet-filteringHOWTO/index.html

[8] The Netfilter Project: Packet Mangling for Linux 2.4: http://www.netfilter.kernelnotes.org.

[9] Nistnet home page: http://www.antd.nist.gov/itg/nistnet/.

[10] NeTraMet home page: http://www2.auckland.ac.nz/net/NeTraMet/

[11] Micheel J., Donnelly S., Graham I.: "Precision Timestamping of Network Packets", ACM SIGCOMM Internet Measurement Workshop, San Francisco, USA, November 1-2, 2001.

[12] Paxson V., Adams A.K., Mathis M.: "Experiences with NIMI", Proceedings of Passive and Active Measurements (PAM) 2000, Hamilton, New Zealand, April 3 and 4, 2000.

[13] Kalidindi S., Zekauskas M.: "Surveyor: An Infrastructure for Internet Performance Measurements", INET'99, San Jose, June 1999.

[14] McGregor T., Braun H-W., Brown J.: "The NLAR Network Analysis Infrastructure", IEEE Communications Magazine, Vol. 38, No. 5, May 2000.

[15] White Paper - NetFlow Services and Applications: http:// www.cisco.com/warp/public/cc/pd/iosw/ioft/neflct/tech/napps_wp.htm.

[16] "SLA Management Handbook", TeleManagement Forum, June 2001.

[17] Thompson S.K.: "Sampling", Wiley & Sons, 1992.

[18] "The challenge of enabling QoS in IP networks", Operax AB, October 2001.

[19] Open Group home page: http://www.opengroup.org/management/arm.htm.

[20] Service Assurance Agent: http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120t/120t5/saaoper.htm.

[21] Körner K., Wahlgren L.: "Statistisk dataanalys", p. 166, Studentlitteratur, Stockholm, 2000.