Adaptation of Real-time Temporal Resolution for Bitrate Estimates in IPFIX Systems

Rosa Vilardi, Luigi Alfredo Grieco, Gennaro Boggia
DEE - Politecnico di Bari - Italy
Email: {r.vilardi, a.grieco, g.boggia}@poliba.it

Chadi Barakat
INRIA - Sophia Antipolis, France
Email: chadi.barakat@inria.fr

***Abstract*—Packet sampling can greatly reduce traffic measurement overhead in high-speed broadband networks. At the same time, this operation introduces estimation errors that have to be carefully handled to ensure a reasonable measurement accuracy. Recently, a frequency-based approach has been proposed to catch the impact of such errors in bitrate estimation of a generic IP traffic flow, binned at both small and high time-scales. In particular, a closed-form expression for the signal-to-noise ratio has been derived as a function of the packet sampling probability, the bin size, and some basic information about the flow (i.e., first and second order moments of the packet size, and long term average packet-rate). In this work, we adopt such a model to design a real-time algorithm, that sets the IPFIX counter export timers in order to grant, to each flow, a target estimation accuracy. Computer simulations carried out using real packet traces have demonstrated the effectiveness of the proposed approach.**

I. Introduction

Advanced tools for traffic engineering, attack/intrusion detection, QoS monitoring and network tomography are becoming fundamental to plan and control the activities of complex communication networks [1]–[4]. They require sophisticated traffic measurement systems in order to estimate network properties and application related parameters.

In high speed networks, packet sampling techniques are usually adopted by network operators to reduce the overall amount of packets to capture and process [5]–[7]. These approaches consist in capturing a subset of packets, used to infer the original traffic properties. Obviously, the reduction of the measurement overhead comes at the expense of the estimation accuracy and, as a consequence, a fine tuning of the sampling strategy is required to satisfy the target measurement requirements [8]. The mostly known sampling pattern consists of a random selection of packets at the incoming interfaces of routers with some predefined and homogeneous probability. Such a probability, p, is called as *sampling rate* and is set by operators according to some policy, for example a constant function of the bitrate of links [2].

Flow level models are used to characterize, within an aggregate traffic, sub-streams of packets sharing a set of common properties, such as TCP/UDP ports and/or portions of both sender and receiver IP addresses. In this way, the overall traffic carried over a backbone link can be described in terms of the properties of its composing flows. For example, Cisco NetFlow solution[1] satisfies this necessity to classify traffic flows belonging to the same aggregate stream. In particular, flow records are generated to keep track of principal measurement parameters observed for each monitored flow. The IETF Working Group, starting from Cisco NetFlow v.9 system guideline, has brought forward a standardization process to define specifications for an exporting traffic flow information protocol, named IP flow information export (IPFIX) [9], [10].

Network measurement systems proposed so far have been conceived under the implicit assumption that flow properties should be evaluated with a coarse time resolution [11]. This implies that only low frequency components of the traffic can be captured. Nevertheless, high frequencies carry information that can be very precious for a correct detection of traffic anomalies [1]. The same remark applies to traffic engineering where decisions on rerouting the traffic are taken by network administrators based on variations in the traffic bitrate.

To face this problem, herein, we propose an IPFIX compliant system for real-time traffic monitoring, named LEMON (Lightweight Enhanced MOnitoring for backbone Networks). Given a target estimation accuracy, LEMON is able to adapt the time resolution analysis of traffic flows by taking into account both flow properties and the sampling probability p. To accomplish this task, LEMON exploits recent results reported in [12], expressing in close-form models the accuracy of bitrate measurements taken from a sampled stream. The effectiveness of LEMON has been demonstrated using real traffic traces from the MAWI project [13].

The rest of the paper is organized as follows: in Section II an overview of IPFIX is provided. Section III describes all details of the LEMON framework. Section IV presents the experimental evaluation of the LEMON performance. Finally, the last Section draws conclusions and forecasts future research.

II. Overview of the IPFIX Protocol

IPFIX is an IETF standard protocol [10] created to support flow-based IP traffic measurement systems. Its reference architecture is composed by interacting IPFIX *devices* and *collectors* (see Fig. 1).

In IPFIX, a flow is defined as a set of IP packets which share common properties (i.e., the *flow key*) referred, for example, to

[1]For details, see “Cisco IOS NetFlow”, available on-line at http://www.cisco.com/web/go/netflow.

the packet header or the transport header (source/destination IP address, source/destination transport port, transport protocol, packet length), or belonging to the characteristics of the packet itself.

Within an IPFIX device [14], the *metering* process is in charge of managing the timestamp, the sampling, the classification, and the organization of IP traffic information, taken from packets relieved at an *observation point* (i.e., router interface). A local database is used to maintain Flow Records, containing flow statistics derived from packet processing.

Under certain conditions, useful to an effective management of *flow record table*, some flows are considered expired. Such conditions include (but are not limited to): traffic overloads, which may generate too much new Flow Record entries; TCP headers with FIN or RST flags set (which indicate a TCP session is going to end); and IPFIX timeouts configured by the metering process. Expired flow records are exported to one or more collectors by an *exporting* process, using an IPFIX message. An IPFIX message consists of a header, which provides basic information about the message itself (protocol version, message length, sequence number), followed by one or more sets. Three set types are defined: *data*, *template*, and *option template* [10].

There are several kinds of IPFIX messages, depending on the measurement task. Within them, flow records attributes are encoded as *information element* data records into data set fields, using information model guideline [15]. Settings of metering process (such as sampling rate, flow timeout interval, and so on) are inserted as *control information* data records. Each message may contains also a detailed description of the structure and the semantics of the data record embedded (e.g., information element or control information) into the template record field [10]. Fig. 2 shows an example of fields for template and data sets in a possible IPFIX message (the header field is omitted to simplify the figure). An IPFIX message could also contain a set of different data records with the corresponding template set.

The exporting process encapsulates IPFIX messages at the transport layer, using TCP or UDP depending on the kind of data they contain (TCP is mandatory only for control information messages).

At the end, a collector (e.g., a remote access router) receives IPFIX messages from different exporting processes and decodes flow record information elements. These information are stored in its database, which is made accessible to network operators. The functionalities of the IPFIX architecture have been enhanced in the Packet SAMPling framework (PSAMP) [16] by extending the information model specifications [17] in order to handle packet payload too.

III. LEMON Framework

LEMON is a real-time traffic monitoring algorithm for IPFIX systems. It has been conceived for monitoring applications targeting a fine time resolution, such as anomaly detection, path characterization and network tomography [2]–[4].

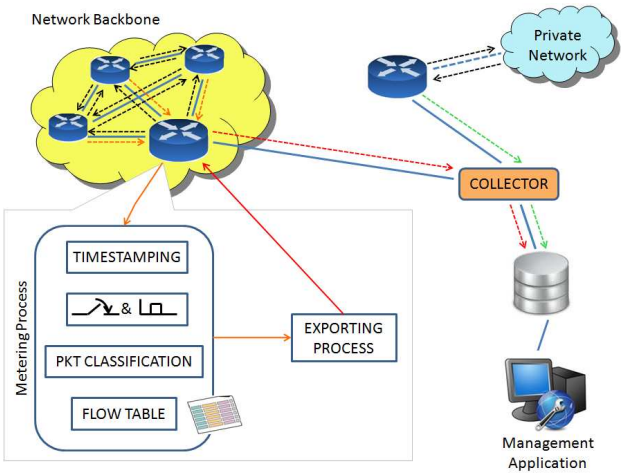


Fig. 1. IPFIX Architecture.

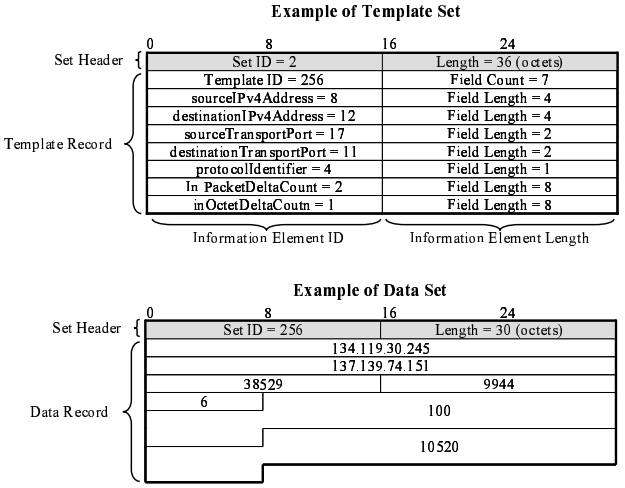


Fig. 2. Examples of IPFIX export template and data sets.

LEMON is a highly scalable technology able to set in real-time some important working parameters such as the sampling probability, the kind of traffic to analyze, the time resolution of measurements, and, more in general, the granularity of traffic monitoring operations. Specifically, LEMON is able to set IPFIX counter export timers in order to grant, to each flow, a target estimation accuracy.

To fulfill this objective, LEMON leverages on the recently proposed frequency analysis of packet sampling [18]. The traffic stream, sampled with a uniform sampling probability, p, is measured in terms of number of bytes sent from a sender to a receiver and averaged over a time interval window, T, that is the *bin time*. Then, these measures are compensated by dividing them by the sampling probability p, to infer the original rate of traffic in the considered bin. An estimation error in valuation of traffic bitrate signal is introduced by sampling a packet stream. It can be modeled as an aliasing affecting the signal spectrum in the frequency domain [18]. Thus, it is possible to recover information about the starting signal bitrate also from a sampled version, with different level

of accuracy and time resolution, using a particular closed-form expression that relates the Signal-to-Noise Ratio (SNR) associated to the traffic spectral density, to some important parameters, as the sampling probability p and the bin time T [12]. In what follows, we will show how theoretical results derived in [12] are used in LEMON to dynamically tune the measurements time resolution.

A. Model for the Signal-to-Noise Ratio

In this subsection, we briefly summarize the SNR closed-form models derived in [12] to catch packet sampling effects. We consider a discretized time axis: each time slot has a regular size t_0, smaller than the bin size T. Under this assumption, no more than one packet can be transmitted in each short time slot. In practice, this t_0 corresponds to the transmission time of the smallest packet over the monitored link. One can also see it as the minimum possible time between two consecutive packets over the monitored link.

Our metrics are function of the sampling rate p, the bin size T, the probability to find a busy slot in the original traffic p_{bs}, and the first and second order moments of the packet size, which will be referred to as $\overline{D}$ and M, respectively. Note that, by definition, $p_{bs} = P\{D(k) > 0\}$ where $D(k)$ is the discrete signal which models the traffic packet size in the slotted time axis. All these parameters can be calculated from the sampled traffic without having access to the original traffic, hence the interest of our approach.

In [12], two SNR models were derived: one assuming a constant packet size and the other one explicitly taking into account packet size variability by using its first (i.e., its mean) and second order moments. They can be summarized by the following equation:

$$SNR = \frac{p_{bs}\overline{D}^2 + (M - p_{bs}\overline{D}^2)0.89 \cdot t_0/T}{\frac{1-p}{p}(M - p_{bs}\overline{D}^2)0.89 \cdot t_0/T}. \quad (1)$$

Note that Eq. (1) derived for variable packet size is still valid for constant packet size by simply considering $\overline{D}^2 = M$, because a constant packet size means, as well known, a zero variance. Thus, eq. (1) becomes:

$$SNR = \frac{p_{bs} + (1 - p_{bs})0.89 \cdot t_0/T}{\frac{1-p}{p}(1 - p_{bs})0.89 \cdot t_0/T}. \quad (2)$$

Our SNR models calculate, for a given sampling probability p and an averaging bin time window T, the amount of error in each frequency band for the traffic rate signal. It allows us to trade off sampling overhead with frequency resolution [12].

B. LEMON Algorithm

As described above, LEMON is designed to be fully integrated in a IPFIX monitoring device. Following the IPFIX architecture scheme, we can summarize LEMON actions in three main processing operations: working parameters setting, flow bin counter management, and data exporting.

Basically, LEMON sets the exporting timer of each flow counter in order to grant for an SNR equal to a threshold value SNR_{th}. Typically SNR_{th} is imposed larger than 10 to achieve a high estimation accuracy. Notice that, the exporting timer plays the same role of the binning window T. As a consequence, given a predefined sampling probability p, by using Eq. (1), the resulting exporting period counter for the i-th flow can be expressed as follows:

$$T_i = \left[\frac{1-p}{p}SNR_{th} - 1\right] \cdot \frac{(M_i - p_{bs_i}\overline{D}_i^2)0.89t_0}{p_{bs_i}\overline{D}_i^{\,2}}, \quad (3)$$

where, with reference to the i-th flow, $\overline{D}_i$, M_i, and p_{bs_i} are the first order packet size, the second order packet size, and the probability to find a busy slot. It is worth to note that p and t_0 parameters does not depend on any specific flow but are set for the aggregate traffic stream passing through a router interface.

In detail, LEMON starts by initializing all counter exporting intervals T_i to a common default value T_d. After that, each time the timer of the i-th counter expires, LEMON performs the following operations:

1) exports the i-th counter in a specific IPFIX message;
2) updates the estimates for $\overline{D}_i$, M_i, and p_{bs_i} parameters;
3) sets the next value of T_i according to Eq. (3);
4) resets the counter;
5) starts again the counter exporting timer.

In order to estimate $\overline{D}_i$, M_i, and p_{bs_i}, LEMON accounts for all sampled packets of the i-th flow.

IV. Performance Evaluation

In this section, we evaluate the effectiveness of LEMON using two distinct packet traces from the MAWI project collected at two trans-pacific 150 Mbps links during December 2005 and January 2009[2]. Main traffic parameters have been summarized in Tab. I. Each trace lasts 15 minutes and it has been sampled with probabilities ranging in the interval $[10^{-4}, 0.8]$. For each sampling probability, ten distinct experiments have been repeated using different seeds for the random number generator. We have considered as flow key the first 8 bits of the sending IP address of each packet. Furthermore, 4 distinct SNR_{th} values have been considered, i.e., 10, 15, 20, and 50. In our experiments, we monitored using LEMON only flows for which a bin size smaller than 15 min was allowed. In fact, remaining ones are very small and hence can be handled using classic measurement techniques.

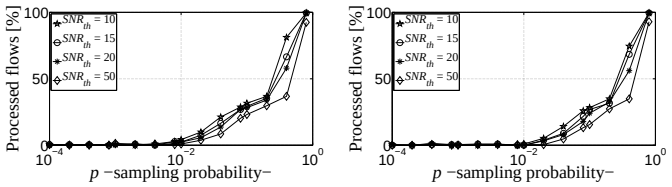
Figs. 3 and 4 show that the number of flows considered by LEMON increases with the sampling probability. This is because as p increases, the SNR increases too, so that, the same threshold value can be reached using a smaller bin size. As said before, a flow can be monitored by LEMON only if the bin size computed according to the algorithm described in Sec. III is smaller than 15 min. Thus, using a larger value of p can allow a larger number of flows to be processed.

In Figs. 5 and 6, we evaluate LEMON accuracy – when it exploits models in Eqs. (2) and (1), respectively – displaying

[2]The traces are available at http://mawi.wide.ad.jp/mawi/samplepoint-F/2009/ and http://tracer.cls.sony.co.jp/mawi/samplepoint-B/2006/.

TABLE I
MAIN TRAFFIC PARAMETERS OF AGGREGATE TRACES

	Link Capacity [Mbps]	Link Usage [%]	p_{bs}	$\overline{D}$ [Byte]	M [Byte^2]	Number of flows
Trace 1 (MAWI) Jan. 2009	150	13	0.015	341	400452	212
Trace 2 (MAWI) Dec. 2005	150	34	0.022	621	829127	150



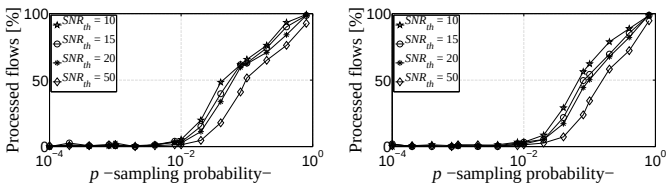
(a) Constant Packet Size Model. (b) Variable Packet Size Model.

Fig. 3. Number of processed flows, trace 1.

for all processed flows measured SNRs values. It is important to note that, regardless the flow size, the proposed approach is able to guarantee SNR values larger than the required threshold (SNR_{th}) using either the constant packet size model in Eq. (2) or the variable packet size one in Eq. (1). Another important consideration is related to the minimum flow size allowed by LEMON as a function of the SNR_{th}. In fact, it is easy to notice that increasing the SNR_{th} parameter yields the minimum size of processed flows to smaller and smaller values. This effect is due to the fact that as the flow size decreases, it becomes more and more difficult to find a suitable value of the bin size granting an SNR larger than the threshold. This suggests that using LEMON one operator should use small values for SNR_{th} if it is interested in small flows.

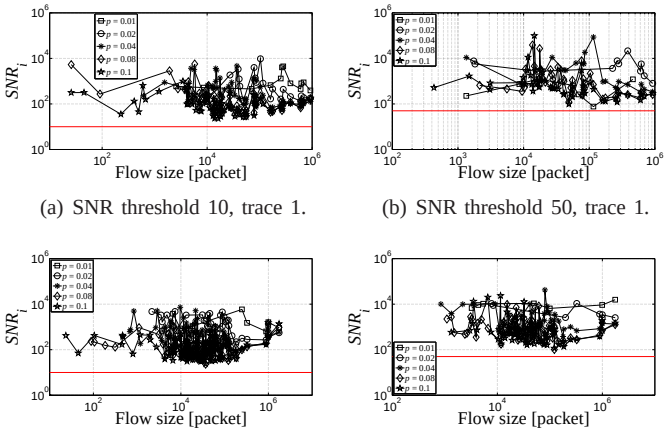
It is also interesting analyzing the bin size assigned to each flow as a function of the flow size (see Figs. 7 and 8). Obviously, LEMON assigns smaller values to the bin size for larger flows. In fact, only in the presence of a reasonable number of packets composing a flow one can analyze it with a fine time resolution.

Finally, Figs. 9 and 10 show the greatly accuracy level achievable in bitrate estimation for a couple of selected flows



(a) Constant Packet Size Model. (b) Variable Packet Size Model.

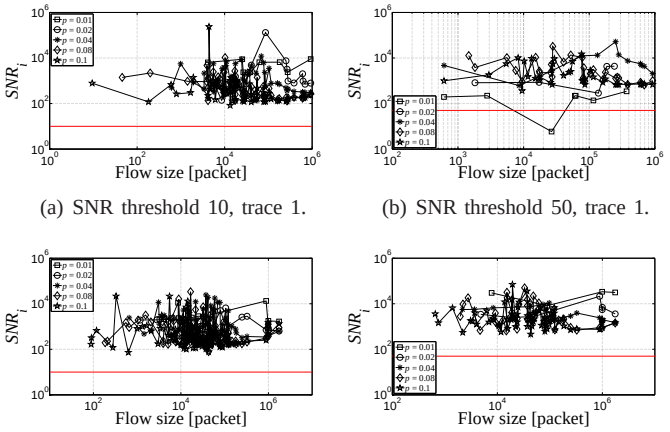
Fig. 4. Number of processed flows, trace 2.



(a) SNR threshold 10, trace 1. (b) SNR threshold 50, trace 1.

(c) SNR threshold 10, trace 2. (d) SNR threshold 50, trace 2.

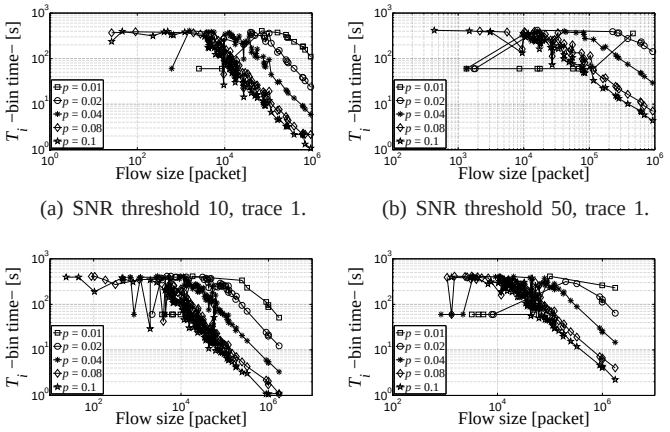
Fig. 5. Mean SNR, Constant Packet Size Model. The straight line represents the SNR threshold.



(a) SNR threshold 10, trace 1. (b) SNR threshold 50, trace 1.

(c) SNR threshold 10, trace 2. (d) SNR threshold 50, trace 2.

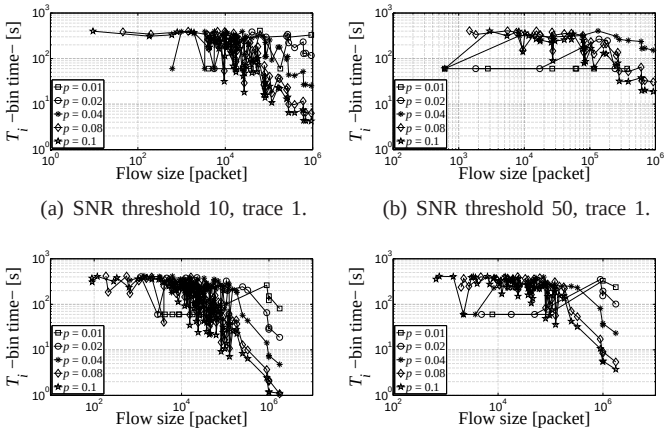
Fig. 6. Mean SNR, Variable Packet Size Model. The straight line represents the SNR threshold.



(a) SNR threshold 10, trace 1. (b) SNR threshold 50, trace 1.

(c) SNR threshold 10, trace 2. (d) SNR threshold 50, trace 2.

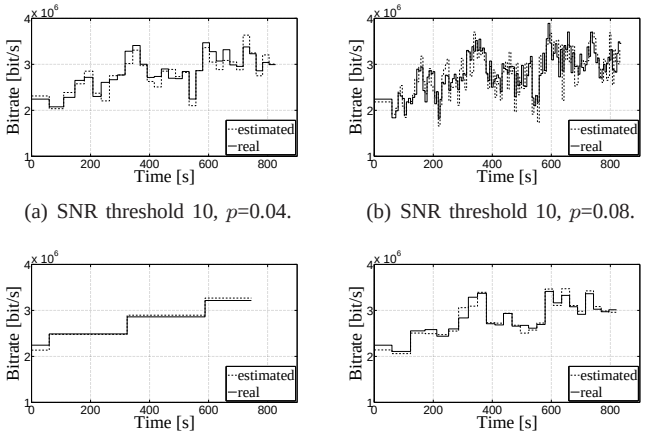
Fig. 7. Temporal resolution (mean T), Constant Packet Size Model.



(a) SNR threshold 10, trace 1. (b) SNR threshold 50, trace 1.

(c) SNR threshold 10, trace 2. (d) SNR threshold 50, trace 2.

Fig. 8. Temporal resolution (mean T), Variable Packet Size Model.

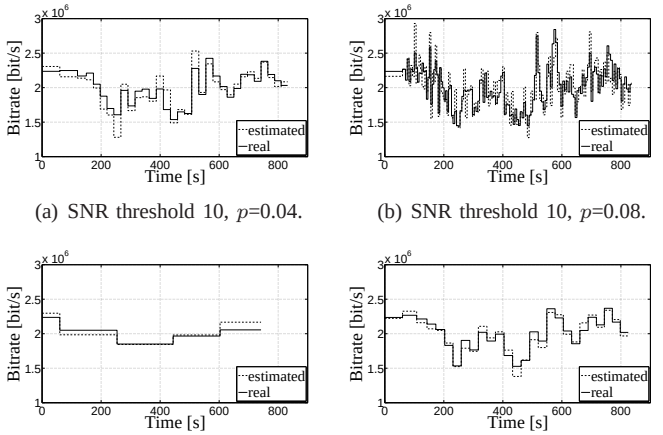


(a) SNR threshold 10, p=0.04. (b) SNR threshold 10, p=0.08.

(c) SNR threshold 50, p=0.04. (d) SNR threshold 50, p=0.08.

Fig. 10. Bitrate of flow #14, trace 1.

from traces 1 and 2, respectively.



(a) SNR threshold 10, p=0.04. (b) SNR threshold 10, p=0.08.

(c) SNR threshold 50, p=0.04. (d) SNR threshold 50, p=0.08.

Fig. 9. Bitrate of flow #4, trace 1.

To conclude, results clearly show the effectiveness of LEMON: (i) in all cases the measured SNRs are larger than the target threshold SNR_{th}; (ii) the larger SNR_{th} is, the larger becomes the smallest flow size processed by LEMON; (iii) as flow size increases, LEMON allows finer and finer time resolution accuracies.

V. Conclusions

In this work, the novel Lightweight Enhanced MOnitoring for backbone Networks, LEMON, algorithm has been conceived to improve the accuracy of IPFIX based monitoring systems. It is based on recently proposed frequency based models catching the impact of packet sampling and binning operations on bitrate estimation. The effectiveness of LEMON has been demonstrated using real packet traces. Future research will consider a wider set of traces, the evaluation of the processing and communication overheads, the integration in anomaly detection frameworks, and the comparison with analogous techniques already proposed in literature.

References

[1] F. Silveira, C. Diot, N. Taft, and R. Govindan, "Astute: detecting a different class of traffic anomalies," in *Proc. of ACM SIGCOMM 2010*.

[2] D. Brauckoff, B. Tellenbach, A. Wagner, M. May, and A. Lakhina, "Impact of packet sampling on anomaly detection metrics," in *Proc. of ACM SIGCOMM IMC 2006*.

[3] P. Kanuparthy, C. Dovrolis, and M. Ammar, "Spectral probing, crosstalk and frequency multiplexing in internet paths," in *Proc. of ACM SIGCOMM IMC 2008*.

[4] S. Katti, D. Katabi, C. Blake, E. Kohler, and J. Strauss, "Multiq: Automated detection of multiple bottlenecks along a path," in *Proc. of ACM IMC 2004*.

[5] K. C. Claffy, G. C. Polyzos., and K. W. Braun, "Application of sampling methodologies to network traffic characterization," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 23, no. 4, 1993.

[6] N. Duffield, "A framework for packet selection and reporting," in *IETF Draft (work in progress)*, Jun. 2008.

[7] C. Estan, K. Keys, D. Moore, and G. Varghese, "Building a better netflow," in *Proc. of ACM SIGCOMM 2004*.

[8] N. Hohn and D. Veitch, "Inverting sampled traffic," *IEEE/ACM Trans. on Networking*, vol. 14(1), pp. 68–80, 2006.

[9] J. Quittek, T. Zseby, B. Claise, and S. Zander, *Requirements for IP Flow Information Export (IPFIX)*, IETF, RFC 3917, Oct. 2004.

[10] B. Claise, *Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information*, IETF, RFC 5101, Jan. 2008.

[11] G. R. Cantieni, G. Iannaccone, C. Barakat, C. Diot, and P. Thiran, "Reformulating the monitor placement problem: Optimal network-wide sampling," in *In Proc. of ACM CoNeXT 2006*.

[12] L. A. Grieco and C. Barakat, "A frequency domain model to predict the estimation accuracy of packet sampling," in *Proceedings of IEEE INFOCOM*. IEEE, 2010, pp. 191–195.

[13] "Mawi working group traffic archive," http://tracer.csl.sony.co.jp/mawi/.

[14] G. Sadasivan, N. Brownlee, B. Claise, and J. Quittek, *Architecture for IP Flow Information Export*, IETF, RFC 5470, Mar. 2009.

[15] J. Quittek, S. Bryant, B. Claise, P. Aitken, and J. Meyer, *Information Model for IP Flow Information Export*, IETF, RFC 5102, Jan. 2008.

[16] B. Claise, A. Johnson, and J. Quittek, *Packet Sampling (PSAMP) Protocol Specifications*, IETF, RFC 5476, Mar. 2009.

[17] T. Dietz, B. Claise, P. Aitken, F. Dressler, and G. Carle, *Information Model for Packet Sampling Exports*, IETF, RFC 5477, Mar. 2009.

[18] L. A. Grieco and C. Barakat, "An analysis of packet sampling in the frequency domain," in *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, ser. IMC '09. ACM, 2009, pp. 170–176.