Impact of Packet Sampling on Link Dimensioning

Ricardo de O. Schmidt, Ramin Sadre, Anna Sperotto, Hans van den Berg, and Aiko Pras

***Abstract*—Link dimensioning is used by network operators to properly provision the capacity of their network links. Proposed methods for link dimensioning often require statistics, such as traffic variance, that need to be calculated from packet-level measurements. In practice, due to increasing traffic volume, operators deploy packet sampling techniques aiming to reduce the burden of traffic monitoring, but little is known about how link dimensioning is affected by such measurements. In this paper we make use of a previously proposed and validated dimensioning formula that requires traffic variance to estimate required link capacity. We assess the impact of three packet sampling techniques on link dimensioning, namely, Bernoulli, n-in-N and sFlow sampling. To account for the additional variance introduced by the sampling algorithms, we propose approaches to better estimate traffic variance from sampled data according to the employed technique. Results show that, depending on sampling rate and link load, packet sampling does not negatively impact on link dimensioning accuracy even at very short timescales such as 10ms. Moreover, we also show that the loss of inter-arrival time of sampled packets due to the exporting process in sFlow does not harm the estimations, given that an appropriate sampling rate is used. Our study is validated using a large dataset consisting of traffic packet traces captured at several locations around the globe.**

I. Introduction

Link dimensioning is often used by network operators aiming at optimal bandwidth provisioning or for network planning. Commonly, network operators use traffic averages obtained by polling SNMP MIBs (*e.g.,* interface octect counters) every couple of minutes [1], with graphical support provided by tools such as MRTG (*Multi Router Traffic Grapher*). These approaches, often referred to as *rules of thumb*, simply add to the traffic average a fixed amount of bandwidth as a safety margin that may depend on various factors such as time of the day. However, since the traffic averages are updated only every few minutes (*e.g.,* 5 minutes), traffic fluctuations that happen at much shorter timescales are completely overlooked. This can have a significant impact on applications that are sensitive to bandwidth fluctuations at small time scales, such as real-time video streaming.

Several alternatives to the rule of thumb have been proposed in the past. Most of them require traffic statistics that can only be obtained from traffic measurements done at the packet level, *i.e.,* through packet capturing. For instance, in [2], [3] the authors propose and validate a dimensioning formula that takes as input the average traffic rate and the traffic variance. However, continuous packet capturing does not scale well with the massive volume of traffic found in today's high-speed links, requiring dedicated and expensive measurement equipment. Therefore, although accurate down to very short timescales, link dimensioning approaches that demand continuous packet capturing are not deployed by network operators due to operational and economical considerations.

However, to cope with the increasing volume of traffic, many network operators deploy traffic monitoring based on *packet sampling*. For example, CERN (*European Organization for Nuclear Research*) [4] and AMS-IX (*Amsterdam Internet Exchange*) [5] use sFlow [6] to monitor and measure their huge volumes of network traffic. The goal of packet sampling is to reduce the excessive load of measurement, storage and processing, while still having highly granular traffic data. This raises the question whether the advantages of packet sampling can be exploited for link dimensioning. To the best of our knowledge, there is no research work that extensively investigates the impact of packet sampling on link dimensioning.

A. Contribution

In this paper we investigate the impact of packet sampling algorithms on the procedure of link dimensioning. Notice that this paper not only extends the initial work in [7], but provides a much more comprehensive study on the addressed topic. Our contributions in this paper are:

1) We demonstrate the feasibility of using sampled data for link dimensioning purposes. In particular, we focus on Bernoulli, 1-in-N [8], and sFlow sampling [6]. To do so, we adopt the link dimensioning approach proposed and validated in [2], [3]. This approach requires traffic statistics that are directly affected by the packet sampling, namely average traffic rate and traffic variance.
2) We propose approaches that provide more accurate estimates of traffic variance from sampled data. These approaches take into consideration the additional variance introduced by the sampling algorithms to better estimate traffic variance.
3) We investigate the impact of the loss of individual packet timestamps caused by the sFlow exporting process on link dimensioning, and we prove that even without knowing the arrival times of individual sampled packets we can obtain accurate estimations of required capacity. Our choice to study sFlow is motivated by the fact that this tool is widely adopted to monitor large networking infrastructures.

In our experiments, we use real network traffic traces captured at several different locations around the globe. These are packet-level captures that allow us to validate our results against an empirically defined ground truth. It is important

R. de O. Schmidt, A. Sperotto and A. Pras are with the Design and Analysis of Communication Systems Group, University of Twente, the Netherlands. E-mail: {r.schmidt,a.sperotto,a.pras}@utwente.nl

R. Sadre is with the Department for Computer Science, Université catholique de Louvain, Belgium. E-mail: ramin.sadre@uclouvain.be

H. van den Berg is with the TNO Information and Communication Technology and with the Design and Analysis of Communication Systems Group, University of Twente, the Netherlands. E-mail: j.l.vandenberg@tno.nl

to highlight that our goal is not to compare various sampling techniques pointing out which one is the best, but to show that one can make use of sampled traffic data to estimate the required capacity for a given link. Furthermore, we only consider sampling techniques that are already widely used in network devices. By using traffic measurement data already available to operators, our approaches are easier to deploy than those that require new sampling techniques, such as adaptive sampling techniques. In this way, we expand the applicability of link dimensioning.

B. Application

Our easy-to-use and accurate approaches expand the applicability of link dimensioning. For instance, we identify three scenarios that can profit from accurate estimations of required capacity. First, link dimensioning can be used for network capacity planning. For this purpose, the operators can initially measure the bandwidth utilization during very long periods (*e.g.,* a full day), aiming at identifying the busiest moments in the network. Then, accurate estimations of required capacity for those moments can be used for a proper provision of resources.

Second, link dimensioning can assist to the fair allocation of capacity resources for on-demand bandwidth reservation services. Nowadays, bandwidth on-demand works with static allocation of resources according to the capacity requested by the customer. However, one can think of services in which bandwidth allocation dynamically adapts to changing traffic demands.

Last, link dimensioning can be used to support adaptation/changes in the network topology performed by auto-configuration loops in self-managed networks. Auto-configuration loops reorganize the network typically targeting optimization of resources usage towards, *e.g.,* load balancing or energy efficiency. In [9], an energy-aware approach periodically reconfigures the traffic matrix of a core network (*e.g.,* every 15 minutes), aiming at reducing the amount of active devices and, hence, energy consumption. To do so, rough measurements of average traffic rates within core links are used. Accurate estimations of required capacity can support decisions on moving traffic streams from one link to another, taking into consideration their actual bandwidth requirements. This way, energy optimization goals can be reached without affecting Quality of Service.

C. Organization

The reminder of this paper is organized as follows. Section II positions this paper within the related work on both link dimensioning and the usage of packet sampling for network management. Background on packet sampling techniques is presented in Section III, as well as the motivation for the chosen techniques used in this work. In Section IV, we introduce the link dimensioning approach used in our experiments. In Section V we describe our proposed approaches to estimate traffic variance from sampled packets. In Section VI, we describe our traces dataset. Experiments methodology is described and results are discussed in Section VII. The impact of the exporting procedure of sFlow is assessed in Section VIII. Finally, in Section IX, we draw our conclusions.

II. Related Work

Link dimensioning has been extensively addressed in the past. On one hand, many approaches are technology-specific, *e.g.,* in [10] the authors proposed a procedure to allocate bandwidth for delay sensitive applications along a MPLS path. On the other hand, generic approaches have different goals, *e.g.,* [11] focuses on packet delay, [2], [3], [12] on link rate exceedance and [13] on packet loss. The main drawbacks of these solutions is that they require continuous packet capturing or modifications on router software. While the former might be a problem when considering high-speed networks, the latter might become difficult to implement in proprietary hardware. It is important to mention that in this paper we make use of the link dimensioning model proposed and validated in [2], [3], as later detailed in section IV.

Aiming at a scalable and easy-to-deploy solution for link dimensioning, in [14] we have proposed a flow-based (NetFlow/IPFIX) approach. The main weakness of the flow-based approach is, however, that it does not provide accurate results on estimation of required bandwidth at smaller timescales (*e.g.,* milliseconds). As a follow-up, we proposed in [15] another approach that combines flow-based measurements with mathematical models and that is able to accurately estimate required capacity at milliseconds timescales. The only downside of this second flow-based approach is that it needs occasional packet captures for parameters tuning. This extra measurement effort makes this approach more difficult to use. Other works have also used flows for link dimensioning. For example, the authors in [16] propose a traffic model, based on Poisson flow arrivals and i.i.d. flow rates, that is able to estimate required capacity for non-congested backbone links; and the authors in [17] provide dimensioning formulas for elastic data (TCP connections) on IP access networks where QoS is measured by useful per-flow throughput. Our work differs from these in that we make use of sampled packets and not flows for estimating the required capacity for a given link and also we do not put any constraint on the nature of the traffic.

Concerning packet sampling techniques, previous works have studied their usage on network operations and management. For example, in [18] the authors propose an adaptive sampling approach in which they can estimate traffic load and variance from sampled packets. Their approach is based on the assumption that sampling errors arise from the dynamics of packet sizes and counts, and that sampling with constant rate cannot guarantee accuracy on estimation of traffic statistics. In [19] the authors propose an adaptive sampling algorithm to estimate flow sizes and, more recently, in [20] the authors implement a sampling-based monitoring algorithm that focuses on accurate estimates of throughput for individual flows. Our work differs from the previous ones in that we do not propose a new sampling algorithm. Instead, focusing on practicality, we validate a complete link dimensioning procedure using simpler but widely adopted sampling algorithms. In addition, we validate our study using a large and heterogeneous measurements dataset consisting of real network traffic traces.

The use of packet sampling has also been studied in other areas. For example, in [21] and [22] the authors address the impact of sampling on anomaly detection operations. To the best of our knowledge, the impact of simple packet sampling techniques on link dimensioning problem has not yet been studied.

Note that in [7] we have briefly assessed the impact of the sampling algorithm used by sFlow on link dimensioning. This work significantly differs from the previous one by providing better estimations of the traffic variance and also by considering other sampling algorithms, as described in [8]. Furthermore, we study the effects on link dimensioning caused by the procedure implemented by sFlow to export sampled data. Finally, we consider link dimensioning at very small timescales, *e.g.,* 1 millisecond, which might be important when considering applications such as real time video streaming.

III. Packet Sampling

Sampling is an attractive solution for measuring high volume of traffic data. It helps to avoid the burden of measuring, storing and processing huge amount of packets. By using an appropriate packet selection strategy, sampled data can still provide a representative view of the whole traffic. There is a vast literature on packet sampling techniques and applications, and packet sampling is widely deployed by network operators and implemented in measurement tools, such as Cisco's *NetFlow* [23] and *sFlow* [6]. In this work we have chosen to study three sampling techniques, namely Bernoulli sampling (called uniform probabilistic sampling in [8]), n-in-N sampling (called n-out-of-N sampling in [8]), and sFlow sampling [6]. We have chosen these sampling techniques because they are largely employed in monitoring tools and have been standardized. The three techniques are briefly described in the following.

A. Bernoulli Sampling

In Bernoulli sampling [8], each packet is independently sampled with constant probability p. That is, for every received packet, the sampling function computes a real random number s between 0 and 1. The packet is sampled when the value s is less than the defined sampling probability p.

B. n-in-N Sampling

In n-in-N sampling [8], the stream of monitored packets is divided into non-overlapping windows of N packets, and from each window n packets are randomly sampled. Despite being more complex than Bernoulli sampling, n-in-N is widely used by operators and implemented in traffic monitoring and measurement technologies such as NetFlow because its design prevents more than $2n$ consecutive packets to be selected, this way avoiding measurement bursts. In this paper, we only consider the case of $n = 1$ since this is the most commonly used setting. For example, Random Sampled NetFlow only supports 1-in-N sampling [24].

C. sFlow Sampling

sFlow [6], [25] is a monitoring tool that uses packet filtering and sampling to provide scalable packet-based measurements in high-speed networks. The monitoring architecture of sFlow consists of agents embedded in switches and routers and a centralized collector. In the context of link dimensioning, we are interested in the whole traffic aggregate and, hence, we only focus on the sampling algorithm of sFlow and ignore packet filtering. Note that, though the name may be misleading, sFlow does not monitor and measure traffic at flow level, but at packet level.

For our experiments, we follow the implementation of the sampling algorithm provided by InMon [25] and used by other well-known sFlow software such as `pmacct` [26]. The decision of sampling a received packet is based on a randomly initialized counter such that in average one in N packets is sampled. The random counter s tells the algorithm how many packets to skip before sampling one. The counter s is progressively decremented every received packet, until it becomes zero and the current packet is sampled.

IV. Link Dimensioning

The link dimensioning approach used in this paper was proposed and validated in [2], [3]. Aiming at "link transparency", this approach provides dimensioning in which users almost never perceive network performance degradations due to lack of bandwidth. To statistically assure transparency to users, the provided link capacity C should satisfy

$$\mathbb{P}\{A(T) \geq CT\} \leq \varepsilon \quad , \qquad (1)$$

where $A(T)$ denotes the total amount of traffic arriving in intervals of length T, and ε indicates the probability that the traffic rate $A(T)/T$ is exceeding C at timescale T.

The link dimensioning formula from [2], [3] requires that traffic aggregates $A(T)$, at timescale T, are *Normal distributed* and *stationary*. The link capacity $C(T, \varepsilon)$, needed to satisfy Eq. (1), can be calculated by

$$C(T, \varepsilon) = \rho + \frac{1}{T}\sqrt{-2\log(\varepsilon) \cdot v(T)} \quad , \qquad (2)$$

where the mean traffic rate ρ is added with a term, that can be seen as a "safety margin", which depends on the variance $v(T)$ of $A(T)$.

By including the traffic variance, this bandwidth provisioning formula is able to take into account the impact of possible traffic bursts on the link capacity. In addition, it is very flexible: network operators can choose T and ε according to the QoS that they want to provide to their customers. For example, while larger T (*i.e.,* around 1s) would be enough to provide good quality of experience to users on web browsing, shorter T (*i.e.,* milliseconds scale) should be chosen when real time applications are predominant in the network, since the formula would be able to capture traffic bursts that happen at such short time scales. Notice that buffer size should also be taken into consideration when defining the size of T. The value for ε should be chosen in accordance to the desired QoS. Essentially, the lower ε the more importance is given to traffic bursts when computing the required link capacity. In short, while T will dictate to which extend traffic fluctuations are important, ε will account for how many intervals of size T the traffic aggregate is allowed to be higher than the required bandwidth $C(T, \varepsilon)$.

V. Estimating Traffic Variance

Eq. (2) requires that a good estimation of the mean traffic rate ρ and the variance $\upsilon(T)$ is available. Without sampling, rate and variance are easily obtained from traffic captures by, respectively,

$$\rho = \frac{1}{nT}\sum_{i=1}^{n} A_i(T) \quad \text{and} \quad \upsilon(T) = \frac{1}{n-1}\sum_{i=1}^{n}(A_i(T) - \rho T)^2 \quad ,$$

where $A_i(T)$ is the amount (in bytes) of observed traffic in time interval i of length T and n the number of monitored intervals.

When deploying packet sampling, only a fraction of the traffic will be available to compute the above statistics. Therefore, a key challenge in this paper is to accurately estimate ρ, $\upsilon(T)$, and, ultimately, $C(T, \varepsilon)$ from sampled traffic data.

Let r be the ratio between the total number of monitored packets and the number of sampled packets (*i.e.,* $r = 10$ for 1:10 sampling). Let ρ' be the mean traffic rate of the sampled traffic and let $A'_i(T)$ be the amount of sampled traffic (in bytes) observed in time interval i of length T. The original amount of traffic $A_i(T)$ in that interval can be estimated by

$$A_{i,est}(T) = r \cdot A'_i(T) \quad , \tag{3}$$

Hence, the original mean traffic can be estimated by

$$\rho_{est} = \frac{r}{nT}\sum_{i=1}^{n} A'_i(T) \quad . \tag{4}$$

Similarly, a (naive) estimation of the original variance can be obtained by

$$\upsilon_{est}(T) = \frac{r^2}{n-1}\sum_{i=1}^{n}(A'_i(T) - \rho' T)^2 \quad . \tag{5}$$

The drawback of this simplistic approach is that, while ρ_{est} is an unbiased estimator of the mean traffic rate ρ, the variance may be overestimated by $\upsilon_{est}(T)$ especially for small T and large r because the additional variance introduced by the sampling process is not taken into account. In the following we will develop better estimators of the traffic variance.

A. Variance Estimation with Bernoulli Sampling

Without sampling, the number of bytes A_i in interval i is

$$A_i = \sum_{j=1}^{P_i} S_{i,j} \tag{6}$$

where P_i is the number of packets in interval i and $S_{i,j}$ is the size of the jth packet in the interval. If we assume that P_i are i.i.d. like a random variable P, $S_{i,j}$ are i.i.d. like a random variable S, and P and S are independent, well-known results for random sums can be applied and it holds

$$\begin{aligned} \rho &= \frac{1}{T}E[P]E[S] & (7)\\ \upsilon(T) &= E[P]Var[S] + E[S]^2 Var[P] & (8) \end{aligned}$$

These two equations can be extended to the case of Bernoulli sampling: We "simulate" Bernoulli sampling with sampling probability p by randomly setting the size of some packets to 0. The size of a sampled packet becomes

$$S'_{i,j} = \begin{cases} 0, & \text{with probability } 1-p \\ S_{i,j}, & \text{with probability } p \quad . \end{cases}$$

Replacing S by S' in Eq. (8) (the complete derivation can be found in the appendix) yields the variance estimation

$$\upsilon_{bern}(T) = \upsilon_{est}(T) - (r-1)E[P]E[S^2] \quad , \tag{9}$$

where $r = 1/p$, $\upsilon_{est}(T)$ is the naive estimation from Eq. (5), $E[P]$ is the average number of packets per time interval before sampling, and $E[S^2]$ is the second moment of the packet size before sampling. $E[P]$ can be estimated by multiplying the measured average number of sampled packets per time interval by r. $E[S^2]$ can be estimated directly from the sizes of the sampled packets since the sampled packets have the same size distribution as the non-sampled packets according to our assumptions. Alternatively, traffic models could be used to obtain $E[S^2]$. For example, assuming packet sizes uniformly distributed between 40 and 1500 bytes would give us

$$E[S^2] = \frac{1}{12}(1500-40)^2 + \frac{1}{4}(1500-40)^2 \quad .$$

In the experiments in this paper, we will estimate $E[P]$ and $E[S^2]$ directly from the measurements, as described above; packet size models will not be discussed further.

B. Variance Estimation with 1-in-N and sFlow Sampling

A mathematical treatment of 1-in-N sampling is much more complex than for Bernoulli sampling. The sampling window of N packets can stretch over several intervals T, making the sampled traffic $A'_i(T)$ and $A'_{i+1}(T)$ in adjacent time intervals dependent. Similar difficulties arise for sFlow sampling. Hence, we simplify the problem and assume that $1/N$ of the packets of each interval are sampled. Furthermore, we assume again that the numbers of packets per interval in the original traffic stream are i.i.d. like P and packet sizes are i.i.d. like S, and P and S are independent. Under these assumption, the number of sampled packets P'_i in time interval i is $P'_i = P_i/r$ with $r = N$ (we ignore the problem that P_i/r might not be a natural number). Replacing P by P/r in Eq. (8), we obtain the variance estimation

$$\upsilon_N(T) = \upsilon_{est}(T) - (r-1)E[P]Var[S] \quad , \tag{10}$$

where $E[P]$ is the average number of packets per time interval before sampling, and $Var[S]$ is the variance of the packet size before sampling. Again, $E[P]$ can be estimated from the number of sampled packets and $E[S^2]$, and consequently $Var[S]$, can be estimated directly from the sizes of the sampled packets or, alternatively, from traffic models. In the

TABLE I
SUMMARY OF MEASUREMENTS

abbr.	description	year	# traces	length	# of hosts	link capacity	avg. load
A	link from university's building to core router	2011	96	24h	6.5k	2 × 1 Gb/s	15%
B	core router of university in the Netherlands	2012	24	6h	886k	10 Gb/s	10%
C	core router of university in Brazil	2012	75	18h45m	10.5k	155 and 40 Mb/s	19%
D	backbone links connecting Chicago and Seattle	2011	16	4h	1.8M	2 × 10 Gb/s	8%
E	backbone links connecting San Jose and Los Angeles	2011–2012	20	5h	3M	2 × 10 Gb/s	10%
F	trans-Pacific backbone link	2012	53	13h15m	4M	n/a	n/a

experiments in this paper, we will estimate $E[P]$ and $Var[S]$ from the measurements. Note that our approximation can be extended to n-in-N sampling by choosing $r = N/n$ but, as explained in Section III-B, we focus on the more common 1-in-N sampling in this paper.

VI. Dataset

In this section we describe the measurement dataset used in our experiments. The entire dataset is composed by traffic captures done at the IP packet level using tools such as `tcpdump`. This allows us to validate the experimental results against a ground truth (*i.e.,* values empirically determined). The entire dataset comprises 284 15-minute traces, totaling 71 hours of captures. Note that the trace duration of 15 minutes has been chosen in accordance with [2], [3]. Longer time periods are generally not stationary due to diurnal patterns. These traces come from different locations around the globe and account for a total of more than 11 billion packets. While a brief description on each measurement location is provided below, Table I summarizes the dataset (note that in this table, the column "length" gives the total duration of the, not necessarily successive, 15-minute traces, *i.e.,* a length of 1h corresponds to four traces).

A. Measurement Locations

In location A, an aggregate link 2 × 1 Gb/s was measured. This link connects a building to the gateway of a university in the Netherlands. Most traffic in this link is actually internal, *i.e.,* it goes from the building to other buildings on the campus. Due to the small number of active hosts in the link, single activities, such as an overnight automatic backup, can completely reshape the traffic for a period of time. This measurement took place in a week day of September 2011, and lasted for 24 successive hours.

Measurements of location B took place at the gateway of a university in the Netherlands. A 10 Gb/s link that comprises all the incoming and outgoing traffic of the university was measured. During 24 hours, the first 15 minutes of traffic every full hour was captured. The measurement took place in December 2012 and most traffic is web browsing and email. Note that B may contain part of the traffic from A.

Location C consists of measurements of an aggregate link (155 Mb/s and 40 Mb/s) at the gateway of a Brazilian university. This measurement are from September to December 2012. As for B, traces are captures of the first 15 minutes of every full hour and most traffic is web browsing and email.

Traces from locations D and E are from CAIDA's public repository[1]. These are 1-hour long traces measured on different days throughout the years. For D, traces are from May to June 2011, and for E, traces are from December 2011 to February 2012. Two 10 Gb/s links, interconnecting two US cities, were measured at each location.

Finally, measurements from F come from MAWI's public repository. F consists of traffic captures in a trans-pacific link. No additional information on link capacity and use (load) is provided by MAWI[2]. Measurements are from Nov.–Dec. 2012.

For all measurements directly performed by us (*i.e.,* locations A, B and C), no packet losses were observed. We know from the CAIDA website that packet losses are likely to happen for location D but they do not keep record of such losses. For traces from location F, no information on packet loss is provided by MAWI.

B. Traffic Characteristics

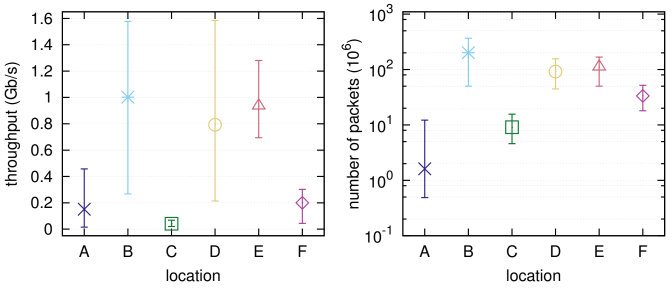
The last column of Table I presents the average link load for each location. Naturally, the load is not expected to be constant over the measurement period. Fig. 1a shows the average, minimum and maximum traffic rate per 15-minute for each location. Locations with higher-capacity links are the ones in which traffic varies most. In case of 24-hour measurements from A and B, differences between minimum and maximum rates are due to diurnal traffic fluctuations. Fig. 1b shows the average, minimum and maximum number of packets per 15-minute trace for each location.

VII. Experiments

In this section we first describe the methodology used in our experiments in Section VII-A. Next, in Section VII-B we show the impact of packet sampling on the Gaussian character of traffic. In Section VII-C we compare the results obtained for link dimensioning from each of the considered sampling strategies. In Section VII-D we assess the impact of sampling on link dimensioning using all traces from our dataset and present the overall results of the validation. Finally, in Section VII-E, we put special focus on quantifying any overestimation.

[1] The CAIDA UCSD Anonymized Internet Traces 2011 and 2012. Available at http://www.caida.org/

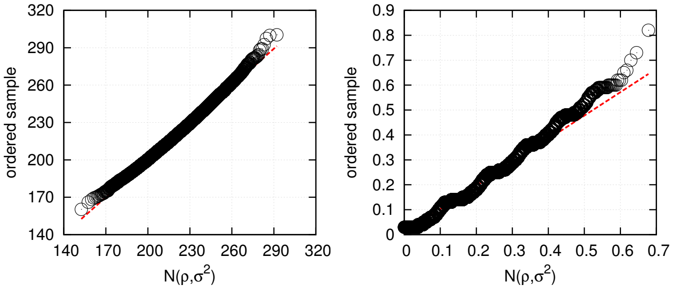
[2] Information on the link capacity provided on http://mawi.wide.ad.jp/ is not consistent with the throughput observed in the traces.



(a) Average traffic rate

(b) Average number of packets

Fig. 1. Traffic characteristics. Error bars indicate minimum and maximum values.



(a) Not sampled

(b) Sampled 1:1000

Fig. 2. Q-Q plots at $T = 100$ms for an example trace.

A. Methodology

We study all sampling techniques at different timescales T and with sampling rates of 1:10, 1:100, and 1:1000. However, 1:1000 sampling is not applicable to most traces in our dataset since the traffic at some locations comprises only a few thousands of packets per time interval (see Section VI). The problems resulting from this are discussed in the next sections.

As already explained, the link dimensioning formula in Eq. (2) requires that the traffic aggregates at timescale T are Normal distributed. To assess the Gaussian character of the traffic traces we use a procedure in accordance to [27], [28], [29]: For each trace in our dataset we compute the *linear correlation coefficient* that indicates whether the distribution of the traffic aggregates is Normal. The procedure and analysis are detailed in Section VII-B.

To validate the accuracy of the link dimensioning procedure using sampled data, we calculate the estimated required bandwidth $C(T,\varepsilon)$ from Eq. (2), using our variance estimations from Eq. (9) and (10) for the different sampling techniques. This estimated bandwidth is then compared with the *empirically estimated bandwidth* $C_{emp}(T,\varepsilon)$. The empirically estimated bandwidth is the $(1-\varepsilon)$-quantile of the empirical CDF distribution of the throughput directly calculated from the full packet trace. This value represents the minimum capacity that should be allocated so that in only the predefined fraction ε of time intervals of size T the throughput will be above the required capacity:

$$C_{emp}(T,\varepsilon) := \min\left\{C : \#\{A_i(T) \mid A_i(T) > CT\}/n \leq \varepsilon\right\}, \tag{11}$$

where $A_1(T),\dots,A_n(T)$ are the empirical traffic aggregates on timescale T and ε is the bandwidth exceedance probability. Additionally, to verify the accuracy of the estimated bandwidth $C(T,\varepsilon)$, we calculate the fraction of measured intervals in which the traffic aggregate $A_i(T)$ exceeds $C(T,\varepsilon)$:

$$\hat{\varepsilon} := \#\{A_i(T) \mid A_i(T) > C(T,\varepsilon)T\}/n \quad . \tag{12}$$

Note that $\hat{\varepsilon} \leq \varepsilon$ is, by definition, equivalent to $C(T,\varepsilon) \geq C_{emp}(T,\varepsilon)$. In order to comply with previous work [2], [3], we set ε to 0.01 and 0.05, with T ranging from 1ms to 1s in the following experiments.

Finally, we quantify a possible underestimation or overestimation of the required capacity by calculating the relative error, in percentage, between $C(T,\varepsilon)$ and $C_{emp}(T,\varepsilon)$:

$$RE = \frac{C(T,\varepsilon) - C_{emp}(T,\varepsilon)}{C_{emp}(T,\varepsilon)} \cdot 100\% \quad . \tag{13}$$

B. Traffic Gaussianity

Gaussianity of traffic is a requirement for Eq. 2 and, hence, its assessment is part of our validation procedure. Being T the timescale of traffic aggregation, and $A_1(T),\dots,A_n(T)$ the amount of traffic observed in time periods $1,2,\dots,n$ of length T, we want to know if $A(T)$ is Gaussian distributed, i.e., whether $A(T) \sim \text{Norm}(\rho, \upsilon(T))$.

There are several ways to assess the Gaussianity of a traffic trace. Quantile-quantile (Q-Q) plots can be used to qualitatively check Gaussianity *goodness of fit*. Q-Q plots are created by plotting the pairs

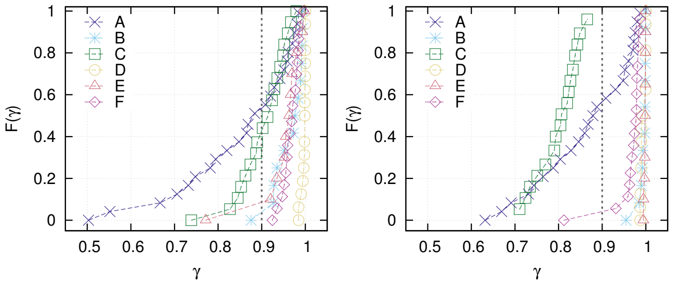
$$\left(\Phi^{-1}\left(\frac{i}{n+1}\right), \alpha_{(i)}\right), \quad i = 1,2,\dots,n \quad , \tag{14}$$

where Φ^{-1} is the inverse of the normal cumulative distribution function and $\alpha_{(i)}$ are the ordered traffic averages for each time interval of length T. Fig. 2 shows Q-Q plots for an example trace without and with sampling. In a Q-Q plot, the traffic is considered "perfectly Gaussian" when all points are perfectly aligned in a diagonal line. One can see that for the example trace in Fig. 2a, traffic is likely to have a good Gaussian fit, since only few points fall out of the diagonal line. However, for the same trace sampled 1:1000 with 1-in-N sampling (Fig. 2b) we can observe many deviations from the diagonal line, which may lead to inaccurate estimations of the required capacity. It is also important to know that the impact of sampling on traffic Gaussian fit is aggravated at shorter timescales.

Q-Q plots provide a good visualization of the Gaussianity of a single traffic trace. However, we also need a way to quantitatively assess such property for a larger set of traces. In accordance to previous works [27], [28], [29], we use the *linear correlation coefficient* [30] defined as

$$\gamma(x,y) = \frac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\overline{x})^2\sum_{i=1}^{n}(y_i-\overline{y})^2}} \quad , \tag{15}$$

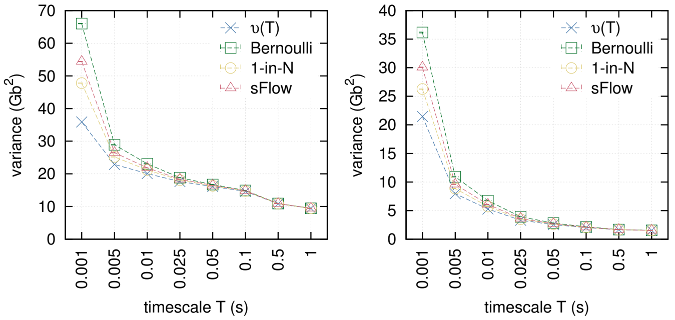
where the pair (x,y) is defined as in Eq. (14). For a given traffic trace, $|\gamma| = 1$ if, and only if, the traffic is perfectly Gaussian. Note that, as proved in [28], $\gamma \geq 0.9$ corresponds to a Kolmogorov-Smirnov test for normality at significance 0.05,



(a) Not sampled

(b) Sampled 1:1000

Fig. 3. CDF of γ for all traces per location at $T = 100$ms.



(a) Trace from A

(b) Trace from F

Fig. 6. Difference between the average traffic variance calculated from 10 runs of sampling for each algorithm. Example using randomly chosen traces and sampling 1:10.

which supports the hypothesis that the underlying distribution is Normal.

The main take away of this section is that Gaussianity fit is not affected when using a proper sampling rate. The selection of a proper sampling rate takes into consideration the volume of the traffic in the monitored link. For example, at $T = 100$ms, 70% of all traces in our dataset have good Gaussian fit, with an overall average γ of 0.9135, and when these are sampled 1:10 using 1-in-N algorithm, the average γ becomes 0.9164 and around 71% of all traces have $\gamma \geq 0.9$. If an inappropriate sampling rate is chosen, however, important traffic characteristics might be lost and, ultimately, Gaussianity fit might be misleading. The choice of an improper sampling rate often happens by considering solely the capacity of the monitored link and not the volume of the traffic, as proposed in [31]. For example, Fig. 3 shows the CDF of γ for all traces in our dataset when not sampled and when sampled 1:1000 using 1-in-N sampling. This sampling rate has a major impact on the Gaussianity of traces with smaller traffic aggregates, in our case traces from locations A and C. The plots in this figure also show that the opposite might happen, where Gaussianity fit of few traces actually increased after sampling. This means that for the traces in our dataset the sampling rate 1:1000 is too low and might culminate in misleading results about traffic Gaussianity fit.

The impact of Gaussianity fit on link dimensioning is shown throughout the analysis of results in the next sections. In addition, a thorough study of Gaussianity fit of the datasets used in this paper can be found in [29].

C. Comparing Sampling Strategies

Fig. 4 compares the traffic time series computed from the complete traces and from sampled data. The latter was computed using the scaling from Eq. (4). From analyzing the time series, one can see that the problem of overestimation is likely to happen at shorter T. For all sampling approaches at $T = 1$ms we can see several traffic peaks in the time series of sampled data that do not actually exist, as observed in the time series created with the complete traces.

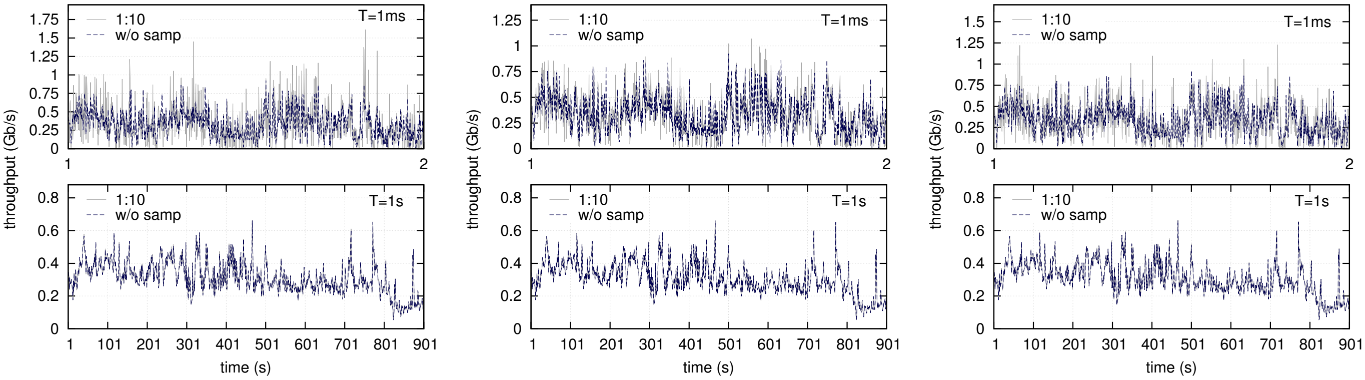
The estimation of traffic within individual time intervals is less accurate with Bernoulli sampling. That is because with Bernoulli sampling, several consecutive packets, or very close in time, might be sampled. Such packets might end up in the same time interval and, ultimately, the scaling of traffic within the interval in Eq. (4) (*i.e.,* by multiplying the sampled traffic by r) creates non-existing traffic bursts.

This problem is alleviated with 1-in-N sampling due to the definition of sampling windows of N packets. Although with lower probability, overestimation of traffic average can also happen for 1-in-N sampling once nearby packets are sampled, for example, if the last packet of a current window and the first packet of the next window happen to be sampled. Once again, these might end up in the same time interval, which later might create non-existing traffic bursts.

Using sFlow sampling, results are in between Bernoulli and 1-in-N ones. Although also defining a kind of sampling window, sFlow does not discard the remaining packets in the current window, as done by the 1-in-N algorithm. If the sampled packet is not the last in the window, all the remaining packets are considered as part of the packets range for the next sampling. This way, the behavior of sFlow is somewhere in between Bernoulli and 1-in-N sampling.

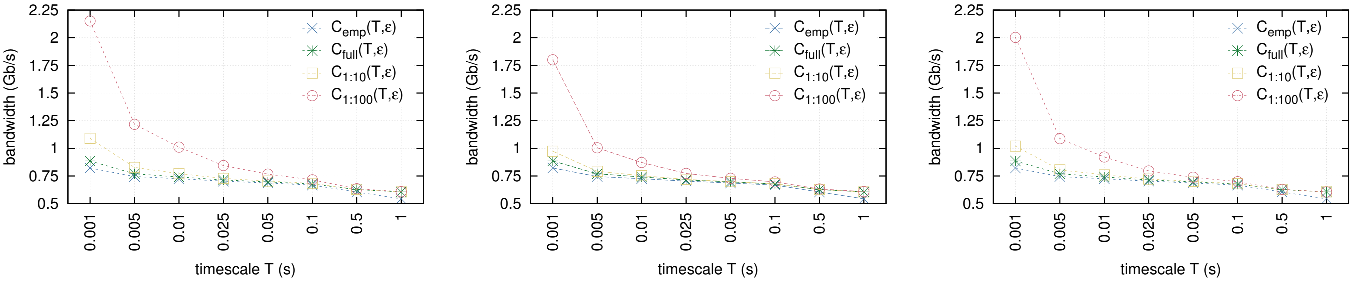
The problem of non-existing traffic bursts disappears at large timescales. As one can see in Fig. 4, at $T = 1$s, time series created from sampled data (independently from the concrete sampling technique) faithfully reproduce traffic fluctuations as seen for the complete traces. These results become clearer when analyzing the estimations of required capacity $C(T, \varepsilon)$ at various T and sampling rates, shown in Fig. 5. Note that the example trace used in Fig. 4 and 5 is the same. From Fig. 5, it is clear that sampling rates of 1:100 are too low and, hence, they result in excessive overestimation of the required capacity at shorter timescales. Nonetheless, the difference of final results of $C(T, \varepsilon)$ between the three sampling techniques is not significant.

This small difference between the results obtained for each sampling approach remains constant for other traces and for several runs of sampling. Fig. 6 shows the average traffic variance computed from 10 runs of sampling for each technique always using the same example trace. Note that the actual traffic variance $\upsilon(T)$ is also plotted for matters of comparison. Although very small and difficult to see, the standard deviation is plotted as error bars in this figure. For both example traces, at any T or sampling algorithm, the standard deviation is



(a) Bernoulli sampling (b) 1-in-N sampling (c) sFlow sampling

Fig. 4. Comparison between sampling algorithms for a trace from location A. For the sake of clarity, time series at $T = 1$ms show one second only.



(a) Bernoulli sampling (b) 1-in-N sampling (c) sFlow sampling

Fig. 5. Comparison between sampling algorithms for a trace from location A. For the sake of clarity, time series at $T = 1$ms show one second only. Note that $C_{\text{full}}(T, \varepsilon)$ refers to estimations using the complete trace (*i.e.,* not sampled).

smaller than 1% of the average variance. This confirms that, even with the random nature of the sampling algorithms, several runs still yield very close results. Concerning the traffic variance, the difference between the three sampling algorithms is larger at $T = 1$ms, although still not very significant. At any other T, this difference is negligible or inexistent. These differences are also in line with the estimations $C(T, \varepsilon)$ presented in the plots of Fig. 5.

Therefore, given the small difference between results from different sampling algorithms, for simplicity in the next section, the overall assessment of the impact of packet sampling on link dimensioning is done using only 1-in-N sampling. In addition, due to the very small difference between variances obtained from 10 runs of sampling, results presented in the next section are obtained from a single run of sampling per trace in our dataset.

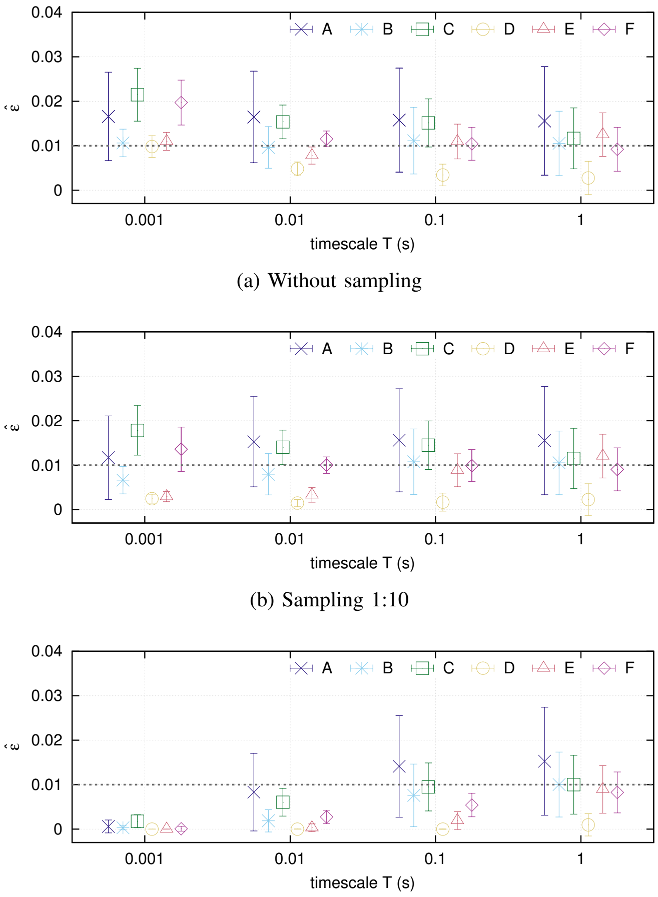
D. Overall Results

In this section we present results of an extensive validation of the use of sampled data for link dimensioning. To do so, we sampled all traces in our measurements dataset using 1-in-N.

Even using the adapted variance formula from Eq. (10), conservative $C(T, \varepsilon)$ due to overshooting of the traffic variance might be a problem at smaller timescales and lower sampling rates. Fig. 7 and 8 clearly show this problem. These figures show the average and standard deviation (error bars) of $\hat{\varepsilon}$ at various T and sampling rates for $\varepsilon = 0.01$ and $\varepsilon = 0.05$, respectively. It is important to recapitulate that $\hat{\varepsilon}$, as defined in Eq. (12), is the fraction of measured intervals of size T in which actual traffic rates are higher than the estimated $C(T, \varepsilon)$. By setting $\varepsilon = 1\%$ in the dimensioning formula of Eq. (2), a successful estimation of required capacity ultimately yields $C(T, \varepsilon) \geq C_{emp}(T, \varepsilon)$, which is equivalent to $\hat{\varepsilon} \leq \varepsilon$.

As shown in Fig. 7 and 8, independently of ε, very conservative results are obtained at small timescales and low sampling rates and they might confront with the intelligent overestimation premiss. For example, Fig. 7c and 8c show that at $T = 1$ms the average and the standard deviation of $\hat{\varepsilon}$ for all locations is near to or equals zero. That is, $C_{emp}(T, \varepsilon)$ was never underestimated. In such cases one should suspect that overestimation might have been too high (as further demonstrated in section VII-E). This can also be expected since, as showed in Fig. 6, at $T < 5$ms the approximation error for the traffic variance becomes larger.

Having in mind that excessive overestimation might happen, from the results in Fig. 7 and 8 one can still argue that estimations from sampled data are close to the ones obtained from complete traces (*i.e.,* without sampling). Therefore, from these results we can conclude that packet sampling can be used for link dimensioning purposes, provided that the sampling rate is wisely chosen, taking into consideration the volume of traffic in the link and the timescale of interest. This statement complies with the results shown in Fig. 5b where, for an example trace, at smaller T and lower sampling rate the estimation was way higher than the actual required capacity.

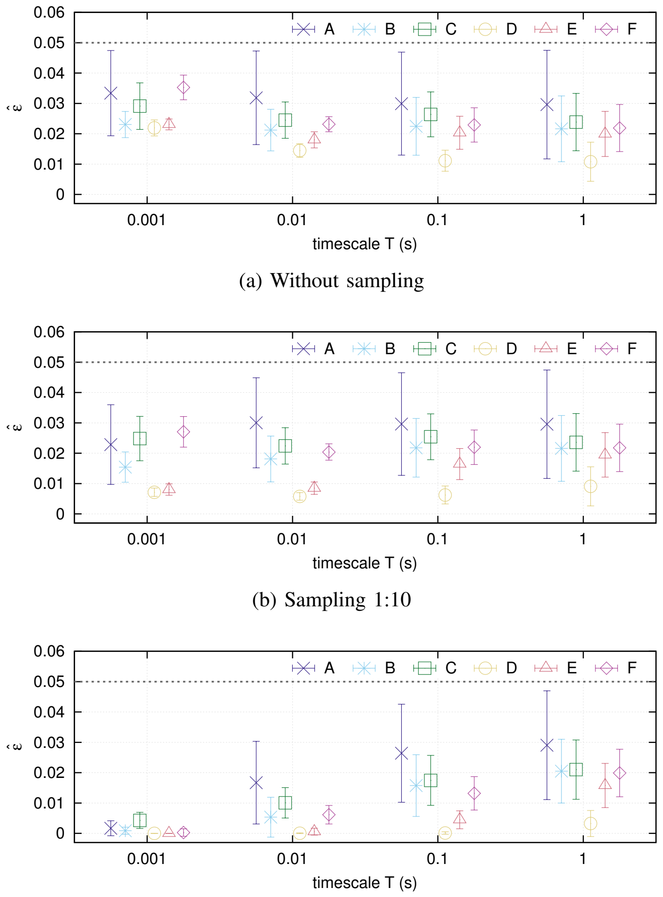


(a) Without sampling

(b) Sampling 1:10

(c) Sampling 1:100

Fig. 7. Average and standard deviation (error bars) of $\hat{\varepsilon}$ at various T for $\varepsilon = 0.01$.

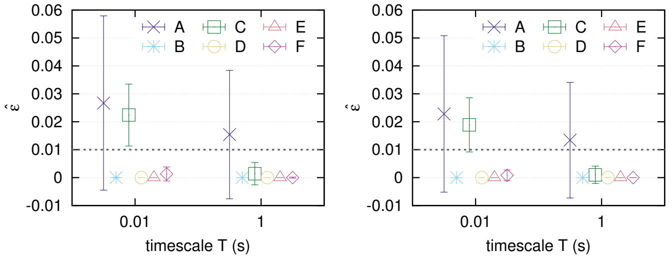


(a) Without sampling

(b) Sampling 1:10

(c) Sampling 1:100

Fig. 8. Average and standard deviation (error bars) of $\hat{\varepsilon}$ at various T for $\varepsilon = 0.05$.



(a) Sampled 1:10

(b) Sampled 1:100

Fig. 9. Average and standard deviation (error bars) of $\hat{\varepsilon}$ at various T for $\varepsilon = 0.01$ when using Eq. (5) to compute traffic variance from sampled data.

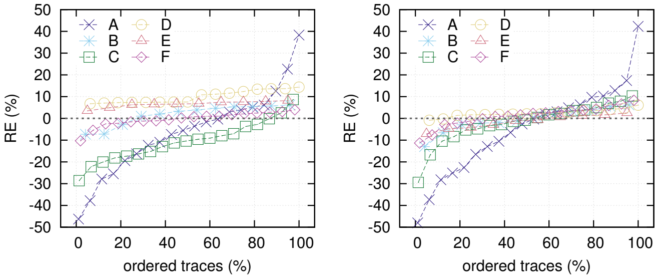
For matters of comparison, Fig. 9 shows the average and the standard deviation of $\hat{\varepsilon}$ for all traces when estimating the required capacity using the traffic variance obtained from Eq. (5). Given that the additional variance introduced by the sampling process is not taken into consideration, the naive estimation in Eq. (5) results in excessive overestimation of the required capacity, independently of sampling rate or timescale. Once again, overestimation does not necessarily mean success, but rather the obtained $C(T, \varepsilon)$ might be way higher than the actual needed $C_{emp}(T, \varepsilon)$ for the given traffic aggregate.

E. Quantifying the Overestimation

Aiming at an intelligent link capacity provisioning, it is important to also quantify how much the estimated required capacity $C(T, \varepsilon)$ overestimates the actual required capacity $C_{emp}(T, \varepsilon)$. The relative error RE is given by Eq. (13) and it quantifies the difference between the two mentioned estimations. Fig. 10 shows the obtained RE for all traces sampled 1:10. One can see that, at $T = 10$ms, around 78% of all traces had under or overestimation within reasonable bounds, *i.e.,* $RE = \pm 15\%$, and at $T = 1$s this value is around 85%. Approximately 65% and 74% of all traces are within a boundary of $RE = \pm 10\%$ at $T = 10$ms and $T = 1$s, respectively. The worst cases are few traces from A for which RE was up to $\pm 50\%$.

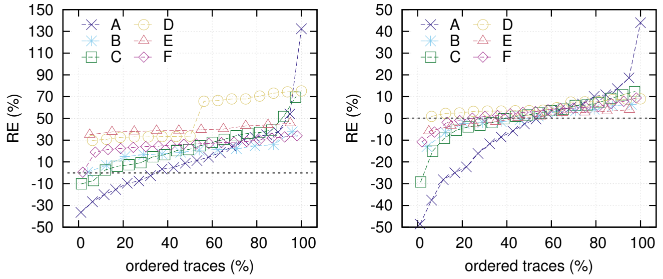
Fig. 11 shows the relative error RE for all traces sampled 1:100. While RE at $T = 1$s does not differ much from when traces are sampled 1:10, the RE obtained at $T = 10$ms demonstrates the problems arising from a combination of short T and low sampling rates. While for sampling 1:10 at $T = 10$ms most traces had $RE = \pm 15\%$, for traces sampled 1:100 this range increases to $-10\% \leq RE \leq 50\%$. The worst cases are some traces from location D and again few traces from location A for which RE was up to 130%.

One important take away is that the combination of short timescale with inappropriate sampling rate might result in



(a) $T = 10$ms (b) $T = 1$s

Fig. 10. RE for all traces per location. 1-in-N sampling 1:10.



(a) $T = 10$ms (b) $T = 1$s

Fig. 11. RE for all traces per location. 1-in-N sampling 1:100.

excessive overestimation. These results, specially those in Fig. 11a, support the statements regarding the problem of excessive overestimation made in the previous section and showed in Fig. 7 and 8.

In addition, it is important to understand that although still present, overestimation is expressively reduced when using the proposed approaches from Section V to estimate traffic variance. This is clearly visible when comparing Fig. 7 and 9. This is because the proposed approaches take into consideration the additional variance introduced by the random nature of sampling. Ultimately, the better estimation of the traffic variance reflects in a higher estimation of required capacity, yielding $\hat{\varepsilon}$ in average greater than 0 but less than 0.01 (the value of ε).

For matters of comparison, when using Eq. (5) to calculate traffic variance at $T = 10$ms from data sampled 1:10 or 1:100, for more than 50% of all traces in our dataset the estimated required capacity resulted in overestimation (*i.e.,* $RE > 15\%$). At $T = 1$s, excessive overestimation increases to approximately 85% of all traces. This clearly proves that our proposed approaches to estimate traffic variance from sampled data significantly help to alleviate the problem of overestimating required capacity.

To further mitigate overestimation, our assumptions in Section V could be sharpened to better represent actual network traffic. For example, accounting for dependent packet numbers and sizes could potentially result in even more accurate estimations of required capacity. Such research is, however, considered as future work.

VIII. Impact of Sampled Data Export

In the previous section we presented the results on how sampled data can impact the accuracy of the link dimensioning procedure. Considering a real networking scenario in which the traffic is monitored using the sFlow tool, the exporting process may also present some challenges to the link dimensioning problem. In this section we first describe how the sFlow tool works and then we assess the impact of the sFlow exporting operation on link dimensioning using the datasets of locations B and C.

A. sFlow Operation

The general operation of sFlow is shown in Fig. 12. sFlow operation can be roughly divided into two entities, namely the *agent* and the *collector*. The sFlow agent is a software located at the router, switch or standalone probe, and it is responsible for monitoring traffic and handling the sampled data. The agent can be remotely accessed and controlled via an sFlow MIB (SNMP Management Information Base). The sFlow agents export sampled data to a centralized collector using sFlow datagrams. At the collector the sampled data is analyzed and stored.

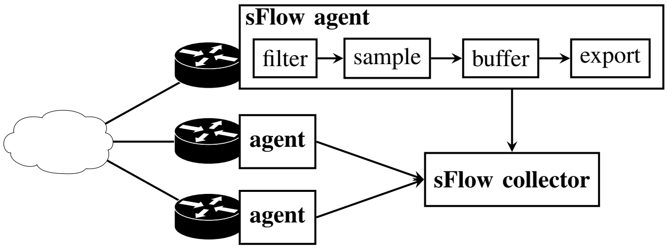


Fig. 12. sFlow exporting process.

Every observed packet by the sFlow agent undergoes two stages: filtering and sampling. The former allows the agent to drop packets that belong to flows that are not of interest. All packets that pass the filtering process are subject to the sampling algorithm. The sampling algorithm implemented by InMon [25], as well as by other sFlow tools, is the same as the one detailed in Section III-C. Before being exported to the collector, sampled packets are kept in a buffer within the agent. sFlow defines two conditions for exporting buffered packets: (1) there are enough packets to fill a sFlow datagram, or (2) the oldest sampled packet in the buffer should not wait for longer than 1 second to be exported.

There is one important characteristic of sFlow that can directly impact on the link dimensioning approach: the loss of individual timestamps of sampled packets. A datagram exported by the agent to the collector may contain information on several sampled packets. However, the sFlow datagram format only specifies a *single* datagram-wide export timestamp. The sampled information of packets does not include their respective timestamps and, therefore, all the packets that are grouped into a datagram share the same timestamp. The amount of missing information depends on the link load, sampling rate and buffer size (which is defined by the capture length). Clearly, not knowing indivual packet arrival times limits any estimation of the traffic variance from the sampled data. In the next section we assess the impact of this missing information on the accuracy of the link dimensioning procedure.

B. Experimental Results

Implementation assumptions. For the specific case of link dimensioning in this paper, we are interested in the whole traffic and, hence, all observed packets undergo sampling. That is, our packet filtering rule is defined as "accept all". Aiming at simplicity we have implemented a sFlow exporter focusing on evaluating the effects of loss of packets timestamps on link dimensioning. Therefore, we make a few assumptions and abstractions: (1) we assume the data analysis at the collector to solely consist of the link dimensioning procedure described and assessed in the previous sections; (2) sFlow agent and collector are located in the same machine and, therefore, sFlow datagrams do not need to experience network transmission delays; and (3) sFlow datagrams are simplified to only carry the headers of sampled packets. Note that, considering the characteristics we are interested in, none of above mentioned decisions and assumptions invalidates our experimental implementation of sFlow when compared to a commercial one. It is also important to mention that we do not consider datagram loss due to, for example, packet drops that may happen in congested networks or because the agent's jobs run with a low priority on the exporting router or switch. Furthermore, all metrics used in the following analysis were already introduced in Section VII-A.

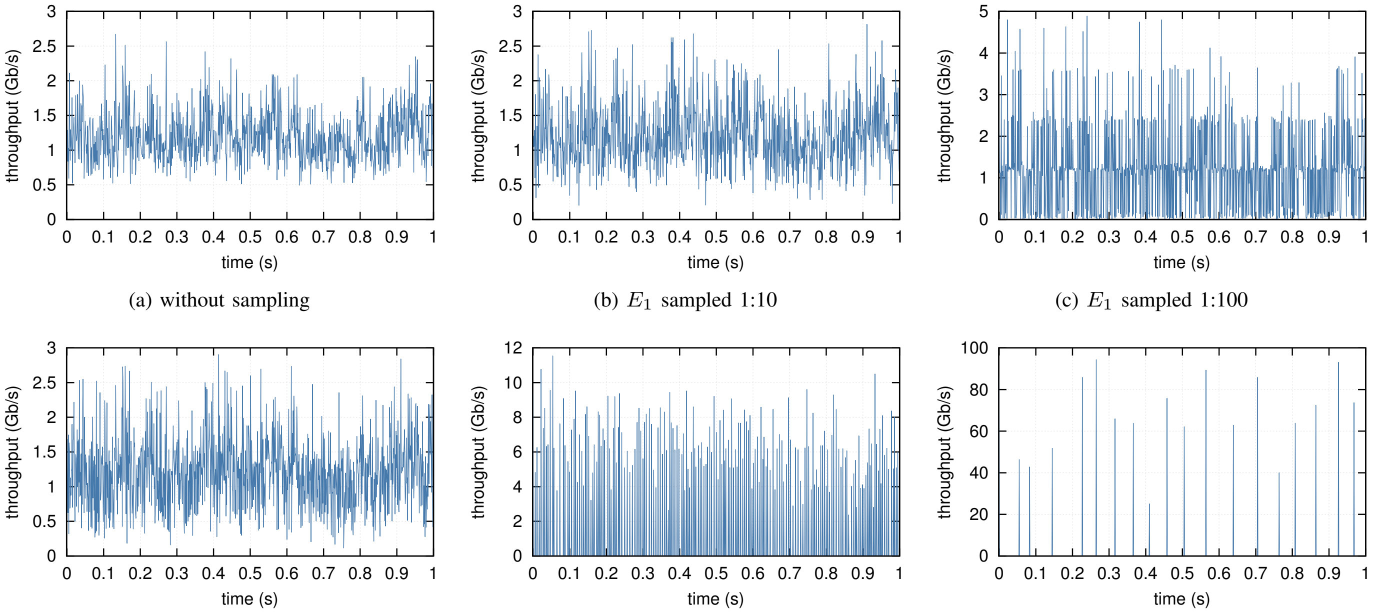
Exporting conditions. In the following experiments we consider two conditions for exporting sampled data from the sFlow agent to the collector. In the first condition E_1, packets are exported individually and as soon as they are sampled. This case aims at simulating the optimal scenario for link dimensioning, given that all sampled packets will have an individual and chronologically ordered timestamp. Clearly, this condition has the disadvantage of excessive traffic between agent and collector, specially in a distributed monitoring environment. Note that E_1 is the same condition adopted in the previous experiments of Section VII, where we focused on validating the sampling algorithms only. The second condition E_2 implements the original sFlow operation as defined in [6]. In E_2 the sampled packets are exported when the maximum buffer size is reached (*i.e.,* a full sFlow datagram), or when the oldest sampled packet in the buffer reaches the maximum waiting time of 1 second. Considering a capture size of 128 bytes, and a maximum datagram size of 1500 bytes, the agent's buffer is full when 10 packets are sampled. Note that if 10 packets are exported together, they will share the same timestamp (*i.e.,* datagram's timestamp) and, consequently, they will be accounted to the time interval $A_i(T)$. The impact of this aggregation is further studied in this section. Additionally, in the following experiments we vary the capture size.

Fig. 13 shows several traffic time series for the same trace from location B, created from data monitored under various circumstances. From these plots one can clearly see the challenges brought to link dimensioning from the sampling and exporting processes. Fig. 13a shows the time series of the complete data (*i.e.,* not sampled). The time series of the sampled data exported using the condition E_1 are shown in Fig. 13b and 13c. Fig. 13d– 13f show the traffic time series of sampled data exported using the condition E_2 with the maximum buffer size set to 10 packets ($BS = 10$). All traces in our dataset have a duration of 15 minutes. However, since $T = 1$ms, for visualization purposes only one second of the time series is plotted (the same second for all plots though).

By comparing the time series of the complete traffic with the ones sampled 1:10 (*i.e.,* Fig. 13b and 13d) one can see that the latter two show a more bursty behavior. That is, inexistent traffic fluctuations are created possibly due to clustering of sampled packets into single time intervals (*i.e.,* sampled packets are too close in time). Visually, at this sampling rate, there is not much difference between the two exporting conditions and both satisfactorily approximate the ground-truth observed in the time series of the complete trace. Nonetheless, time series created with exporting condition E_2 are slightly more bursty than with E_1.

The problem of sampling traffic at unreasonable rates, combined with the problem of loss of individual packet timestamps, becomes evident in the time series of traces sampled 1:100 and 1:1000. Clearly, these sampling rates are too low for the volume of traffic in location B. In Fig. 13c the ON/OFF behavior is very strong, and it becomes even worse in Fig. 13e. In this case, many time intervals have a throughput of zero and others have their throughput overestimated from the scaling process in Eq. (4). That is because at least 10 packets, that originally were some milliseconds apart, are grouped into the same time interval defined by the datagram's timestamp, while neighboring time bins are empty, *i.e.,* no datagrams were exported during those empty time bins. Fig. 13f shows the time series of the traffic sampled 1:1000 and exported using condition E_2. In this case, the same problem is even further aggravated by the lower sampling rate. Traffic fluctuations at small timescales cannot be estimated from data sampled at such low rates and, therefore, the monitored data is useless for link dimensioning.

To quantify the loss on accuracy of link dimensioning, we sample the traces from the datasets of location B and C at rates 1:10 and 1:100 with E_2. Tables II and III present the average and standard deviation of $\hat{\varepsilon}$ (see Eq. (12)) obtained for all traces from locations B and C, respectively. For these experiments we use Eq. (10) to estimate the variance from sampled data, and we set $\varepsilon = 1\%$ in the dimensioning formula from Eq. (2). The conclusion from these results is that the buffer size BS (in number of packets) does not seem to play a major role in the accuracy of the link dimensioning (for those sizes we have tested). Once again, $\hat{\varepsilon}$ is actually influenced by the chosen sampling rate and obtained values are comparable to those in Fig. 7. For the majority of the cases the link dimensioning is successful, *i.e.,* $\hat{\varepsilon} \leq \varepsilon$. However, it is important to remember that one should be careful on assuming the link dimensioning was successful when $\hat{\varepsilon}$ equals or is close



(a) without sampling (b) E_1 sampled 1:10 (c) E_1 sampled 1:100

(d) E_2 sampled 1:10 (e) E_2 sampled 1:100 (f) E_2 sampled 1:1000

Fig. 13. Time series at $T = 1\text{ms}$ for a trace from location B. For the sake of clarity, only one second is shown.

TABLE II
$\hat{\varepsilon}$ FOR TRACES FROM B.

r	*BS*	$T = 1\text{ms}$		$T = 10\text{ms}$		$T = 1\text{s}$	
		$\hat{\varepsilon}$	σ	$\hat{\varepsilon}$	σ	$\hat{\varepsilon}$	σ
1 : 10	2	0.0052	0.0028	0.0072	0.0046	0.0105	0.0072
	5	0.0052	0.0028	0.0072	0.0046	0.0105	0.0072
	10	0.0024	0.0017	0.0070	0.0045	0.0105	0.0072
	20	0.0051	0.0028	0.0072	0.0046	0.0105	0.0072
1 : 100	2	0.0001	0.0002	0.0010	0.0016	0.0098	0.0073
	5	0.0001	0.0002	0.0010	0.0015	0.0098	0.0073
	10	0.0000	0.0000	0.0001	0.0002	0.0097	0.0073
	20	0.0001	0.0002	0.0010	0.0016	0.0099	0.0075

TABLE III
$\hat{\varepsilon}$ FOR TRACES FROM C.

r	*BS*	$T = 1\text{ms}$		$T = 10\text{ms}$		$T = 1\text{s}$	
		$\hat{\varepsilon}$	σ	$\hat{\varepsilon}$	σ	$\hat{\varepsilon}$	σ
1 : 10	2	0.0162	0.0052	0.0133	0.0038	0.0114	0.0069
	5	0.0162	0.0052	0.0133	0.0038	0.0115	0.0069
	10	0.0162	0.0052	0.0133	0.0038	0.0114	0.0069
	20	0.0162	0.0052	0.0133	0.0038	0.0115	0.0069
1 : 100	2	0.0014	0.0012	0.0047	0.0026	0.0093	0.0066
	5	0.0014	0.0012	0.0046	0.0025	0.0092	0.0065
	10	0.0014	0.0012	0.0047	0.0025	0.0095	0.0066
	20	0.0014	0.0012	0.0046	0.0025	0.0093	0.0065

to zero. As explained before, in Fig. 10 and 11, when $\hat{\varepsilon}$ is too low or equals to zero, the link dimensioning procedure might have excessively overestimated the actual required capacity. One of the reasons for such overestimation might be very high traffic peaks that the dimensioning approach tries to account for in the estimation.

C. Practical Considerations

Besides the configured sampling rate and sampled data exporting process, there are other issues with sFlow implementations that might have an impact on the quality of the measurements and, consequently, on the accuracy of the link dimensioning. sFlow datagrams use UDP as transport layer protocol. According to [6], UDP is more robust than a connection-oriented protocol for this purpose and the only effects on the performance of an overloaded system is a slightly increase in transmission delay and a greater number of packet losses. Although claimed as insignificant, packet loss might drastically reduce accuracy of operations such as link dimensioning. From real-world deployments we have also experienced very high numbers of packets loss. This might also be caused by running the sFlow agent jobs with a low priority on the exporting router or switch. Therefore, we generally advisable to avoid losses of sFlow datagrams as much as possible by giving agent jobs the right priority and by providing enough residual link capacity between the agents and the collectors if they are not physically located on the same host.

Finally, the sFlow protocol also defines and exports a counter called *sample_pool*. This counter gives total number of observed packets before sampling. The collector can use this counter to determine the actual effective sampling rate. This information could help to further improve the estimation of the traffic variance in section V. A thorough study is planned as future work.

IX. Conclusions

In this paper we have evaluated the impact of using sampled packets for link dimensioning. Many monitoring technologies implement sampling strategies to cope with the ever increasing traffic rates in high-speed links. Aiming at practicality, we validated a complete link dimensioning procedure using sampled data obtained from three widely deployed sampling algorithms.

Our findings show that, for any of the considered sampling algorithms and using the link dimensioning formula proposed in [2], [3], it is possible to have accurate link capacity estimations from sampled data provided that a reasonable sampling rate is chosen. Inappropriate sampling rates will likely provide sampled data that might be not enough to represent the actual traffic. Therefore, desirable characteristics such as Gaussianity might be lost and, ultimately, the link dimensioning procedure will underestimate or excessively overestimate the actual required capacity. Furthermore, the formulas to estimate traffic variance that we propose provide better results by taking into account the additional variance introduced by the sampling process. Nonetheless, we consider as future work the extension of such formulas to also account for variable packet sizes, which might give us better estimations at very short timescales.

We have also assessed the impact of the concrete sFlow export procedure and its parameters on the accuracy of the capacity estimation. Our results show that, on one hand, the buffer size does not influence the link dimensioning. On the other hand, however, the lack of individual packet timestamps in the sFlow protocol can invalidate the sampled data for link dimensioning purposes. However, higher sampling rates can mitigate these effects.

Finally, a more detailed analysis of the sFlow tool, including the use of the *sample_pool* counter and deployment in a real network is planned as future work. We also plan to study the use of adaptive sampling algorithms for link dimensioning purposes.

ACKNOWLEDGEMENTS

The authors would like to thank Paul Aitken (Cisco Systems Inc.), Jürgen Schönwälder (Jacobs University Bremen), Ronald van der Pol (SURFnet), Pieter Venemans (TNO) and Bert Wijnen (RIPE-NCC) for their valuable support and contributions.

This work has been partially funded by Flamingo, a Network of Excellence project (318488) and by UniverSelf project (257513), both supported by the European Commission under its Seventh Framework Programme.

APPENDIX

Variance Estimation with Bernoulli Sampling

Let μ and $v(T)$ be the mean and the variance of the observed traffic per time interval before sampling. Under the assumptions given in section V-A, it holds

$$\begin{aligned} \mu &= E[P]E[S] \\ v(T) &= E[P]Var[S] + E[S]^2Var[P] \end{aligned}$$

Let μ' and $v'(T)$ be the mean and the variance of the sampled traffic per time interval. The “trick” with the 0-size packets described in section V-A gives

$$\begin{aligned} \mu' &= E[P]E[S'] \\ &= pE[P]E[S] = p\mu \\ v'(T) &= E[P]Var[S'] + E[S']^2Var[P] \\ &= p^2E[P](Var[S] + (\frac{1}{p} - 1)E[S^2]) + p^2E[S]^2Var[P] \\ &= p^2(v(T) + (\frac{1}{p} - 1)E[P]E[S^2]) \end{aligned}$$

The last equation can be solved for $v(T)$. With $r = 1/p$ and $v_{est}(T) = r^2v'(T)$, we obtain the desired estimation of $v(T)$ shown in Eq. (9).

Variance Estimation with 1-in-N and sFlow Sampling

Again, let μ and $v(T)$ be the mean and the variance of the observed traffic. As explained in section V-B, we assume that the number P'_i of sampled packets in time interval i is i.i.d. as P' with $P' = P/N$, where P is the number of packets per time interval before sampling. Thus, the mean μ' and the variance $v'(T)$ of the sampled traffic per time interval are

$$\begin{aligned} \mu' &= E[P']E[S] \\ &= \frac{1}{N}E[P]E[S] = \frac{\mu}{N} \\ v'(T) &= E[P']Var[S] + E[S]^2Var[P'] \\ &= \frac{1}{N}E[P]Var[S] + \frac{1}{N^2}E[S]^2Var[P] \\ &= \frac{1}{N^2}\left(v(T) + (N-1)E[P]Var[S]\right) \end{aligned}$$

By solving for $v(T)$ and using $r = N$ and $v_{est}(T) = r^2v'(T)$, we obtain the desired estimation of $v(T)$ shown in Eq. (10).

REFERENCES

[1] Cisco Systems Inc., “How To Calculate Bandwidth Utilization Using SNMP,” http://www.cisco.com/image/gif/paws/8141/calculate_bandwidth_snmp.pdf, 2005, online. Accessed May 2014.

[2] H. van den Berg, M. Mandjes, R. van de Meent, A. Pras, F. Roijers, and P. Venemans, “QoS-aware bandwidth provisioning for IP network links,” *Elsevier Computer Networks*, vol. 50, no. 5, pp. 631–647, 2006.

[3] A. Pras, L. J. M. Nieuwenhuis, R. van de Meent, and M. R. H. Mandjes, “Dimensioning Network Links: A New Look at Equivalent Bandwidth,” *IEEE Network*, vol. 23, no. 2, pp. 5–10, 2009.

[4] M. M. Hulboj and R. E. Jurga, “CERN Investigation of Network Behaviour and Anomaly Detection,” in *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection*, ser. RAID, 2009, pp. 353–354.

[5] E. Jasinska, “sFlow: I can feel your traffic,” in *Proceedings of the 23rd Chaos Communication Congress*, ser. 23C3, 2006, pp. 1–8.

[6] P. Phaal, S. Panchen, and N. McKee, “InMon Corporation’s sFlow: A Method for Monitoring Traffic in Switched and Routed Networks,” RFC 3176, 2001.

[7] R. de O. Schmidt, R. Sadre, A. Sperotto, and A. Pras, “Lightweight Link Dimensioning using sFlow Sampling,” in *Proceedings of the 9th International Conference on Network and Services Management*, ser. CNSM, 2013, pp. 14–18.

[8] T. Zseby, M. Molina, N. Duffield, S. Niccolini, and F. Raspall, “Sampling and Filtering Techniques for IP Packet Selection,” RFC 5475, 2009.

[9] M. Charalambides, D. Tuncer, L. Mamatas, and G. Pavlou, “Energy-Aware Adaptive Network Resource Management,” in *Proceedings of the IFIP/IEEE International Symposium on Integrated Network Management*, ser. IM, 2013, pp. 369–377.

[10] B. Anjum, H. Perros, X. Mountrouidou, and K. Kontovasilis, "Bandwidth allocation under end-to-end percentile delay bounds," *International Journal of Network Management*, vol. 21, no. 5, pp. 536–547, 2011.

[11] C. Fraleigh, F. Tobagi, and C. Diot, "Provisioning IP Backbone Networks to Support Latency Sensitive Traffic," in *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications*, ser. INFOCOM, 2003, pp. 375–385.

[12] M. Mandjes and R. van de Meent, "Resource Dimensioning Through Buffer Sampling," *IEEE/ACM Transactions on Networking*, vol. 17, no. 5, pp. 1631–1644, 2009.

[13] W. Lautenschlaeger and F. Feller, "Light-Weight Traffic Parameter Estimation for On-Line Bandwidth Provisioning," in *Proceedings of the 24th International Teletraffic Congress*, ser. ITC, 2012, pp. 1–8.

[14] R. de O. Schmidt, A. Sperotto, R. Sadre, and A. Pras, "Towards Bandwidth Estimation using Flow Measurements," in *Proceedings of the 6th IFIP WG 6.6 International Conference on Autonomous Infrastructure, Management, and Security*, ser. AIMS, 2012, pp. 127–138.

[15] R. de O. Schmidt, R. Sadre, A. Sperotto, H. van den Berg, and A. Pras, "A Hybrid Procedure for Efficient Link Dimensioning," *Elsevier Computer Networks*, vol. 67, pp. 252–269, 2014.

[16] C. Barakat, P. Thiran, G. Iannaccone, C. Diot, and P. Owezarski, "Modeling Internet backbone traffic at the flow level," *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 1–12, 2003.

[17] T. Bonald, P. Oliver, and J. Roberts, "Dimensioning high speed IP access networks," in *Proceedings of the 8th International Teletraffic Congress*, ser. ITC'03, 2003, pp. 241–251.

[18] B.-Y. Choi, J. Park, and Z.-L. Zhang, "Adaptive Random Sampling for Traffic Load Measurement," in *Proceedings of the 38th IEEE International Conference on Communications*, ser. ICC, 2003, pp. 1552–1556.

[19] ——, "Adaptive Packet Sampling for Accurate and Scalable Flow Measurement," in *Proceedings of the 47th IEEE Global Telecommunications Conference*, ser. GLOBECOM, 2004, pp. 1448–1452.

[20] R. Vilardi, L. A. Grieco, C. Barakat, and G. Boggia, "Lightweight Enhanced Monitoring for High-Speed Networks," *Transactions on Emerging Telecommunications Technologies*, pp. 1–19, 2013.

[21] D. Brauckhoff, B. Tellenbach, A. Wagner, M. May, and A. Lakhina, "Impact of Packet Sampling on Anomaly Detection Metrics," in *Proceedings of the 6th ACM SIGCOMM Internet Measurement Conference*, ser. IMC, 2006, pp. 159–164.

[22] J. Mai, C.-N. Chuah, A. Sridharan, T. Ye, and H. Zang, "Is Sampled Data Sufficient for Anomaly Detection?" in *Proceedings of the 6th ACM SIGCOMM Internet Measurement Conference*, ser. IMC, 2006, pp. 165–176.

[23] Cisco Systems Inc., "Cisco IOS Flexible NetFlow Configuration Guide. Release 12.4T," http://www.cisco.com/en/US/docs/ios/fnetflow/configuration/guide/12_4t/fnf_12_4t_book.pdf, 2008, online. Accessed May 2014.

[24] ——, "Random Sampled NetFlow," www.cisco.com/en/US/docs/ios/12_0s/feature/guide/nfstatsa.pdf, 2005, online. Accessed May 2014.

[25] InMon Corp., "sFlow Agent Software Description," http://www.inmon.com/technology/InMon_Agentv5.pdf, 2003, online. Accessed May 2014.

[26] Paolo Lucente, "pmacct: a free open-source traffic accounting tool," http://www.pmacct.net/lucente_pmacct_esnog10.pdf, 2012, online. Accessed May 2014.

[27] J. Kilpi and I. Norros, "Testing the Gaussian approximation of aggregate traffic," in *Proceedings of the 2nd ACM SIGCOMM Internet Measurement Workshop*, ser. IMW, 2002, pp. 49–61.

[28] R. van de Meent, M. Mandjes, and A. Pras, "Gaussian Traffic Everywhere?" in *Proceedings of the IEEE International Conference in Communications*, ser. ICC, 2006, pp. 573–578.

[29] R. de O. Schmidt, R. Sadre, and A. Pras, "Gaussian Traffic Revisited," in *Proceedings of the 12th IFIP Networking*, ser. Networking, 2013, pp. 1–9.

[30] B. M. Brown and T. P. Hettmansperger, "Normal Scores, Normal Plots and Tests for Normality," *Journal of the American Statistical Association*, vol. 91, no. 436, pp. 1668–1675, 1996.

[31] sFlow, "Sampling rates," http://blog.sflow.com/2009/06/sampling-rates.html, 2009, online. Accessed May 2014.

Ricardo de Oliveira Schmidt is a Postdoctoral researcher at the Design and Analysis of Communication Systems Group (DACS) of the University of Twente, The Netherlands. He received a PhD degree from the University of Twente in 2014. He also received a M.Sc. degree in Computer Science from the Federal University of Pernambuco (UFPE) in 2010, and a B.Sc. degree in Computer Science from the University of Passo Fundo (UPF) in 2007, both in Brazil. His research interests include network management, traffic measurements, DNS, routing and addressing.

Ramin Sadre is an Assistant Professor at the Université catholique de Louvain, Belgium. He received a Ph.D. degree from the University of Twente for his thesis titled "Decomposition Based Analysis of Queuing Networks". His research interests include traffic modeling, the design and analytical performance evaluation of communication systems, and the design of network intrusion detection systems.

Anna Sperotto is a Postdoctoral researcher at the Design and Analysis of Communication Systems Group (DACS) of the University of Twente, The Netherlands. She received a M.Sc. degree in Computer Science from the Ca'Foscari University, Venice, Italy, in 2006 and a Ph.D. degree from the University of Twente, in 2010. Her main topics of interest include intrusion detection and traffic monitoring and modeling.

Hans van den Berg received the M.Sc. and Ph.D. degree in mathematics (stochastic operations research) from the University of Utrecht, The Netherlands, in 1986 and 1990 respectively. From 1986, he worked at the Centre for Mathematics and Computer Science (CWI), Amsterdam. In 1990, he joined KPN Research (now TNO Information and Communication Technology, since January 2003). He is particularly working on performance analysis, traffic management and QoS provisioning for wired and wireless multi-service communication networks (ATM, IP, UMTS, WLAN, ad-hoc networks). Currently, he is a Senior Research Member and leader of the QoS group within TNO ICT, Delft, The Netherlands. Since July 2003 Hans van den Berg has a part-time position as full professor within the faculty of Electrical Engineering, Mathematics and Computer Science at the University of Twente.

Aiko Pras is professor in the area of Network Operations and Management at the University of Twente, the Netherlands and member of the Design and Analysis of Communication Systems Group. He received a PhD degree for his thesis titled "Network Management Architectures". His research interests include network management technologies, network monitoring, measurements and security. He is chairing the IFIP Technical Committee on "Communications Systems", and is Coordinator of the European Network of Excellence on "Management of the Future Internet" (FLAMINGO). He is steering committee member of several conferences, including IM/NOMS and CNSM, and series/associate editor of ComMag, IJNM and TNSM.