Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk, and Ralf Steinmetz
Multimedia Communications Lab, Technische Universität Darmstadt, Germany
{rhaban.hark|nils.richerzhagen|bjoern.richerzhagen|amr.rizk|ralf.steinmetz}@kom.tu-darmstadt.de

***Abstract*—Next generation Software-defined Networks (SDN) aim at deeply programmable switches which can be leveraged by SDN controllers to offload self-contained, logically persistent tasks. One such task is flow and network monitoring, specifically, fault detection and loss estimation, which is essential for SDN applications that provide quality of service guarantees under network dynamics. In this work, we devise, implement, and evaluate fault detection and loss estimation techniques built upon tasks devolved to SDN switches. Subsequently, we contribute *(i)* an analysis and empirical evaluation of the benefits and costs of different packet loss estimators depending on the network conditions; *(ii)* a case study showing how an adaptive monitoring framework which flexibly exchanges the estimation techniques would retain a thoroughly good fidelity while optimizing the monitoring costs.**

I. Introduction

Software-defined Networking (SDN) has become the de-facto standard architecture for new generation networks. A recent paradigm change within the SDN research indicates a shift from first generation strict network partitioning of intelligent SDN controllers and fast-but-dumb switches to more flexible, logically persistent and programmable SDN switches. The aim of this shift is to move from a rigid separation of programmable elements in controllers and stateless forwarding elements in switches [16], [18], [23] to offloading tasks that require persistent packet state information into schedulers [8], [10], [32]. A significant example of tasks requiring per packet state information is loss estimation, fault detection and flow monitoring. Many SDN research concepts, e.g. [2], assume regularly updated if not real-time monitoring information. The ability to detect packet loss surges and network faults is crucial for accurate network management. This monitoring information forms a basic building block for adaptive behavior, such as dynamic network resource slicing in multi-tenancy scenarios [15] or upgrade and migration of Virtual Network Functions. Such networks possess requirements that go beyond the abilities of traditional monitoring solutions such as sFlow [28] and NetFlow [14], e.g., by estimating flow correlations [12], detecting heavy hitters [21], and estimating traffic matrices [19].

Current monitoring frameworks usually provide *one* method of choice for estimating packet loss [35] which might not be as *accurate or as efficient* as required depending on the network and traffic conditions. Subsequently, in this work we address two research gaps that are associated with this problem, i.e., *(i)* understanding the varying characteristics of different measurement and estimation techniques for fault detection and packet loss in SDN, and *(ii)* analyzing the required adaptivity of monitoring applications depending on the network environment. This adaptivity not only entails adjusting measurement parameters depending on the network conditions, but rather replacing the entire fault detection technique for higher accuracy or lower measurement overhead.

The contributions of this paper are as follows: First, we devise different measurement techniques for fault detection and packet loss estimation in SDN that fall into two categories: *(i)* controller-based match-action techniques and *(ii)* offloaded switch-based techniques. We propose and analyze different loss estimators based on these measurement techniques and finally compare their benefits and costs in different scenarios. Ultimately, we discuss how to combine different loss estimation techniques into an adaptive monitoring framework which minimizes monitoring overhead depending on the network state and the required measurement accuracy.

The remainder of the paper is structured as follows: Section II defines the techniques that we investigate in this paper including first insights into their assets and limitations. Subsequently, Section III evaluates and compares those techniques regarding changing network conditions and requirements. Section IV provides a cost assessment for each technique followed by a discussion of an adaptive technique selection strategy and a corresponding case study. Section V summarizes relevant related works, while Section VI concludes the paper and provides an outlook about ongoing and future work.

II. Loss Measurement Techniques

In this section, we describe a number of techniques that we devised for fault detection and packet loss measurement. We use the term *technique* to denote a collection of procedures and formats to measure a network characteristic such as the loss rate or the state of a network entity, e.g., switch, middlebox or link. In a nutshell we require:

1) The desired information, *e.g. the loss rate*[1].
2) The captured metric and how it is processed to derive the desired information, *e.g. from the number of observed packets N* to the average loss rate, *as*

[1]In this work we use the terms loss rate and packet loss probability interchangeably. This is generally true under very mild conditions of stationarity and ergodicity of the loss process [3]. The rate is the intensity of the simple stationary 0 (no loss) - 1(loss) process $\{l_n\}_{n\in\mathbb{Z}}$, while the loss probability is defined as $\mathsf{E}[l_0] = P_L$.

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

$\lim_{N\to\infty} \frac{1}{N} \sum_{i=1}^{N} l_i$ *with* $l_i \in \{0, 1\}$*, where* 1 *indicates packet loss.*

3) A procedure to transfer this information (either raw or processed) to a monitoring entity for further analysis and/or processing, *e.g. active polling*.

Generally, we would like to compare the performance of techniques that desire the same information (first property). If the captured information (second property) is identical, however, the processing differs then the comparison is quantitatively easy. Note, however, that in this paper we also consider vastly diverse techniques that seek the same information (loss rate) to explore their design space and also *leverage* their different strengths. In addition, some techniques only differ through the information transfer procedure (third property).

In the following, we compare four techniques to infer the loss rate P_L of a network link of interest as depicted in Fig. 1: Using *(i) packet counters* on switches; *(ii) in-line sent packet counter values*; *(iii) packet samples*; *(iv) active network probing*. The network links of interest are defined as a contiguous connection between two ports of two SDN network nodes that are either connected via one direct link or multiple links as in a network segment (see Fig. 1). Note that for fault detection and real-time monitoring we are mainly interested in a running estimate of the packet loss rate P_L. In the following, we first describe the loss estimation techniques before providing a comparison of the techniques in Sect. III.

A. Legacy packet Counters (LC):

First, we investigate a technique that uses simple packet counters that are common, e.g., in OpenFlow switches but also in legacy networks using various management protocols like SNMP. Basically, switches increment their packet counters for every forwarded packet. OpenFlow provides counters on a flow-level basis which are usually actively fetched by the controller. Later OpenFlow versions (1.5) provide optional push-based counter value transmission. In the following, we describe a technique that infers the flow loss rate (specifically estimates the packet loss probability) from packet counters. For the sake of illustration we describe this technique for a single monitored flow, however, this technique can be easily adapted to collections of data flows using OpenFlow rules.

Estimating the loss probability: Consider the illustrated link in Fig. 1. Let $c_s(t_i)$ be the cumulative packet counter value at a discrete time t_i for $i \in \mathbb{N}$ for the flow of interest at switch s where $s \in \{in, eg\}$ denotes the ingress or egress switch, respectively. Assuming negligible queueing we calculate the number of lost packets L_i in the time interval $[t_{i-1}, t_i]$ as

$$L_i = (c_{in}(t_i) - c_{in}(t_{i-1})) - (c_{eg}(t_i) - c_{eg}(t_{i-1}))\,. \quad (1)$$

Subsequently, we calculate the packet loss probability estimate $\hat{P}_{L,i}$ for the ith time interval as

$$\hat{P}_{L,i} = \frac{L_i}{c_{in}(t_i) - c_{in}(t_{i-1})} = 1 - \frac{c_{eg}(t_i) - c_{eg}(t_{i-1})}{c_{in}(t_i) - c_{in}(t_{i-1})}. \quad (2)$$

Note that we are mainly interested in a running estimate of the loss rate, i.e., $\hat{P}_{L,i}$, for fault detection and real-time monitoring applications. Under stationary packet losses it is straightforward to calculate the steady state loss rate P_L.

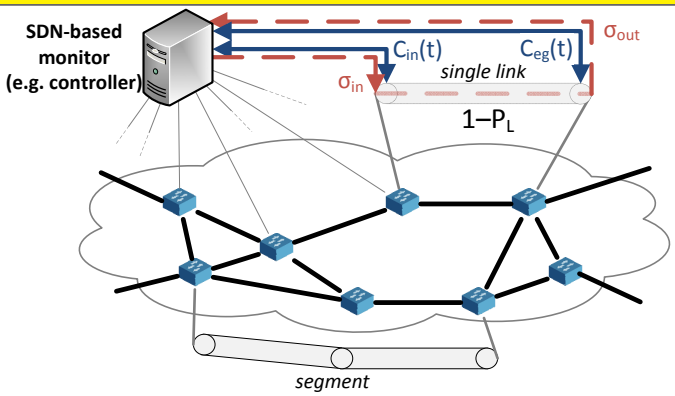


Fig. 1: Conceptual overview of SDN loss estimation / fault detection including relevant measurement metrics.

Limitations of Legacy Counters (LC): This technique has some limitations arising from unsynchronized counter requests. A controller may request statistics containing counter values from the ingress and egress switch at the same time, however, the time a statistic request reaches a switch may vary, e.g. due to delay variations on the control paths or due to load dependent processing times at the switches [11]. Another limitation is the number of packets *in flight*, e.g., that are queued. These packets introduce a bias in $\hat{P}_L$. This bias in the average loss rate can, however, be accounted for using information on the average queue lengths. Although one could argue that the packets in-flight in one estimation interval $[t_{i-1}, t_i]$ are carried over to the next interval $[t_i, t_{i+1}]$ and also that, for large estimation intervals the impact of the number of packets in-flight diminishes, both mentioned limitations lead to uncertainty when loss is calculated using legacy packet counters (LC). Adding these uncertainties to the model gives:

$$\hat{P}_{L,i} = 1 - \frac{c_{eg}(t_i + t^{\epsilon}_{i,eg}) - c_{eg}(t_{i-1} + t^{\epsilon}_{i-1,eg})}{c_{in}(t_i + t^{\epsilon}_{i,in}) - c_{in}(t_{i-1} + t^{\epsilon}_{i-1,in})}, \quad (3)$$

where $t^{\epsilon}_{i,in/eg}$ is the time delay between the request timestamp t_i and when the counter is actually reported. Minimizing the time error $t^{\epsilon}_{i,in/eg}$ as well as taking estimates of the average in-flight packets into account yields better estimates in (2).

B. In-line packet Counters (IC):

This technique uses production traffic based counter updates from the ingress and egress switch to infer the loss probability. The difference to the polling-based legacy packet counter (LC) technique is that switches send the counter values *in-line* with other control messages, thus, the updates arrive at the controller at irregular times. For this, we assume switches capable of piggybacking packet counter values to the control messages that are sent to the controller. Although to the best of our knowledge no such piggybacking mechanism is standardized, the threshold-based statistic trigger from the (not yet widely used) OpenFlow version 1.5[2] is a good example

[2]https://www.opennetworking.org/sdn-resources/technical-library, accessed 21.12.2016

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

of a comparable approach with similar properties. One main advantage of this technique is that there are no extra control messages necessary for monitoring purposes. However, due to the irregular counter updates, the controller cannot compare counter values from the ingress switch and the egress switch directly. Hence, to estimate the loss probability, we compare throughput changes (in packets per second) both at the ingress and the egress.

Estimating the loss probability: First, we use the in-line counters to calculate the most recent incoming packet throughput as

$$\hat{R}_{in}(t_{i-1}, t_i) = \frac{c_{in}(t_i) - c_{in}(t_{i-1})}{t_i - t_{i-1}}, \tag{4}$$

while $c_{in}(t_i)$ indicates the counter value at the ingress switch at timestamp t_i. The outgoing packet throughput $\hat{R}_{eg}(t'_{i-1}, t'_i)$ is calculated analogously while t_i need not be equal to t'_i. Using the throughput estimates of the ingress and egress switches, we estimate the loss probability on every (ingress or egress) counter update using

$$\hat{P}_{L,i} = 1 - \frac{\hat{R}_{eg}(t_{k-1}, t_k)}{\hat{R}_{in}(t_{i-1}, t_i)}, \tag{5}$$

where $t_k \leq t_i$ is the most current time a control message is sent to the egress switch.

Limitations of in-line counters (IC): This technique relies on the frequency of control message that are exchanged between the controller and the switches. As control protocols such as OpenFlow use keep-alive messages which are sent when no control communication is present, this frequency has a defined minimum (values of 15s are common[3]). We model the IC limitation due to no synchronization using the difference between $[t_{k-1}, t_k]$ and $[t_{i-1}, t_i]$ in (5) as

$$\hat{P}_{L,i} = 1 - \frac{R_{eg}(t_{k-1}, t_k)}{R_{in}(t_{k-1} + t^{\Delta}_{k-1}, t_k + t^{\Delta}_{k})}, \tag{6}$$

where we used t^{Δ}_k to indicate the time difference to t_i. The accuracy of this estimator is improved when minimizing t^{Δ}_k.

C. Sampling-based packet Counters (SC):

Maintaining accurate counter values in high-speed software-defined networks is a challenging task requiring expensive and power hungry TCAM counters. In the following, we propose a sampling-based counter approach which does not need to take every traversing packet into account and increments a counter only with a fixed probability to decrease the overhead on a switch allowing the use of cheaper counter implementations (e.g. SRAM). Sampling-based counters provide another possibility to determine the loss rate $\hat{P}_{L,i}$. This technique is similar to the legacy counter approach (LC) while having additional uncertainty due to sampling. Note that this technique is fundamentally different from the technique in [31] since we do not sample and forward packets to the controller. A mathematical illustration of the difference between the two methods is given in [12].

Estimating the loss probability: Consider a stochastic sampling of arriving packets according to independent and identically distributed (iid) Bernoulli random variables taking values in $\{0, 1\}$ standing for (take no sample) and (take sample), respectively. The sampling probability at the ingress switch and at the egress switch are denoted p_{in} and p_{eg}, respectively. We estimate the number of incoming packets as $\hat{c}_{in}(t_{i-1}, t_i) = {}^{c^*_{in}(t_{i-1}, t_i)}/_{p_{in}}$, where $c^*_{in}(t_{i-1}, t_i)$ is the number of sampled packets at the ingress switch in the time interval $[t_{i-1}, t_i]$. We estimate the number of outgoing packets $\hat{c}_{eg}(t_{i-1}, t_i)$ analogously, so that we estimate the loss probability for interval $[t_{i-1}, t_i]$ as

$$\hat{P}_{L,i} = 1 - \frac{\hat{c}_{eg}(t_{i-1}, t_i)}{\hat{c}_{in}(t_{i-1}, t_i)}. \tag{7}$$

Limitations of sampling-based counters (SC): In addition to the timing problem which we stated for the legacy counter (LC) technique, the sampling based approach is also vulnerable to low traffic rates. A sampling rate of $p_{in} = p_{eg} = 1$ recovers the legacy counter technique, while a sampling rate of $p_{in} = p_{eg} = 0.1$ requires *on average* $1/p_{in}$, i.e., ten times, the number of packets to have the same number of samples. This simply implies that the sampling-based counters trade accuracy for less overhead.

D. Active Probing (PR):

Active probing - a known technique to infer network properties - provides estimates solely based on injected extra probe traffic. In our setting, this requires controller knowledge, e.g., appropriate Openflow rules, making it difficult to be offloaded to switches. Here, the controller sends the active probing packets in each estimation interval to the ingress switch which forwards the packets to the egress switch. The egress stream is then collected and processed for statistics by the controller. The controller can craft the probing packet stream in different manners. In this work, we limit to two simple types: a smooth constant rate stream or a burst probing stream. Note, that under equal rate probing bursts remove the risk of in-flight probes and lead to shorter but higher load on the network. The burst rate must be set carefully to avoid artificially congesting the network.

Estimating the loss probability: We define $\sigma_{in,i}$ as the number of packets sent out in interval $[t_{i-1}, t_i]$. Then, given the number of received probes $\sigma_{out,i}$ at the end of interval i we calculate the loss rate estimate as

$$P_{L,i} = 1 - \frac{\sigma_{out,i}}{\sigma_{in,i}}. \tag{8}$$

Limitations of active probing (PR): The main limitation of actively probing communication networks is the induced extra load which perturbs the network state. Active probing has to be carefully designed such that it does not introduce artificial packet loss which yields biased estimates $\hat{P}_{L,i}$.

[3]Dell OpenFlow Deployment and User Guide: echo-request interval http://www.dell.com/support/manuals, accessed Jan 20, 2017

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

TABLE I: Relevant notation and parameters. Default values are underlined.

Notation	Values	Description
r [pkts/s]	10^2, $\underline{10^3}$, 10^4, 10^5	Average packet rate
–	Poisson, Bursty	Traffic model
$[t_i, t_{i+1}]$ [s]	2, 5, 45, 180	Estimation interval length
α	0.2	EWMA smoothing factor
p_{in}, p_{eg}	2^{-2}, $\underline{2^{-4}}$, 2^{-6}, 2^{-8}, 2^{-10}	Sampling probability (ingress/egress)
P_L	deterministic, Markov (bursty)	Loss model

III. Technique Comparison

In this section, we investigate the techniques described in Section II with respect to their accuracy under *(i)* different traffic conditions, *(ii)* various estimation interval lengths, and *(iii)* for different packet loss burstiness. Here, we consider the accuracy in terms of the estimation error $\hat{P}_L - P_L$, where $\hat{P}_L$ is the loss rate estimate and P_L is the given ground truth. For emulated random losses, we consider the average estimation error. Table I shows the relevant system parameters and, if not stated otherwise, underlined parameters are the default values.

A. Technique implementation

We analyze the techniques in a test environment within the GENI [7] testbed based on a topology such as the one sketched in Figure 1. We use a central Floodlight[4] SDN controller, capable of communicating with the network devices, while Open vSwitches [27] run on the network devices to forward the emulated UDP traffic. We emulate losses using the Linux kernel traffic scheduler *tc* on a link between a rack at Stanford and a rack at the University of Illinois.

For the legacy packet counter technique **(LC)** we use the OpenFlow protocol version 1.3. Here, the controller polls the statistic request for the flow of interest at the end of every estimation interval. Once it receives statistics from both (ingress and egress) switches, it extracts the number of incoming packets and outgoing packets using the previous statistics to get the number of lost packets as described in Section II-A.

We implement the in-line packet counter technique **(IC)** by emulating control-traffic events as they may occur in production networks. We trigger such switch-to-controller message events with piggybacked counter values according to an approximate Poisson process with inter-arrival times from a truncated exponential distribution ($\lambda = 15$) with maximum of 15 seconds, as this is a common default value for echo messages of OpenFlow switches in the absence of production control messages. Further, when a counter value reaches the controller, it updates the packet throughput estimation using an exponentially weighted moving average such as $R_{MA}(t) = \alpha \cdot R'(t) + (1-\alpha) \cdot R_{MA}(t-1)$, where $R'(t)$ is the instantaneous measured throughput and $R_{MA}(t)$ is the smoothed throughput at time t.[5] This smoothing factor avoids rough estimates on frequent, between ingress and egress unbalanced counter updates. Note that α resembles a tradeoff between the responsiveness and the accuracy under stable loss rates. Finally, we compare the ingress and egress throughput to estimate the loss probability as described in Section II-B.

To implement the sampling based packet counter **(SC)** we use sFlow [28]. A built-in agent on the switches sends packet samples to the controller, where an sFlow collector takes the samples to determine the sample count. Denote the packet sample probability at the ingress switch as p_{in} and at the egress as p_{eg}. In an auxiliary evaluation, cf. Figure 4b, we found a sampling probability of $p_{in} = p_{eg} = 2^{-4}$ to be an empirically reasonable choice as higher sampling rates do not improve the estimates. At the end of the estimation interval the sampled collector compares the counters to determine the number of lost packets as described in Section II-C and resets the counter.

The active probing approach **(PR)** is, in conjunction with LC, implemented using basic OpenFlow features. The controller prepares a set of minimum-sized probing packets and in each estimation interval it dispatches these packets to the ingress switch towards the measured link with minimal delay. The egress switch intercepts these packets and sends them to the controller, where the number of retrieved packets is used to calculated the loss rate (cf. Section II-D).

B. Traffic rate sensitivity

We first analyze the accuracy of the techniques with regard to the amount of forwarded traffic the techniques can utilize. Therefore, we change the packet rates of the Poisson traffic arrival processes to 10^2, 10^3, 10^4, and 10^5 $\frac{pkt}{s}$, respectively, with exponentially distributed inter-arrival times. Figures 2a to 2d depict the accuracy in terms of absolute estimation error per technique. For the lowest packet rate of $10^2 \frac{pkt}{s}$, we see that the legacy counter technique (LC), the active probing approach (PR), and the in-line counter (IC) achieve reasonable results, having a median error around 2% or less. The sampling-based approach (SC) delivers poor results and suffers, in particular, when the traffic rates are low (compare Figure 2a to Figure 2d).

Hence, for the default estimation interval of $2s$ the legacy counter technique (LC) improves with higher traffic. Note that LC and the in-line packet counter technique (IC) possess reasonable accuracy given low packet rates. Active probing (PR) is independent of the packet rate, however, a combination of high traffic load and (PR) should be avoided to prevent biasing the measurements through network overloading. An important observation is that the sampling-based technique (SC) is only feasible with fairly high packet rates.

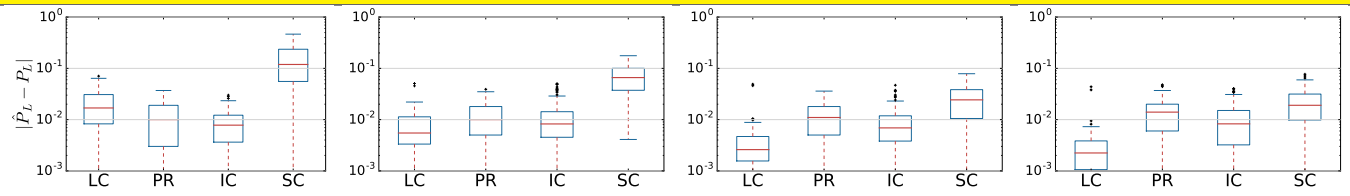
C. Impact of the estimation interval length

In a second investigation, we measure the absolute estimation error for different lengths of the estimation interval which we show in Figure 3. where $\hat{P}_L$ is the estimate and P_L is

[4] http://www.projectfloodlight.org/floodlight/, accessed Jan 04, 2017

[5] Here, we empirically optimize the EWMA parameter $\alpha = 0.2$, however, we note that it basically controls the tradeoff of the oscillation and responsiveness of the measured throughput.

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .



(a) 100 pkt/s. (b) 1.000 pkt/s. (c) 10.000 pkt/s. (d) 100.000 pkt/s.

Fig. 2: Different techniques under varying average traffic rates with Poisson traffic arrivals, deterministic loss ($P_L = 0.5$), and a estimation period of $2s$.

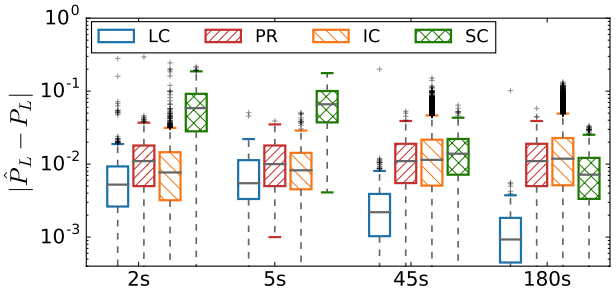


Fig. 3: The impact of the different loss estimation frequencies (time between two consecutive estimates) for Poisson traffic and deterministic loss rate P_L.

the ground truth. Here, LC improves with longer intervals, PR is independent of the interval length, and SC requires larger intervals, since it requires many samples to produce meaningful results. However, as IC is independent of the interval length, its accuracy does obviously not change.

D. Non-stationary loss rate jumps (Fault detection)

We consider link faults as non-stationary changes in the loss rate as shown in the exemplary excerpt from a testbed emulation in Fig. 4a where the changes occur at $\{73, 124, 185, 226\}$ seconds. Here, we keep this example simple to illustrate how the different techniques react to such loss rate changes. In the next section we introduce a Markov loss model that we used in the next emulations. A qualitative observation of the time (respectively the number of samples) needed by a technique to converge to the ground truth P_L after a loss rate jump shows that the different techniques possess different sensitivities. For example, LC, PR, and SC are able to quickly conflate both the estimate and the ground truth P_L values while inline packet counters IC are generally slower to converge. The convergence speed of IC partially depends on the smoothing factor α (here $\alpha = 0.2$) as motivated in Sect. III-A. Note that the sampling-based approach SC overestimates P_L as seen in Fig. 4a. A deep investigation revealed that sampling at the egress switch in our setup becomes biased under high load. Sampling fewer packets lets this bias vanish.

E. Accuracy comparison for bursty traffic and bursty losses

In the previous sections, we evaluated the estimation techniques for *friendly* Poisson traffic. Next, we consider bursty traffic with uniformly distributed burst length $\mathsf{U}_\mathsf{L}[1, 10^3]$ packets and uniformly distributed inter-burst times $\mathsf{U}_\mathsf{t}[1, 10]$ sec. In addition, we model packet losses using a Gilbert-Elliot 2-state Markov discrete time model [20] with slot length of 3 seconds that alternates between a peak loss of $P_{L,max} = 0.5$ and no loss with the following parameters: We parameterize the average loss in this case as $P_{L,avg} = 0.25$ and vary the burstiness parameter T of this Markov chain in the experiments. The parameter T is defined as the average number of slots for this Markov chain to change states twice and can be used to control the burstiness of the losses given fixed peak and average losses. Hence, a higher T denotes slowly alternating long-lived losses while smaller T refers to quickly alternating short-lived losses. We vary the burstiness of the losses to explore how fast the different techniques converge to the ground truth.

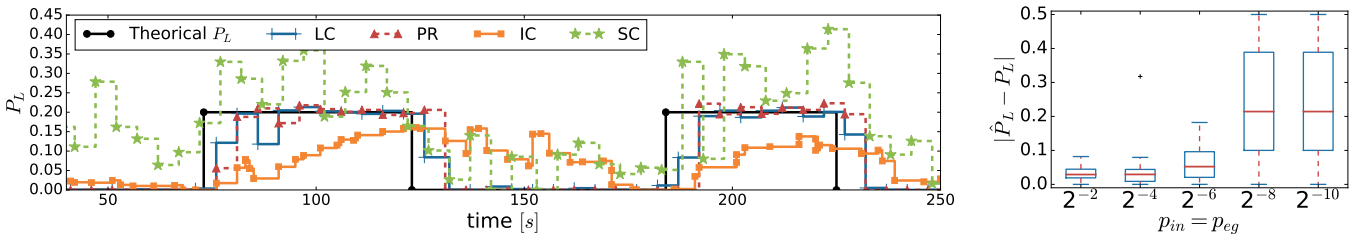
In Figures 4c to 4h we present cumulative distribution functions (CDF) of the estimation error $\hat{P}_L - P_L$ for combinations of Poisson and bursty traffic together with different loss burstiness values T. In Figure 4c we show friendly Poisson traffic and long-lived loss bursts ($T = 50$). Here, legacy counters and active probing (LC, PR) behave similarly while inline counters IC have more outliers due to longer ramp-up times after loss rate changes. The sampling-based counters technique SC shows overestimation issues as discussed in Sect. III-D. Moving to short-lived loss bursts as depicted in Figures 4d and 4e, we see that inline counters IC suffer the most as it is not able to quickly converge to the correct value. For LC and PR the ratio of estimates with an error under 10% reduces from over 95% for long-lived loss bursts ($T = 50$) to 50% for short-lived loss bursts ($T = 5$).

Figures 4f to 4h show the estimation error CDFs for emulated bursty traffic and bursty losses. In general, the accuracy of all techniques degrades for bursty traffic when compared to the friendly Poisson traffic case. With bursty traffic, we can first observe in Figure 4f that active probing PR is slightly superior to legacy counters LC on rather long-lived loss bursts. Note, that moving to short-lived loss bursts in Figures 4g and 4h affects the accuracy of all loss estimation techniques.

F. Discussion

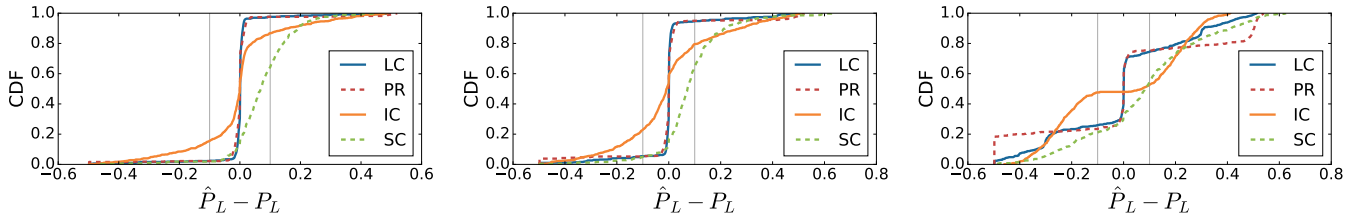
Next, we discuss qualitative advantages and drawbacks of the considered loss estimation techniques as summarized in Table II. Apart from the techniques' accuracy, a comparison, in particular, regarding the overhead and costs is not straightforward. For example, the message overhead produced by legacy and sampled counters (LC and SC) is equal under the same conditions. However, taking the packet counting overhead into account or the advantage of having packet samples at a monitoring entity for further processing tasks, then it can be

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .



(a) Excerpt from an emulation run of the estimated loss over time with loss rate jumps (Gilbert-Elliott 2-State Markov Model; $P_{L,avg} = 0.1$; $P_{L,max} = 0.2$).

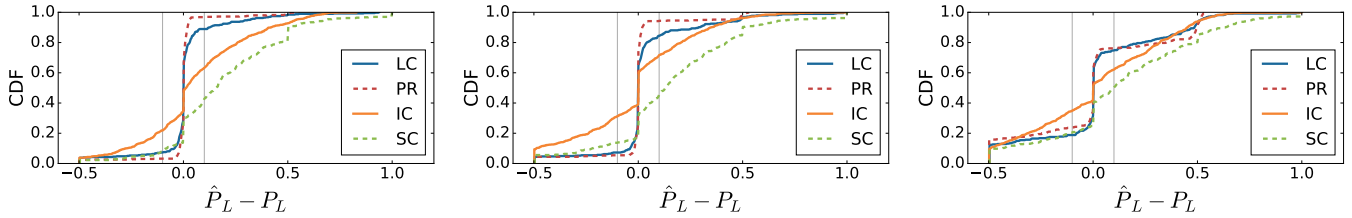
(b) Accuracy of sampling-based counters with changing ingress/egress sampling rates.



(c) Poisson traffic and Gilbert-Elliot loss profile ($T = 50$ long-lived bursts).

(d) Poisson traffic and Gilbert-Elliot loss profile ($T = 25$ medium size bursts).

(e) Poisson traffic and Gilbert-Elliot loss profile ($T = 5$ short-lived bursts).



(f) Bursty traffic and Gilbert-Elliot loss profile ($T = 50$ long-lived bursts).

(g) Bursty traffic and Gilbert-Elliot loss profile ($T = 25$ medium size bursts).

(h) Bursty traffic and Gilbert-Elliot loss profile. ($T = 5$ short-lived bursts)

Fig. 4: Loss rate estimation techniques show contrasting accuracy under different conditions such as variable traffic and loss burstiness. (**LC:** Legacy Counter, **PR:** Active Probing, **IC:** In-line Counter, **SC:** Sampling-based Counter).

argued for either technique. Nevertheless, it is clear that under certain classes of network conditions and requirements some techniques are superior to others. *This is the key motivation to an adaptive selection of the used loss estimation technique.* In the next section, we transform the conducted comparison into a guideline for an adaptive loss rate technique selection.

IV. Towards an Adaptive Selection of Loss Estimation Techniques

Backed up by the analysis in the previous section, we provide, in the following, guidelines for the dynamic selection of the best-fitting loss estimation technique based on the requirements and conditions in the network. Note that we do not consider the accuracy as the sole criterion to determine the best-fitting technique, but also the costs of each technique, if possible. Although we leave a detailed quantitative comparison of the costs of the investigated techniques for future extensions of this work, we introduce the following qualitative cost guideline: *(i)* First, we suggest to use active probing (PR) if and only if none of the other techniques provide proper estimates to avoid notable probe traffic on the control plane and especially on data plane. *(ii)* With respect to costs, the legacy counter (LC) approach is inferior to the sampled counter (SC) approach since all packets must be taken into account in LC. It is also inferior to the in-line counter (IC) as it creates explicit control traffic to fetch counter values. Thus, it is only of choice if IC and SC are not applicable due to conditions known to provide poor accuracy. *(iii)* A qualitative separation of the in-line counter (IC) and sampling counter approach (SC) is not easily possible, however, we argue that the IC approach requires more resources: For example, the number of bytes piggybacked in a control packet could exceed the bytes for explicit statistic requesting and reporting. Furthermore, all packets must be processed by the counter, which is not trivial in high speed networks and forces the devices to use expensive TCAM counter. *(iv)* Thus, we would recommend starting with the sampled counter technique (SC) if the approach delivers reasonable accuracy.

We use this guideline in combination with the performance analysis from Section II to discuss the suitability of different techniques under various conditions given performance and overhead requirements. We consider a network operator that uses some knowledge on the production traffic rates (minimal, low or high traffic amounts and whether it is bursty or not) to select the most suited loss estimation technique. The following recommendations are summarized in Table III.

Minimal traffic: Given negligible amounts of production traffic the passive techniques that count present traffic, may not

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

TABLE II: Qualitative review of the considered loss estimation techniques.

Technique	Review
LC: Legacy Counters	Usable with low production traffic rates and short estimation intervals, however, better with higher packet rates and longer estimation intervals. Acceptable impairment due to loss burstiness and traffic dynamics. *Costs:* Requires available packet counters on forwarding devices. Costs are mainly due to the controller-switch communication: One statistic request and one statistic reply for each counter update.
PR: Active Probing	Good performance independent of the production traffic packet rate or estimation interval length. Highest accuracy on idealized static loss rates and little impairment due to loss burstiness. *Costs:* Probes consume resources on the control plane and mainly on the data plane. Active probing may perturb the network state leading to additional packet losses, e.g., in case of heavy production traffic.
IC: In-line Counters	Good performance throughout various traffic rates but requires smoothing/filtering. Does not depend on a fixed estimation interval, i.e., capturing samples on different time scales. Smoothing, e.g., using a exponential moving average, leads to a trade-off between steady-state accuracy and reaction time to changing loss ratios. *Costs:* Requires available packet counters on the forwarding elements, but only a small number of bytes are piggybacked on control messages.
SC: Sampling-based Counters	Achieves reasonable estimates given high packet rates or long estimation intervals. Saves on resources since only a fraction of all packets are counted and packet samples can be processed to determine other metrics, e.g., traffic matrix. *Costs:* Reduces the packet counting overhead significantly, however, it still possesses the controller-switch communication costs for statistic requests and replies.

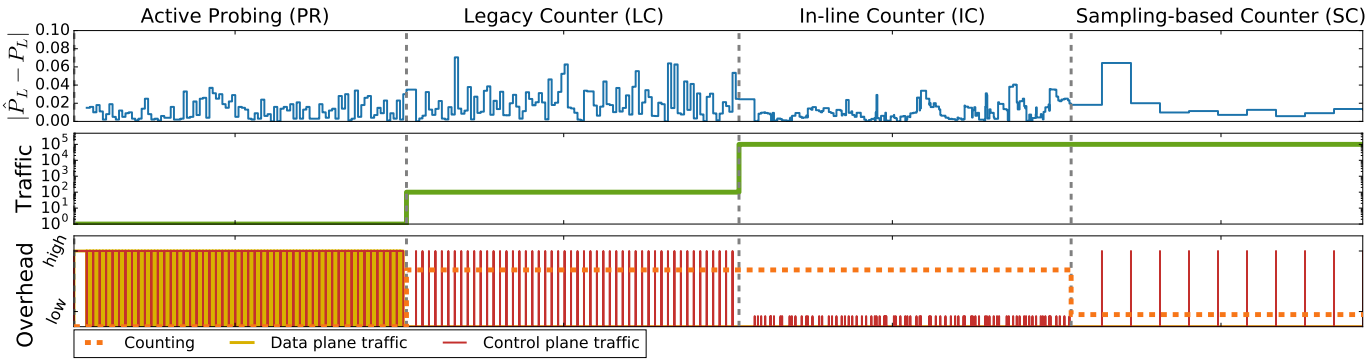


Fig. 5: Dynamic technique selection to estimate P_L with varying traffic conditions and estimation intervals.

TABLE III: Technique selection scheme based on coarse traffic properties and the desired estimation interval.

Traffic Profile	Short Intervals	Long Intervals
Minimal	**PR**	**PR**
Non-bursty, Low	**LC**	**LC**
Non-bursty, High	**IC**$^{\triangle}$	**SC**
Bursty, Low	**LC**$^{\bigcirc}$	**LC**$^{\bigcirc}$
Bursty, High	**IC**$^{\triangle}$	**SC**$^{\triangle}$

$^{\triangle}$**LC** for failure detection $^{\bigcirc}$**PR** if probing traffic acceptable

produce usable results due to sparse data points. For this case, we suggest to use the active PR approach independent of the interval length. The additional probe traffic on the control and data path would obviously not perturb the network operation.

Non-bursty, low traffic load: For this case we showed, e.g. in Figure 2a or 2b, that LC, PR and IC deliver results where 75% of the values have at most a 2% error. However, considering Figures 4c to 4e and Figure 4a, we showed that in-lined counters (IC) are more vulnerable to non-stationary loss jumps such that we favor using (LC). In this scenario of low traffic load, it is justifiable to take the burden of counting all traffic packets as in the legacy counter technique (LC). This decision holds true for short, but especially also for long intervals as the accuracy of LC increases even further.

Non-bursty, high traffic load: With a higher amount of production traffic there are many more packets which can be used to estimate the loss rate, such that less costly techniques provide good results as well. For short estimation intervals, we see in Figures 2c, 2d and 3 that in-line counters (IC) deliver results where 75% of the estimates possess errors below 1,5%. Hence, we prefer IC for short intervals with high traffic rates, however, so far only for deterministic loss rates or non-bursty losses. Given the slow convergence of IC on loss rate jumps (cf. Figure 4a), we propose to use LC as fallback if such jumps are expected or if we are interested in failure detection rather than the stationary loss rate. If long estimation intervals are desired, we recommend SC given its performance in Figure 3.

Bursty, low traffic load: Considering bursty traffic which may alternate between times of no traffic and times of long traffic bursts, however, for an overall low traffic load. Here LC provides high accuracy as depicted in Figures 4f and 4g. If probing traffic is acceptable under this low traffic load, we recommend using PR which delivers continuous estimates instead of depending on sporadic production traffic while

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

retaining the same accuracy as shown in Figures 4f and 4g.

Bursty, high traffic load: Here, we find that less costly techniques IC and LC show acceptable performance in Figure 4f. For long estimation intervals, we recommend the use of SC as a primary technique and LC as fallback alternative whenever larger loss jumps are expected or failure detection is targeted (cf. Figures 4g and 4h). For short estimation intervals, we showed in Figures 4c to 4h (for an estimation interval length of 5 sec) that SC performs poorly and that IC or LC perform much better. Table III summarizes the above recommendations.

Case study of dynamic technique selection for loss estimation:

Next, we describe a crafted scenario in which we emulate changing network conditions and requirements to show the benefit of a loss monitoring system that adapts its estimation techniques. Here, we select the best-fitting technique to maintain a low estimation error while optimizing the costs as discussed previously. We vary the production traffic rates to emulate changing conditions. To emulate a change in the requirements, we change the desired loss estimation interval at the end. This emulates a system that changes between coarse and fine-grained measurements (interval length) depending on whether loss/faults are suspected.

Figure 5 depicts an example of the dynamic selection of the most suitable loss estimation technique over four equally long time frames. In the uppermost subplot, we present the absolute estimation error over time. It can be seen, that in the first two time frames (PR & LC) the system uses a small estimation interval (in this case 2 seconds), while IC in the third time frame is independent of the estimation interval, and in the last time frame the system uses a larger estimation interval of 45s. The middle subplot shows the production traffic rate over time. In the first time frame only negligibly low traffic traverses the network. In the second time frame flows of a low packet rate of 10^2 pkt⁄s are transmitted while in the latter two time frames the traffic rate increases to 10^5 pkt⁄s. The lower subplot shows qualitatively the costs for counting (dashed), control traffic (red spikes), and data plane probe traffic (yellow).

As we find low traffic conditions in the first time slot PR is of choice using its own test traffic. Here, the lowest subplot shows the use of the control plane as well as the data plane probe traffic. In the next time frame production traffic rises, which can be used to measure loss passively. Hence, we use the LC technique. The regular statistic requests and replies between the controller and the switches are sketched in the lower subplot. Here, we also face the overhead of counting packets in the switches. In the third time frame, the traffic rises to a high load such that we can switch over to IC to further save resources on the control plane using only piggybacked counter values. The upper subplot depicts again a good accuracy. Note that the control channel traffic is not deterministic due to the fact that the times at which estimates are taken are also not deterministic. For the last time slot the system switches to a coarse estimation interval of 45 seconds, such that, in combination with the high traffic load, SC produces estimates with good fidelity and low overhead. Using SC we save resources when counting packets (in this case only $1/16$ of all packets are counted on average at the switches) and the number of statistic requests and replies decreases as shown in the lower subplot.

V. Related Work

Communication networks of various sizes ranging from regional networks over Internet service providers to autonomous systems exhibit temporary and persistent packet losses due to many reasons such as failures, misconfigurations and simply heavy traffic [9], [25], [26], [34], [37]. Software-defined networks provide new abilities for fault detection and loss measurement as some monitoring tasks can be delegated to network switches.

Generally, it can be discerned between active and passive measurements for loss estimation. On the one hand, active loss estimation techniques use dedicated probes to infer packet loss [1], [30]. A number of approaches aim to improve active loss measurements through optimizing the probing patterns [4], [22]. They show that within the class of popular ASTA (Arrivals See Time Averages) probing patterns one could still optimize the variance of the estimation error. On the other hand, passive measurement techniques use network switch/router capabilities to infer the loss rate from traffic information [6], [17], [24]. Such passive techniques became also common in SDN where switches provide packet counter information. The authors of [35] propose OpenNetMon which is capable of measuring loss through fetching counters between two endpoints. Besides loss estimation, SDNs enables monitoring other network metrics, in particular, traffic amounts [12], [13], [19], [33] not only using counters but also using controller-offloaded switch-based techniques [31], [36]. Our work here differs from the aforementioned work, such as [31], [35], [36], as we investigate, first, the accuracy and properties of different loss estimation techniques and, secondly, propose to adaptively select the best suited technique depending on the current network conditions and requirements.

Apart from works that consider a single technique to infer loss, there is only little work regarding the trade-offs between different techniques. In [29] the authors provide a quantitative comparison of Poisson and Uniform probing patterns, while showing the benefits of irregular patterns. To the best of our knowledge the closest work to ours is given in [5]. The authors quantitatively compare active probing based loss measurements approaches (ZING and PING) with passively measured router information (SNMP) to estimate loss in wide-area networks. In [5] the authors focus on the correlation between active and passive measurements while in this work we compare the absolute accuracy of different SDN-based techniques under different network conditions.

VI. Conclusion and Future Work

In this work, we discussed how a monitoring service benefits from exchanging different loss estimation or fault detection techniques based on the network and traffic conditions. To achieve this, we first introduced, revised, and evaluated four

Rhaban Hark, Nils Richerzhagen, Björn Richerzhagen, Amr Rizk and Ralf Steinmetz *Towards an Adaptive Selection of Loss Estimation Techniques in Software-defined Networks* In: Proceedings of 16th IFIP Networking 2017 Conference (NETWORKING'17), pp: 1–9, IEEE, June 2016, ISBN 978-3-901882-94-4 © 2017 IFIP .

techniques (three passive and one active technique) that range from sending probes through the network to offloaded switch-based counter sampling techniques. Switch-based techniques leverage forwarding device capabilities such as packet counters or follow more sophisticated approaches such as piggybacked packet counters and sampled counters. Through extensive evaluation, we showed how the accuracy of the various techniques is affected by the network traffic, the estimation interval length, and the loss rate dynamics. Furthermore, we qualitatively assessed the monitoring costs of each technique to find the best-fitting techniques under various network conditions and requirements. We showed in a case study that exchanging estimation techniques retains a high fidelity while keeping the monitoring costs as low as possible.

Future work will consider a detailed quantitative comparison of the monitoring costs. Using a quantitative measure will allow on-the-fly switching between the techniques. Extensions also include the analysis of further offloaded techniques, such as, sketch-based counters or threshold-based counters.

Acknowledgment

This work has been performed in the framework of the CELTIC EUREKA project SENDATE-PLANETS (Project ID C2015/3-1), and it is partly funded by the German BMBF (Project ID 16KIS0471). The authors alone are responsible for the content of the paper. Parts of this work have been conducted within the Collaborative Research Center (CRC) 1053 – MAKI funded by the German Research Foundation (DFG) as part of subprojects B1, B4, C2 and C3.

References

[1] A. Adams, J. Mahdavi, M. Mathis, and V. Paxson, "Creating a Scalable Architecture for Internet Measurement," *IEEE Network*, 1998.

[2] M. T. Arashloo, Y. Koral, M. Greenberg, J. Rexford, and D. Walker, "SNAP: Stateful Network-Wide Abstractions for Packet Processing," in *Proc. of ACM SIGCOMM*, 2016, pp. 29–43.

[3] F. Baccelli and P. Bremaud, *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*, 2nd ed. Berlin: Springer, 2003, vol. 26.

[4] F. Baccelli, S. Machiraju, D. Veitch, and J. C. Bolot, "On Optimal Probing for Delay and Loss Measurement," in *Proc. of ACM IMC*, 2007, pp. 291–302.

[5] P. Barford and J. Sommers, "Comparing Probe- and Router-Based Packet-Loss Measurement," *IEEE Internet Computing*, vol. 8, no. 5, pp. 50–56, 2004.

[6] P. Benko and A. Veres, "A Passive Method for Estimating End-to-end TCP Packet Loss," in *Proc. of IEEE GLOBECOM*, vol. 3, 2002, pp. 2609–2613.

[7] M. Berman, J. S. Chase, L. Landweber, A. Nakao, M. Ott, D. Raychaudhuri, R. Ricci, and I. Seskar, "GENI: A federated testbed for innovative network experiments," *Computer Networks*, vol. 61, pp. 5 – 23, 2014.

[8] G. Bianchi, M. Bonola, A. Capone, and C. Cascone, "OpenState: Programming Platform-independent Stateful Openflow Applications Inside the Switch," *ACM SIGCOMM CCR*, vol. 44, no. 2, pp. 44–51, 2014.

[9] J.-C. Bolot, "End-to-end packet delay and loss behavior in the internet," in *Proc. of ACM SIGCOMM*, 1993, pp. 289–298.

[10] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming Protocol-independent Packet Processors," *ACM SIGCOMM CCR*, vol. 44, no. 3, pp. 87–95, 2014.

[11] Z. Bozakov and A. Rizk, "Taming SDN controllers in heterogeneous hardware environments," in *Proc. of EWSDN*, 2013, pp. 50–55.

[12] Z. Bozakov, A. Rizk, D. Bhat, and M. Zink, "Measurement-based flow characterization in centrally controlled networks," in *Proc. of IEEE INFOCOM*, 2016.

[13] S. R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, "PayLess: A Low Cost Network Monitoring Framework for Software Defined Networks," in *Proc. of IEEE NOMS*, 2014, pp. 1–9.

[14] B. Claise, "RFC 3954: Cisco Systems NetFlow Services Export Version 9," 2004. [Online]. Available: https://tools.ietf.org/html/rfc3954

[15] D. Dietrich, A. Rizk, and P. Papadimitriou, "Multi-provider virtual network embedding with limited information disclosure," *IEEE Transactions on Network and Service Management*, vol. 12, no. 2, pp. 188–201, 2015.

[16] N. Foster, R. Harrison, M. J. Freedman, C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: A network programming language," in *Proc. of ACM SIGPLAN ICFP*, 2011, pp. 279–291.

[17] A. Friedl, S. Ubik, A. Kapravelos, M. Polychronakis, and E. P. Markatos, "Realistic Passive Packet Loss Measurement for High-Speed Networks," in *Proc. of TMA*. Springer Berlin Heidelberg, 2009, pp. 1–7.

[18] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: Towards an Operating System for Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 3, pp. 105–110, Jul. 2008.

[19] R. Hark, D. Stingl, N. Richerzhagen, K. Nahrstedt, and R. Steinmetz, "DistTM: Collaborative Traffic Matrix Estimation in Distributed SDN Control Planes," in *Proc. of IFIP Networking*, 2016, pp. 82–90.

[20] G. Hasslinger and O. Hohlfeld, "The Gilbert-Elliott Model for Packet Loss in Real Time Services on the Internet," in *Proc. of GI/ITG Conference*, 2008, pp. 1–15.

[21] L. Jose, M. Yu, and J. Rexford, "Online Measurement of Large Traffic Aggregates on Commodity Switches," in *Proc. of USENIX Hot-ICE*, 2011, pp. 13–13.

[22] S. Machiraju, D. Veitch, F. Baccelli, and J. C. Bolot, "Adding Definition to Active Probing," *ACM SIGCOMM CCR*, vol. 37, no. 2, pp. 17–28, 2007.

[23] Open Networking Fundation, "Software-Defined Networking: The New Norm for Networks," *ONF White Paper*, 2012.

[24] A. Papadogiannakis, A. Kapravelos, M. Polychronakis, E. P. Markatos, and A. Ciuffoletti, "Passive end-to-end packet loss estimation for grid traffic monitoring," in *Proc. of CoreGRID Integration Workshop*, 2006, pp. 79–93.

[25] K. Papagiannaki, R. Cruz, and C. Diot, "Network Performance Monitoring at Small Time Scales," in *Proc. of ACM IMC*, 2003, pp. 295–300.

[26] V. Paxson, "End-to-end Internet Packet Dynamics," *ACM SIGCOMM CCR*, vol. 27, no. 4, pp. 139–152, 1997.

[27] B. Pfaff, J. Pettit, T. Koponen, E. J. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon, and M. Casado, "The Design and Implementation of Open vSwitch," in *Proc. of USENIX NSDI*, 2015, pp. 117–130.

[28] P. Phaal and M. Lavine, "sFlow Version 5," 2004. [Online]. Available: http://sflow.org/sflow_version_5.txt

[29] M. Roughan, "A Comparison of Poisson and Uniform Sampling for Active Measurements," *IEEE JSAC*, vol. 24, no. 12, pp. 2299–2312, 2006.

[30] S. Savage, "Sting: A TCP-based Network Measurement Tool," in *Proc. of USENIX USITS*, vol. 2, 1999, pp. 7–7.

[31] S. Shirali-Shahreza and Y. Ganjali, "FleXam: Flexible Sampling Extension for Monitoring and Security Applications in Openflow," in *Proc. of ACM HotSDN*, 2013, pp. 167–168.

[32] A. Sivaraman, A. Cheung, M. Budiu, C. Kim, M. Alizadeh, H. Balakrishnan, G. Varghese, N. McKeown, and S. Licking, "Packet Transactions: High-Level Programming for Line-Rate Switches," in *Proc. of ACM SIGCOMM*, 2016, pp. 15–28.

[33] A. Tootoonchian, M. Ghobadi, and Y. Ganjali, *OpenTM: Traffic Matrix Estimator for OpenFlow Networks*. Springer Berlin Heidelberg, 2010, pp. 201–210.

[34] D. Towsley, M. Yajnik, S. Moon, and J. Kurose, "Measurement and modeling of the temporal dependence in packet loss," in *Proc. of IEEE INFOCOM*, 1999.

[35] N. L. M. van Adrichem, C. Doerr, and F. A. Kuipers, "OpenNetMon: Network monitoring in OpenFlow Software-Defined Networks," in *Proc. of IEEE NOMS*, 2014, pp. 1–8.

[36] M. Yu, L. Jose, and R. Miao, "Software Defined Traffic Measurement with OpenSketch," in *Proc. of USENIX NSDI*, 2013, pp. 29–42.

[37] Y. Zhang and N. Duffield, "On the Constancy of Internet Path Properties," in *Proc. of ACM SIGCOMM IMW*, 2001, pp. 197–211.