Packet loss estimation using distributed adaptive sampling

René Serral-Gracià, Albert Cabellos-Aparicio, Jordi Domingo-Pascual
Advanced Broadband Communications Centre, Technical University of Catalunya (UPC), Spain
{rserral, acabello, jordid}@ac.upc.edu

***Abstract*—Packet losses are a critical metric for network performance assessment. In this paper we present a novel methodology to accurately estimate the packet loss ratio in real-time in a fully distributed scenario.**

The constraint such systems must face is the large amount of resources required for keeping live performance assessment.

Our contribution has two main parts. On the one hand we study the behaviour of packet losses among the traffic sharing a path, and extend the classical definition of *loss burst* by the concept of density. On the other hand, with the knowledge acquired in the loss distribution study, we present an adaptive sampling technique that schedules the network resources in order to distributely estimate the packet losses with reasonable accuracy.

In order to validate the proposal we perform some real tests over an European-wide testbed. The results show a great improvement in packet loss estimation over previous research, while using a controlled amount of resources.

I. Introduction

Companies and service providers are slowly realizing that the provisioning of multimedia contents on the network imply to provide some kind of Service Level Agreement (SLA) to their customers. This inevitably leads to propose means for verifying such SLA. There are some recent research efforts in this area [1]–[4], all using different approaches, but all sharing the same idea: to estimate the end-to-end metrics, namely One Way Delays (OWD), Packet Loss Ratio (PLR) and Inter Packet Delay Variation (IPDV).

This paper focuses on the estimation of the Packet Loss Ratio. This metric can be estimated by either using active or passive traffic measurements. Active measurements for SLA verification such as [1] can be used when the access to the network under study is restricted. However, if access is granted, passive measurements provide better accuracy [5].

Related to passive approaches several on-line traffic monitoring infrastructures for network performance assessment have been published [3], [4], [6]. These systems usually focus on direct traffic observations for assessing the network performance, and they use a distributed infrastructure in order to gather the network metrics. This infrastructure uses at least two collection points where packets are time-stamped, sent to a central processing unit that performs packet matching, and finally the performance metrics are extracted. This means that these distributed infrastructures require additional control traffic to maintain the live reporting.

Gathering this information in a per packet basis is very expensive in terms of bandwidth and processing resources needed by the control traffic. In our previous work we proposed a full distributed infrastructure named Network Parameter Acquisition System (NPAS) that uses mechanisms of traffic aggregation [3] and static sampling techniques [7] in order to reduce the resource consumption. In addition we presented in [8] an adaptive sampling technique to further reduce the used resources. With adaptive sampling the sampling rate is dynamically adjusted to a given value that minimises the used resources providing reasonable accuracy. The problem highlighted in these works [8] is that estimating the PLR is not straightforward by means of traffic sampling. With traffic sampling the accuracy is tied to the number of samples, hence it is reduced due to low rate flows or when using low sampling rates. Despite of this, we believe that improving PLR estimation is a critical issue, given that for all metrics packet losses have the most noticeable effect on the network performance, especially when dealing with real-time traffic and user perception [9]. In this paper we present an adaptive sampling technique to estimate the PLR, our technique is intended to be used in any passive distributed infrastructure such as NPAS [3] or perfSONAR [4].

The main issue addressed on this paper is: how can we adapt the sampling rate to minimise the used resources while maintaining a reasonable accuracy? The sampling rate should be higher during congestion periods and lower when there is no congestion (i.e PLR close to zero). An obvious option may be to increase the sampling rate when a packet loss is detected and decrease it otherwise. The main issue is that (as we will show later empirically) some packet losses are sporadic while others belong to bursts. Even more, during congestion periods not all the packets are lost. This means that our technique should not overreact increasing the sampling rate when a sporadic loss occurs, instead of this it should increase the sampling rate when a bursts of losses are detected. On the contrary, our technique should not underreact during a congestion period when a packet is not lost. Therefore it should decrease the sampling rate only when the congestion period has finished.

In order to solve this issue we broaden the classical definition of burst [10]. Instead of considering a burst as the list of consecutive packet losses, we introduce the concept of density

This work was partially funded by IST under contract 6FP-004503 (*IST-EuQoS*) and NoE-038423 (*Content*), MCyT (Spanish Ministry of Science and Technology) under contract TSI 2005-07520-C03-02 and the CIRIT (Catalan Research Council) under contract 2005 SGR 00481.

978-1-4244-2067-4/08/$25.00 ©2008 IEEE.

to enhance the burst definition (our definition is similar to that of [11]) and we use it for not over or underreact in a real scenario. In addition, we also show that during congestion the packets are lost uniformly among all the flows over a path. We use the knowledge acquired in these experiments for designing an adaptive sampling technique which permits to improve the PLR estimation with bounded resource consumption.

The evaluation of the system is performed by using the European-wide testbed provided by the EuQoS project [12]. Our results show an improvement of 25% in average in the PLR estimation, compared with the results obtained by using static sampling. Moreover, the proposed solution bounds the resource utilisation regardless the number of active flows, while static sampling increases the amount of resources proportionally to the underlying traffic.

The rest of the paper is structured as follows, in the next section we present the work related to our proposal, by focusing in packet loss and SLA validation infrastructures in distributed scenarios. Later in Section III we present the Network Parameter Acquisition System (NPAS) as the base of our work. After we study the behaviour and the distribution of packet losses in a real scenario. This together with NPAS is the starting point over which we designed the dynamic adaptive sampling methodology, which is detailed in Section V. Section VI presents the evaluation of the methodology. In Section VII we discuss about the deployment of the whole system, and finally in Section VIII we draw some conclusions and point the open lines for further study.

II. Related Work

Packet losses have been subject of study in many different environments. Often in the area of QoS, which evolved with works such as [13] into a new research topic, network measurements. In this area a formal metric for packet losses was defined in RFC-2680 and RFC-3357. From there, many efforts have been invested in modelling the packet loss behaviour [10], [14], [15] and estimate them by using active measurements tools, starting from `ping` to `zing` [16], continuing with `sting` [17] and recently with `badabing` [11].

As we have mentioned earlier, our work focus on passive estimating the PLR for SLA verification through adaptive sampling. Recently some work can be found centered on this SLA assessment in [1] and [18]. In [1] the authors propose a way of accurately estimating the packet losses by means of active traffic generation, this largely differs from our approach, which is based on a distributed passive Network Parameter Acquisition System (NPAS). While in [18] the author compares the accuracy achievable in counting the SLA violators and does not consider a full fledged reporting infrastructure.

As generic performance assessment infrastructures, at the best of our knowledge very few tools have been published. Firstly, perfSONAR [4] is a tool intended to monitor any network metric on a distributed fashion. Specifically the authors present a methodology that provides meaningful network performance indicators, which can be used to visually monitor the status of the traffic. The main difference between this tool and our proposal is that while perfSONAR limits the study to the link status by active traffic generation, NPAS analyses directly the network metrics by passively collecting the traffic. Secondly InterMON [6] is based on interaction and coordination of different kinds of tools with access to a common database. The system's main goal is to gather network metrics for off-line processing, analysis and data-mining. In this case, the main difference between InterMON and our proposal is that we are focused on providing live values for some network metrics not just gathering information for off-line processing.

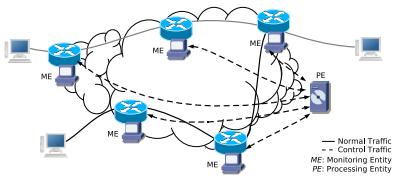

Fig. 1. Example Scenario

III. Description of Network Parameter Acquisition System

Network Parameter Acquisition System as proposed in [3] is a distributed network performance assessment infrastructure. It computes the network metrics suitable to quantify the network performance over sensible traffic (e.g. VoIP or Videoconferencing) on-line.

The system collects the traffic on each ingress and egress points in the network, analysing only the traffic selected by the configured filters. It computes the following network metrics: One-Way Delay (OWD), Packet Loss Ratio (PLR), Inter Packet Delay Variation (IPDV) and used bandwidth.

A. Basic system

This infrastructure presents a framework which reports the intra-domain traffic metrics to higher layer entities to assess whether the specified performance constraints (policies) are fulfilled or not. This mechanism can also be used for triggering alarms to give feedback to other system units in case that the network is not providing the contracted quality. Both alarms and policies management are out of the scope of this specification.

Figure 1 shows a generic network scenario where the system is deployed. It is composed by the following entities:

1) The Monitoring Entity (ME): is in charge of the traffic collection via selection filters. The traffic selection policy aggregates the data in a per flow or per Class of Service (CoS) basis, but it can be easily extended to others.

The traffic information collected by MEs is sent to a per-domain analyser entity *(Processing Entity - PE)* which computes the network metrics of all the traffic under analysis.

2) The Processing Entity (PE): is the gathering point within the domain. It configures the ME policies, performs most of the processing, matches the information of the packets coming from the MEs and computes the network metrics on-line. The

results are published as a network management service to monitor the status of the traffic under analysis in real-time.

Depending on the information sent by the ME more control traffic is required to perform the monitoring. In general this can be a problem for under provisioned domains, or ones without dedicated links for network management traffic. We believe that this is the case in most ISPs. That's why NPAS controls this issue by having three different modes of operation: *i)* per packet reporting, where the traffic information is sent in a per packet basis. *ii)* using time aggregation, where a time sliding window is used to optimise the network usage. *iii)* by using traffic sampling, as further discussed in the next section.

More detail in the specification of the first two modes and the analysis of the bandwidth usage can be found at [3]. In this paper we enhance the static sampling mode of operation (described in the following subsection) by adaptively adjusting the sampling rate wherever is more optimal.

B. Static distributed sampling

The complexity of applying sampling in this distributed scenario is that centralised techniques (e.g. the techniques studied in [19]) are not well suited for this task, since they do not guarantee that all the ME capture exactly the same set of packets. Thus, accurately determining packet losses or one way delays is not feasible. We addressed this issue in [7] by using a deterministic sampling technique (hash sampling [20]) to match exactly the same packets all over the different ME for later computing the network metrics. This permits to all the different ME to collect traffic independently, and only the selection function at start up time has to be shared by the MEs.

This distributed sampling technique computes a hash function over a set of fields on the packet's header, and depending on its value the packet will be considered for later analysis. In our environment using only one hash table is not sufficient, since the monitoring framework has to guarantee that all the flows under analysis are monitored. So, the sampling must be applied within individual flows, not directly to all the collected traffic. Hence, the packets are identified by using two different keys, *flow ID* and *packet ID* with the corresponding two level hash table. Thus, a packet is only analysed if it falls within specified positions in the hash table (with size A), of which only the first r packets are considered. Changing r gives different sampling rates $(\frac{r}{A})$. The hash table is sent from each ME to the PE when A packets have arrived or after a timeout t. This time interval t is known as a bin and it limits the upper bound for the live reporting interval. This permits to efficiently control the resources and the applied sampling rate.

The challenge one must face when using distributed sampling for live reporting is that estimating the packet loss ratio is difficult. In such environments it is well known [21] that the achieved accuracy (within a 95% confidence interval) when classifying sampled traffic into categories is bounded by Equation 1,

$$\varepsilon \leq 1.96\sqrt{\frac{1}{c}} \tag{1}$$

where c is the amount of sampled packets within the category (packet losses in our case). The above equation assumes normality in the distribution of the samples, but it is feasible to use it if we assume randomness in the sampling process.

As a matter of fact, with the sampling methodology presented previously, the selected packets are unpredictable in advance since the selection depends on packet contents, hence the process complies with the restriction, as guarantied with hash sampling [20].

Although, the only way of improving the accuracy is by increasing c, which in its turn means increasing the sampling rate ρ or by having a bigger bin size t. The problem with increasing t is that delaying the reporting gives less responsiveness to the system. Hence in this paper we only consider adjusting the sampling rate, leaving as an important part of our future work the study of the bin size and its implications on the results.

IV. Packet Loss Analysis

Before designing a mechanism to improve PLR estimation, first we must understand how packet losses behave in a network.

Packet losses are an important metric when assessing network performance. Most of the previous work [1], [10], [14] proof that packet losses usually have a bursty nature, with a non negligible amount of sporadic packet losses.

Bursty losses degrade the quality to a higher degree than sporadic packet losses, since application's correcting algorithms do not work in such situations [9]. In [10] and in RFC-3357 the authors define a loss burst as the sequence of consecutive lost packets. With this strict definition it is possible to detect lossy periods, but in a congested environment many of those loss periods can be chained together, with few successfully transmitted packets, forming a longer time period with poor network conditions. It is advisable to take into account such periods for improving the PLR estimation in our algorithm. Hence we propose to generalise the concept of bursty losses by using density thresholds as explained in this section.

A. Definitions

In a congested link, when using a common drop tail queue, the last packets arriving a node are discarded. But while the queue is emptying there will be room for accepting new packets, at least until the queue is full again. In this situation there would be several packet loss bursts with few successfully sent packets in-between. This situation is difficult to detect with the classical burst definition because the loss bursts are much shorter than the congestion period. For NPAS this situation must be handled in order to react in time and improve the packet loss estimation in such cases.

Therefore, in this work we extend the classical burst definition: A packet loss burst is the interval of time with a density of packet losses higher than a threshold τ.

Formally, given a stream of numbered packets $\mathcal{P} = \{p_1, \ldots, p_n\}$ belonging to a flow f with sending timestamps

$\mathcal{T}_{tx} = \{t_{1_{tx}}, \ldots, t_{n_{tx}}\}$, a packet p_i is considered as lost if it does not reach its destination with a predefined time T; that is when $T \leq t_{i_{rx}} - t_{i_{tx}}$. Then $\ell_i = 1$ if packet p_i is lost and 0 otherwise. Thus the flow stats regarding packet losses can be defined as $\mathcal{L} = \{\ell_1, \ldots, \ell_n\}$.

Using the above nomenclature we propose the following definitions:

Definition 1: Distance between a pair of packets p_i, p_j $d_{(i,j)}$, is the amount of transmitted packets between p_i and p_j. Where $i < j$, hence $d_{(i,j)} = j - i$.

Definition 2: The density $\Lambda_{(i,j)}$ of packet losses between packet p_i and packet p_j is $\Lambda_{(i,j)} = \frac{\sum_{k=i}^{j} \ell_k}{d_{(i,j)}}$.

Definition 3: A burst of packets β starting at packet i then is defined by the tuple:

$\beta = \langle d_{(i,j)}, \Lambda_{(i,j)} \rangle$. Where $\forall k \mid i < k \leq j, \Lambda_{(i,j)} \geq \tau$ and $\nexists\ z \mid d_{(i,z)} > d_{(i,j)}$ with $j < z \leq n$ and $\Lambda_{(i,z)} \geq \tau$.

Hence, a burst is composed by the longest list of packets starting at i where the loss density between i and j is higher than τ.

Definition 4: A burst $\mathcal{B}$ for a flow (f) is defined as $\mathcal{B}_f = \{\beta_0, \ldots, \beta_m\}$ as the list of bursts in a flow.

With the constraint that any β_i cannot overlap with any β_j for any $j \neq i$.

Using the above definitions, we aim to study the behaviour of packet losses in a real scenario. Moreover we analyse the effects of packet losses in concurrent flows with the same source and destination. This study will help later in our adaptive sampling solution to estimate the packet loss ratio.

B. Testbed

In order to study the packet loss behaviour we performed a set of more than 500 experimental tests during 2006 and 2007 using twelve different testbeds across Europe. The testbeds were provided by EuQoS [12] partners, covering a total of 5 countries and 4 different access technologies (LAN, xDSL, UMTS and WiFi) with an overlay architecture over the Géant research network [22].

The packet losses were studied by actively generating UDP traffic on the network with different properties. Specifically we generated periodic flows, with varying packet rates, from 16 to 900 packets per second among all the involved nodes in the testbed. We used different packet sizes ranging from 80 to 1500 bytes per packet.

With this broad range of tests we think that our results are representative of typical network behaviour regarding packet losses. Even if it could be improved by using real traffic in our analysis, which we leave as an important part of our future work.

C. Burst study

Since many tests do not have any packet loss, we removed them from the analysis, keeping the 22% of tests with some loss. In some of the performed tests we intentionally generated more traffic than the available bandwidth, in order to obtain some congested periods for our analysis.

	Burst	Sporadic
Bursts in lossy periods	43.3%	56.7%
Packets in burst	99.2%	0.8%

TABLE I
DETAILS OF BURST IN THE EXPERIMENTAL ANALYSIS

In the tests we identified all the bursts with *Definition 4* and we used $\tau = 0.45$. In this section we do not aim to evaluate the effects of changing τ, we will focus on this in Section VI.

As expected from the tests most of the losses belong to a burst (around 99%), with an average distance of 182 packets, which indicates an average burst duration of $\sim 180ms$, but with a large standard deviation (around $1100ms$). The distance's 99^{th} percentile is around 6 seconds, this percentile is that high because of the induced congestion described above.

Considering the total amount of lossy periods[1] in all the tests, just $\sim 43\%$ were the beginning of a burst, while the other $\sim 57\%$ were only sporadic losses. This is detailed in Table I. As it can be noted the periods with sporadic losses are higher than those for bursts. Later we will use this property in order to not overreact in the presence of sporadic losses.

But the most surprising outcome of this analysis is that counting the total amount of packet losses as a whole these $\sim 43\%$ in reality represent the $\sim 99\%$ of the total lost packets. Meaning that once in a burst, the probability of having lots of losses is very high.

D. Loss behaviour among flows

As we discussed before in Equation 1 the more packets we collect the better is our PLR estimation, but sometimes we can be analysing low rate flows where after applying the sampling rate maybe just one or two packets are considered per bin. In this situation we can fail to estimate the PLR very easily.

In the following study we aim at proving experimentally that packet losses spread regularly among all the flows in the same congested path.

It is well known from queueing theory that the probability of queueing of a packet depends only on the traffic load in the network [23] and it is independent of the size of the packet or cross-traffic packets. This means that the probability of losing a packet does not depend on the flow to which it belongs and thus, during congestion packets will be lost among all the flows randomly. In this subsection we will analyse this theory empirically.

Then we can use this reasoning to infer that when we detect packet losses on one flow we should increase the sampling rate of all the flows of the path since most probably they are also experiencing losses.

To prove this regular spreading we selected three different partners from the testbed presented before. They were located in Warsaw University of Technology (WuT), University of Bern (UoB) and Technical University of Catalonia (UPC), and generated traffic with several flows simultaneously, from UPC

[1] A lossy period starts at the first packet lost after a period of no losses and ends when the loss burst is finished (or immediately in the case of sporadic losses).

(a) 10 flows

	Min.	Max.
Path 1	$8 \cdot 10^{-4}$	$9 \cdot 10^{-4}$
Path 2	$1.5 \cdot 10^{-3}$	$1.7 \cdot 10^{-3}$
Path 3	$1.7 \cdot 10^{-3}$	$4 \cdot 10^{-3}$

(b) 600 flows

	Min.	Max.
Path 1	$13 \cdot 10^{-2}$	$17 \cdot 10^{-2}$
Path 2	$12 \cdot 10^{-2}$	$18 \cdot 10^{-2}$
Path 3	$2 \cdot 10^{-4}$	$5 \cdot 10^{-4}$

TABLE II
MIN. MAX. PLR AMONG THE FLOWS

to UoB, from UoB to WuT and from WuT to UPC. Some tests were performed with groups of 10 flows, for others we used 600 flows, each group of flows were sent from the same source to the same destination during a 5 minutes period. The groups of 10 flows had a packet size of 315 bytes with a rate of 200 packets per second. While the groups of 600 flows used a packet rate of 20 pkt/sec and 60 bytes each. The tests were performed at different hours with different cross traffic conditions produced by the Géant network.

The comparison is performed by computing the PLR of each flows and compare the minimum and maximum PLR as detailed in Table II. As it can be noted the PLR bounds even if not exact on each flow within each group are close, with a bit more of variation for the 600 flows as expected due to the bigger amount of flows.

From the results obtained in this section we conclude that when we detect high density of packet losses in one or more flows within a path, the probability that all the flows in the path exprerience similar conditions is very high. Hence it is worth increasing the sampling rate of *all* the flows sharing that path. This is an important property to exploit in order to improve the PLR estimation accuracy.

V. DISTRIBUTED ADAPTIVE SAMPLING

Using traffic sampling is usually a good way of controlling the required resources to perform a task. But choosing the optimal sampling rate with minimum loss of accuracy is not straightforward.

Moreover, knowing that control traffic (the reported information from MEs to PEs) can be limited on the network, might force the per flow sampling rate to be further reduced. Hence, selecting the right flows to reduce the sampling rate leads to better results than just equally sharing the sampling rate among the flows.

The methodology presented here permits to efficiently estimate the PLR by using the knowledge acquired in Section IV. In this paper we do not consider OWD, since its estimation requires less resources to obtain accurate estimates as we pointed out in [8].

A. Problem Formulation

Given a domain $\mathcal{D}$ with $\mathcal{R} = \{r_1, \ldots, r_n\}$ ingress and egress points of the network, and $\mathcal{F}_\mathcal{D} = \{f_1, \ldots, f_m\}$ the list of active flows within the domain. Then $R_{f_i}(t)$ (R_i from now on) is the rate of the i^{th} flow at time t, being n the number of ME and m the number of flows.

For simplicity we consider, without loss of generality, that the required resources for a ME express the bandwidth requirements to send live information to the PE. Then the resources required to monitor all the existing flows from ME i to ME j are:

$$x'_{ij} = \mathcal{S} \sum_{k=1}^{m} \delta_{ijk} R_k \tag{2}$$

where $\mathcal{S}$ is a constant defining the amount of resources needed to send a single control packet, for example the number of bytes. Since it is a constant, for the ease of exposition, we will assume $\mathcal{S} = 1$. While $\delta_{ijk} = 1$ if f_k goes from r_i to r_j and 0 otherwise. From this we infer that the resources required for each ME (X'_i) are: $X'_i = \sum_{j=1}^{n} \left(x'_{ij} + x'_{ji} \right)$. Hence, reporting in a per packet basis implies that the total amount of resources required for on-line monitoring in domain $\mathcal{D}$ is $X_\mathcal{D} = \sum_{i=1}^{n} X'_i$.

In general per packet reporting requires too many resources to be feasible. We ease this requirement by applying adaptive traffic sampling. We consider that X are the global resources available to perform the collection and performance assessment. We need a fair share of those resources among all the ME, considering that the ME with more load (e.g. with more number of monitored flows) deserve more of the resources. Hence, we need to adapt the per flow sampling rate (ρ) in order to comply with this restriction:

$$X_\mathcal{D} \geq X \geq \mathcal{S} \sum_{i=1}^{m} \rho_i R_i \tag{3}$$

where ρ_i is the sampling rate ($0 \leq \rho_i \leq 1$) for flow i, R_i is the rate in packets per second and X is the maximum resources reserved for the monitoring. Note that R_i (in packets/sec) is known since the ME initially must collect all the traffic to decide whether it falls within the specified threshold indicated in the hash table or not as described in III-B.

We need a lower bound of the sampling rate, namely $\rho_{i_{min}}$, which guaranties that it is adjusted depending on the rate in a per flow basis, since low rate flows at small time scales are very sensible to sampling rates. This lower bound determines the minimum required resources for the whole monitoring task. Thus,

$$X_{min} = \sum_{i=1}^{m} \rho_{i_{min}} R_i \tag{4}$$

Being $\rho_{i_{min}}$ the minimum ρ_i for flow i. This works well if the traffic has constant rates, but since traffic in general is variable these values need to be updated periodically.

The minimum applicable sampling rate has to take into account the rate of the flows, equation 5 limits such rate,

$$\rho_{i_{min}} = \frac{1}{R_i} \quad \forall i \setminus 1 \leq i \leq m \tag{5}$$

where $\rho_{i_{min}}$ is the minimum applicable sampling rate for all f_i. This guaranties at least that one packet per flow is captured.

If c_{min} is the number of samples per f_i then,

$$\begin{aligned} \rho_{i_{min}} &= 1 \quad \forall i \setminus c_{min} \geq R_i \\ \rho_{i_{min}} &= \frac{c_{min}}{R_i} \quad ,otherwise \end{aligned} \tag{6}$$

when more precision is required or more resources are available, assuring that at least c_{min} packets per flow are considered.

In the same way we can define the maximum sampling rate to

$$\begin{aligned} \rho_{i_{max}} &= 1 \quad \forall i \setminus c_{max} \geq R_i \\ \rho_{i_{max}} &= \frac{c_{max}}{R_i} \quad ,otherwise \end{aligned} \tag{7}$$

where c_{max} determines the maximum number of per flow packets to be collected. Then $X_{max} = \sum_{i=1}^{m} \rho_{i_{max}} R_i$ and $X_{max} \leq X \leq X_{\mathcal{D}}$. Analogously to $x_{i_{min}}$, $x_{i_{max}}$ refers to the maximum resources for a particular flow.

B. Resource Sharing

With equations 3 and 4 we know that after X_{min} are used the available resources (ΔX) in order to share among the flows are $\Delta X = X - X_{min}$.

The goal now is to fairly share the ΔX resources among all the flows. Let's define ω_i as the weight assigned to the flow i where $\sum_{i=1}^{m} \omega_i = 1$, then the resources assigned to the flow follow equation 8.

$$x_i = \min\{\lfloor \Delta X \omega_i \rfloor + x_{i_{min}}, x_{i_{max}}\} \tag{8}$$

It can be noted that the resources assigned to the flow are bounded by the X_{max} as assigned previously. Now the weights (ω) may be assigned in different ways, we propose an approach, namely *Path Driven*, whose goal is to improve the PLR estimation without exceeding the resources reserved for the reporting process.

Path driven PLR estimation is based on the search for lossy paths and adjust the sampling rate of all flows within the path accordingly to its PLR.

The rationale behind this methodology becomes from the fact shown in Section IV regarding the distribution of losses among all the flows on a path. Once a burst is detected on one flow between two ME, we must increase the resources used on estimating the PLR on that path. This way we increase the amount of samples, and thus the accuracy.

We define three different states for paths: *i)* Normal path, *ii)* Path with sporadic losses, and *iii)* Congested path.

A path is normal when $PLR \leq \gamma$[2]. Sporadic losses is when $PLR > \gamma$ and $\Lambda_{(i,j)} \leq \tau$. Where i and j are the first and the last packets on the bin. Finally if $\Lambda_{(i,j)} > \tau$ and $PLR > \gamma$ then the path is considered as congested.

This avoids overreacting if losses are sporadic (i.e. the sampling rate is increased for the whole path while having sporadic losses), and also underreacting in case of congestion.

[2]In QoS environments $\gamma \leq 10^{-3}$ usually (See Rec. ITU-T Y-1541 for more detail).

Algorithm 1 Pseudo-code for weight assignment

Input: $ME_{pair}[1..n]$, F_{list} {ME Pairs and flow list}
for all me $\leftarrow ME_{pair}$ **do**
  $plr \leftarrow$ computeMEPLR(me, F_{list}) $\{plr = n * max(me_{plr})$ in case of congestion$\}$
  if $plr \neq 0$ **then**
5: density = $computeDensity(me, F_{list})$
    $D \leftarrow D \cup \langle me, density, plr \rangle$
    $totalPLR \mathrel{+}= plr$
  end if
end for
10: **for all** $d \leftarrow D$ **do**
  if $d[density] > \tau$ *and* $d[plr] > \gamma$ **then** {We have a congested path}
    $\Omega \leftarrow \frac{d[me]_{PLR}}{totalPLR}$ {Ω is the total weight for the path}
    for all flow $\leftarrow$ allFlowsWithLosses$(F_{list}, d[me])$ **do**
      $F[flow]_{\omega_i} \leftarrow \Omega \frac{1}{numberFlows(d[me])}$
15: **end for**
  else if $d[plr] > \gamma$ **then** {We have sporadic losses}
    for all flow $\leftarrow$ allFlows$(F_{list}, d[me])$ **do**
      $F[flow]_{\omega_i} \leftarrow \frac{F[flow]_{PLR}}{totalPLR}$
    end for
20: **end if**
end for
Output: F_{list} with all the ω_i initialised

Here the most critical parameter to avoid under or overreacting is τ as we will show later.

The detailed process for weight assignment is shown in Algorithm 1. The algorithm's input is the list of ME pairs with the flow list. This list has the flow properties together with each flow's PLR.

For each ME pair we compute the total PLR in the ME $(computeMEPLR)$ as follows:

- If any flow has a $\Lambda_{(i,j)} > \tau$ then all the flows set its PLR to the maximum within the flow, hence the aggregated PLR will be $n * max(PLR)$ with n the number of flows between the ME pair. Here we use the property of uniform spreading of the PLR within a path.
- If the flows just have sporadic losses (i.e $\Lambda_{(i,j)} \leq \tau$) the returned PLR will be $\sum_{i=1}^{n} \ell_i * f_i[plr]$.

From line 10 to the end of the algorithm we share fairly the weights ω proportionally to the *PLR* among all the flows on the congested path. While we only give the proportional share to the flows with sporadic losses, not to the whole path. Hence optimising the resource utilisation where is more needed.

VI. Evaluation

This section is devoted to the evaluation and validation of the proposal, we describe the used methodology used for validating the distributed adaptive sampling solution. This validation is performed by using the testbed presented in Section IV to show the accuracy of our solution in a real environment.

Moreover, we complete our validation showing the effects on the accuracy obtained by tweaking the density threshold (τ).

A. Experimental evaluation

The tests were performed by actively generating synthetic periodic traffic as described before. In the validation we study the accuracy in the estimation of PLR and we also compare it with the one obtained by the distributed static sampling.

1) Methodology: To ease the test management, the tests were performed independently one from the others, at different hours, and we collected the full trace of the traffic for later processing. With the obtained traces we emulated a system with the flows entering randomly to the system. We also combined the tests with the 600 simultaneous flows used in Section IV.

We defined the maximum number of active flows to 100 plus the 600 parallel flows which were active during all the experiment. Besides, new flows were entering the system using an uniform random distribution with random duration. With this set up we applied off-line the distributed adaptive sampling. Hence we were able to evaluate the results for different resource reservations (X) by taking as reference the complete original trace. We chose for the experiments the following X: $20000, 15000, 10000$ and 5000 with $\mathcal{S} = 1$. These resources are treated as a global property of the tests, and given the total generated traffic among the MEs these X correspond to the following effective sampling rates (ρ): $26\%, 25\%, 22\%$ and 16% respectively.

Another important parameter to the system is the reporting interval (bin), where we assume that a good upper bound is $t = 175ms$ because it falls within typical QoS constraints (as stated Rec. ITU-T 1540) without flooding the network with reporting traffic. We used this time interval with success previously in [7] and [8].

2) Accuracy of the solution: In order to quantify the accuracy obtained by our solution we performed two different sets of comparisons. First we analyse the effects of the different resources X detailed before, and second we compare the accuracy of our solution with the equivalent ρ of applying static sampling.

As expected, the more resources used for the reporting the more accuracy we obtain on the results. In Figure 2 we can see the empirical Cumulative Density Function (CDF) for the different resources and a $\tau = 0.45$. For clarification the figure shows the effective sampling rate since it states the reduction in resources more clearly.

In the figure it is possible to see that increasing the resources brings a big boost to the accuracy, even if the final effective sampling applied is similar. Just increasing a 1% the sampling rate delivers a much higher accuracy. Table III details the most important statistics for the study.

As the second study, we apply to the same traces the uniform static sampling and compare the results with our distributed adaptive solution. The differences can be observed in Figure 3 where the results highlight the improvement in accuracy of the absolute error, which is clearly noticeable for equivalent sampling rates. The figure details only mean values and in the case of our approach the 95% confidence interval is shown. We do not show the confidence interval for the static sampling because it is too large, which confirms the better results obtained with our solution.

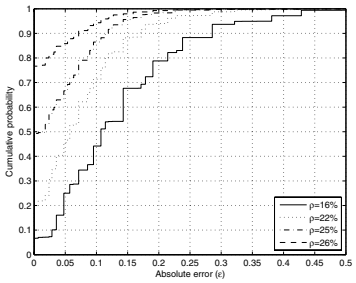

Fig. 2. PLR estimation error respect to the full trace. Empirical CDF per sampling rate ($\tau = 0.45$)

	Mean	StDev.	95^{th} Prc.
$\rho = 16\%$	0.13%	0.10	0.38
$\rho = 22\%$	0.07%	0.07	0.21
$\rho = 25\%$	0.04%	0.05	0.14
$\rho = 26\%$	0.01%	0.04	0.11

TABLE III
STATISTIC VALUES FOR THE ERROR WITH REAL TESTS

B. Density threshold

The density threshold parameter τ is very important in this environment. It determines the sensibility of the reporting system when detecting packet losses. The lower we set τ the quicker we will reserve more resources to lossy flows or paths. But on the other hand this can force an over-reservation of the resources to flows with low loss ratio. The opposite is also true, with very high values of τ we take too long to react, hence we are underreacting to a potential congestion.

Figure 4 shows exactly this effect. In the figure the X-axis has the different τ values while in the Y-axis we show the absolute error, taking as reference the full trace. The figure shows that both for low and high density values the absolute error increases considerably. While the lowest values are accomplished for middle thresholds.

On the contrary, if we had an scenario without congestion and we wanted to detect shorter loss bursts or sporadic losses we should lower the density threshold. But, in the case of extreme congestion we should chose a higher value, just for guaranteeing that we do not over-react in case of sporadic losses. Hence τ is an input parameter which can be decided depending on specific needs.

VII. DISCUSSION

As we discussed before, our distributed adaptive sampling technique clearly outperforms static sampling. Although there is one open issue to be discussed, it is analysing of the cost of a real deployment of the architecture. All over the paper we considered $\mathcal{S} = 1$ which in a real scenario it is not true.

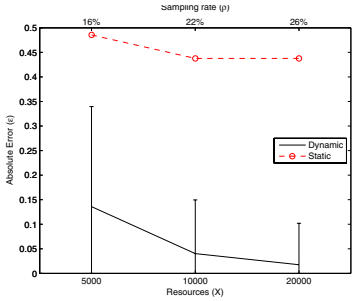

Fig. 3. Static versus Dynamic adaptive sampling ($\tau = 0.45$)

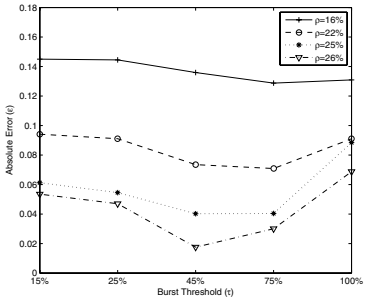

Fig. 4. Density effect on PLR accuracy

The main goal of our proposal is that the final resource utilisation is bounded by X while when using static sampling, once we decide the sampling rate, the used resources will grow or decrease dynamically following this equation: $X_S = \rho_S \; \mathcal{S} \sum_{k=1}^{m} R_k$ where m is the total amount of flows within the domain, R_k as defined before is the rate of flow k. Which depends linearly of the sampling rate.

For the testbed used in this paper, the total traffic generated in average is $\sim 200Mbps$ counting all the flows among all the 12 testbeds. Thus, considering the $\mathcal{S}$ values obtained in [3], the equivalent resources for the adaptive sampling solution are: $X = 5000 \sim 470Kbps, X = 10000 \sim 600Kbps, X = 15000 \sim 700Kbps$ and $X = 20000 \sim 800Kbps$ in average. While not using traffic sampling the amount of required resources is $2.6Mbps$.

VIII. Conclusions and Future Work

In this paper we proposed a technique for on-line Packet Loss Ratio estimation for SLA assessment. Our solution is based on a distributed adaptive traffic sampling mechanism, which dynamically adjusts the sampling rate in a per flow basis in order to minimise the error, but always bounding the maximum resources used for the task.

The proposal uses an enhanced definition of loss burst which avoids overreacting to sporadic packet losses in a configurable way. Moreover the algorithm permits to maintain the sampling rate proportionally to the loss ratio on the path, not underreacting when a packet is not lost during a congestion period.

We applied this mechanism in a real scenario and the results show reasonable accuracy. Moreover, the proposal outperforms the static sampling solution in all the different experiments by $\sim 25\%$.

As work for further study we plan to study and effects of dynamic bin sizes over the system's accuracy, this problem is not straightforward in a distributed environment since all the ME must agree on the bin size policy.

References

[1] Joel Sommers, Paul Barford, Nick Duffeld, and Amos Ron. Accurate and Efficient SLA Compliance Monitoring. In *Proceedings of ACM SIGCOMM*, 2007.

[2] Tanja Zseby. Deployment of Sampling Methods for SLA Validation with Non-Intrusive Measurements. In *Passive and Active Measurements*, 2002.

[3] René Serral-Gracià, Pere Barlet-Ros, and Jordi Domingo-Pascual. Coping with Distributed Monitoring of QoS-enabled Heterogeneous Networks. In *IT-News (QoSIP) - 2008*, pages 142–147, 2008.

[4] A. Hanemann, J. W. Boote, and et. al. PerfSONAR: A Service Oriented Architecture for Multi-Domain Network Monitoring. In *Third International Conference on Service Oriented Computing, LNCS 3826, ACM SIGsoft and SIGweb*, pages 241–254, 2005.

[5] Paul Barford and Joel Sommers. Comparing probe -and router- based methods for measuring packet loss. *IEEE Internet Computing*, Special Issue on Measuring the Internet, 2004.

[6] I. Miloucheva, P.A. Gutierrez, and et. al. Intermon architecture for complex QoS analysis in inter-domain environment based on discovery of topology and traffic impact. In *Inter-domain Performance and Simulation Workshop, Budapest*, March 2004.

[7] René Serral-Gracià, Pere Barlet-Ros, and Jordi Domingo-Pascual. Distributed sampling for on-line qos reporting. In *Submitted and pending acceptance*, 2008.

[8] René Serral-Gracià, Albert Cabellos-Aparicio, and Jordi Domingo-Pascual. Network performance assessment using adaptive traffic sampling. In *IFIP Networking*, pages 252–263, 2008.

[9] ITU-T Recommendation G.107. The E-model, a computational model for use in transmission planning, 03/2005.

[10] M. Borella, D. Swider, S. Uludag, and G. Brewster. Internet Packet Loss: Measurement and Implications for End-to-End QoS. In *International Conference on Parallel Processing*, 1998.

[11] Joel Sommers, Paul Barford, Nick Duffeld, and Amos Ron. Improving Accuracy in End-to-end Packet Loss Measurement. In *Proceedings of ACM SIGCOMM*, 2005.

[12] [IST] EuQoS - End-to-end Quality of Service support over heterogeneous networks - http://www.euqos.eu/, September 2004.

[13] Vern Paxson. Strategies for sound internet measurement. In *Internet Measurements Conference*, 2004.

[14] Evan Ettinger, Brian McFee, and Yoav Freund. Pinpoint: Identifying Packet Loss Culprits Using Adaptive Sampling. Technical report, UCSD, 2007.

[15] M. Yajnik, S. B. Moon, J. F. Kurose, and D. F. Towsley. Measurement and modeling of the temporal dependence in packet loss. In *INFOCOM*, pages 345–352, 1999.

[16] J. Mahdavi, V. Paxson, A. Adams, and M. Mathis. Creating a scalable architecture for Internet measurement. In *INET*, 1998.

[17] S. Savage. Sting: A tool for measuring one way packet loss. In *INFOCOM*, 2000.

[18] Tanja Zseby. Comparison of Sampling Methods for Non-Intrusive SLA Validation. In *E2EMon*, 2004.

[19] C. Veciana-Nogués, A. Cabellos-Aparicio, J. Domingo-Pascual, and J. Solé-Pareta. Verifying IP Meters from Sampled Measurements. In *Kluwer Academic Publishers. Testing of Communicating Systems XIV. Application to Internet Technologies and Services. IFIP TC6/WG6.1. TestCom*, pages 39–54, 2002.

[20] Nick Duffeld. Sampling for Passive Internet Measurement: A Review. *Statistical Science*, 19(3):472–498, 2004.

[21] Baek-Young Choi, Jaesung Park, and Zhi-Li Zhang. Adaptive Packet Sampling for Accurate and Scalable Flow Measurement. In *Proceedings of IEEE Globecom '04*, 2004.

[22] GÉANT - http://www.geant.net.

[23] Len Kleinrock. *Queuing Systems*, volume I: Theory. Wiley, September 1975.