This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2009 proceedings.

Towards an efficient Service Level Agreement assessment

René Serral-Gracià*, Yann Labit†‡, Jordi Domingo-Pascual*, Philippe Owezarski†
* Advanced Broadband Communications Centre, Technical University of Catalunya (UPC), Spain
Email:{rserral, jordid}@ac.upc.edu
†LAAS-CNRS; Université de Toulouse; 7, avenue du Colonel Roche, F-31077 Toulouse, France
Email:{ylabit,owe}@laas.fr
‡ Université de Toulouse ; UPS

***Abstract*—On-line end-to-end Service Level Agreement (SLA) monitoring is of key importance nowadays. For this purpose, past recent researches focused on measuring (when possible) or estimating (most of the times) network QoS or performance parameters. Up to now, attempts to provide accurate techniques for estimating such parameters have failed. In addition, live reporting of the estimated network status requires a huge amount of resources, and lead to unscalable systems.**

The originality of the contribution presented in this paper, relies on the statement that the accurate estimation of network QoS parameters is absolutely not required in most cases: specifically it is sufficient to be aware of service disruptions, i.e. when the QoS provided by the network collapses. For this purpose, we propose an algorithm for disruption detection of network services. The proposed solution is based on the use of the well-known Kullback-Leibler Divergence algorithm. More specifically, we work on simple to measure time series, i.e. received inter-packet arrival times. In addition of efficiently detecting network QoS disruptions, the algorithm, also drastically reduces the required resources, and the overhead produced by the traffic collection for scalable SLA monitoring systems.

The validity of the proposal is verified both in terms of accuracy and consumed resources in a real testbed, using different traffic profiles.

I. Introduction

On-line end-to-end Service Level Agreement (SLA) monitoring is of key importance nowadays. Both ISP and customers want to know at any time the quality of the network services, and whether it is respecting the contracted SLA. For this purpose, classical approaches for SLA assessment [1], [2], [3] focused on accurately measuring (when possible) or estimating (most of the times) network QoS or performance parameters such as One Way Delay (OWD), Inter Packet Delay Variation (IPDV), Packet Loss Ratio (PLR), etc.

In this context, computing the metrics presents several drawbacks:

- Estimating the metrics implies to gather distributed information about the traffic and the synchronisation among the involved entities. Such control traffic is an important bottleneck of any solution using this approach [1].
- Computing the metrics requires multiple capture points. And in the case of using active probing a traffic generation station located in some advantage point.

Therefore, such systems suffer from large scalability issues.

In such context, we do not intend to improve the accuracy on the QoS metric estimation. The originality of the contribution presented in this paper relies on the statement that the accurate estimation of network QoS parameters is absolutely not required in most cases: specifically it is sufficient to be aware of service disruptions, i.e. when the QoS provided by the network collapses. In our original approach we just focus on the actual scalable detection and reporting of any potential violation of the SLA in the network. For this purpose, we propose in this paper a new approach which:

1) Works as much as possible with a single point of analysis
2) Computes data very efficiently in order to have a scalable system
3) Relies on the use of existing correlation between measured parameters and network quality

We then propose to use Inter Packet Arrival Time (IPAT) because it complies with the above restrictions: IPATs can be easily computed at destination by getting the reception timestamps of the packets; IPAT computation only involves a subtraction of two integers (timestamps). Finally, and this is what we want to prove in the rest of the paper, it exists a strong correlation between IPAT distribution and network performance: it was demonstrated by previous work that IPATs are tightly correlated with network congestion [4]. In this work, we bring this correlation one step further by mapping these IPATs with the actual network conditions by using some information about the real metrics. In particular, our proposal relies on statistical analysis of the IPATs, with the goal of detecting changes on the network status. This is done by comparing different IPAT distributions using *Kullback-Leibler Divergence*.

We validate our solution with a real scenario, using a controlled testbed with customisable network conditions. Our results confirm the adaptability and accurate detection of the different SLA violations found in our traces, all accomplished by reducing the resource usage down to a $\sim 25\%$, with an

This work was partially funded by IST under contract 6FP-004503 (*IST-EuQoS*), 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.

978-1-4244-3513-5/09/$25.00 ©2009 IEEE

accuracy of $\geq 85\%$ in the worst presented case, compared to the perfect knowledge of the SLA violations.

The rest of the paper is structured as follows. Next section details some related work. After this, as background, we describe the *Kullback-Leibler Divergence*. In Section IV we present the main contribution and methodology used to perform the on-line SLA assessment. After this the paper describes the different testbeds and the evaluation results. Finally in Section VII we conclude and explain the open issues for future work.

II. Related Work

In the active traffic analysis area, SLA assessment has been previously studied. Some important work has been performed by Sommers et al. in [3], where the authors present *SLAm*, an active probing tool, which implements innovative packet loss, delay, and delay variation estimation techniques for SLA assessment. In this research, the authors stress the need of proper metric estimation in order to lead to correct SLA assessment. Our work differs from this in the sense that our methodology does not focus on accurate metric estimation, but rather in the search for relevant packet information to infer the network quality. Moreover, we use a passive traffic analysis approach against the active solution adopted by the mentioned work.

Regarding passive traffic analysis, some research has been done in our previous work [1], [2], in which we proposed a distributed infrastructure, for inferring the network quality by extracting the performance metrics from detailed packet information. The metric computation is centralised and requires several collection points on the edges of the network that send packet information (timestamps, etc.) to the central entity, with the consequent use of bandwidth due to this, so called, control traffic. Such received information is used by the central entity to compute the flow's network metrics (OWD, IPDV and PLR). In this paper, we use this mechanism as base for acquiring perfect knowledge about the network status for our SLA assessment algorithm.

In the main contribution of this work we use the *Kullback-Leibler Divergence* [5] in order to infer SLA violations. This algorithm has been used before in [6] to perform anomaly detection based on destination ports. Nevertheless, we apply it to a completely different scenario.

III. Detection of SLA violations

As described before, we plan to use IPAT in order to infer violations in the SLA. Even with their good characteristics in terms of computational efficiency, just gathering IPATs is not enough to provide SLA assessment. First, IPATs do not contain information about the useful metrics of the network. Second, IPAT might change unexpectedly, sometimes due to real changes on the network conditions, but also due to the change in the traffic profile (e.g. change in the codec, silence on the conversation, etc.). Third, there is no direct mapping between IPAT and SLA violations.

Consequently, the contribution of this work is the detection of the SLA disruptions with minimal computation of the network performance metrics. We achieve this by periodically comparing the current IPAT distribution with a reference distributions set. Of course, getting the reference distributions set means integrating an on-line training process which records all new IPAT distributions observed on the network. It also establishes the link between each of these new IPAT distributions and the current QoS parameters (by measuring them). Then, each IPATs distribution will be associated to a particular QoS level of the network. Then, in this section we focus on the generic description of the *Kullback-Leibler Divergence*.

A. Kullback-Leibler Divergence

Entropy, in the area of information theory, is a measure of the uncertainty associated with a random variable. Sometimes the actual entropy value is not directly indicative of any interesting property. In this context, it is more useful to consider the *relative entropy* or *Kullback-Leibler Divergence*, which indicates the difference (i.e., how far) a distribution is from another. *Kullback-Leibler Divergence* [5] is defined as: $K(P,Q) = \sum_i P(x) \cdot log \frac{P(x)}{Q(x)}$.

The above expression gives the degree of similitude between both distributions (P and Q), taking P as the reference, against Q, the one to be tested. The outcome is the divergence between both distributions.

In this paper we use this divergence as a measure of the difference on the IPAT distribution, which indicates potential changes in the network conditions.

IV. SLA Violation Detection

Computing the divergence among distributions determines how different are the reference and the acquired traffic profiles at the egress node, but this information alone is not sufficient to perform the SLA assessment. In this section we present the methodology to detect SLA violations by using the IPAT information.

A. General Methodology: SLA violation detection

Informally, the base algorithm we use for the violation detection is the following: first, we collect the IPATs during a time period at the egress node. Second we compare their distribution with a reference distribution set. If the distributions are *similar*, we assume similar network behaviour in both cases. Otherwise, we query the ingress node to acquire detailed packet information, from which we compute the real performance metrics. Finally we can assess the status of the network and report any encountered SLA violations.

In summary, we use the performance metrics to map the IPAT distribution to the real network status, and we use such distribution as a reference to infer the SLA compliance.

The collection of IPAT distributions is performed in a per flow f basis, during a predefined time interval t. The empirical distribution is computed in bins of width w (referring to IPAT ranges), therefore, a particular IPAT i falls in bin $k = \lfloor \frac{i}{w} \rfloor$.

Due to space limitations we defer the study of proper t and w values to [7].

After the distribution acquisition, to cope with the aforementioned IPAT variability, our algorithm considers the following actions:

- *Training and update* the traffic profile.
  - *Map* it to the actual network status.
  - *Update* the set of valid distributions if needed.
- *Compare* the current profile with the learned status.
  - *Decide* whether the traffic conditions changed or not.

The rest of the section performs a thorough description of the algorithm. We start by how the system learns the real offered network quality. Later we focus in the distributions comparison.

B. Training and Update Strategy

Any system with adaptability requirements must have a robust Training mechanism. Before entering with the full description of the Training and Update Strategy, we need to define a few concepts.

Definition 1: A *Valid IPAT Distribution* (VID) $\mathcal{V}$ is such a distribution where a function of the real metrics (OWD, IPDV and PLR) fall above a specified SLA threshold ν. The complete discussion about how $\mathcal{V}$ is computed is done in Section IV-D.

Definition 2: Let's define a *Reference Distribution Set* (RDS) $\mathcal{D}$, as a set of strictly different VID distributions. Where $|\mathcal{D}|$ is the cardinality of the set, $\mathcal{D}^1$ is the first element and $\mathcal{D}^{|\mathcal{D}|}$ the last one, with a maximum size for the RDS bounded by a predefined Δ, which limits the maximum memory usage of the RDS.

The *Training and Update Strategy* is in charge of keeping an updated and valid version of the RDS.

A prerequisite of the RDS is that all the stored distributions must represent good reference traffic conditions to compare with. Therefore, our system must have some means for correctly assessing them; we use the technique presented in [1], where the source measurement point (i.e., the ingress router) sends per packet information, such as transmission timestamps, which are matched on the destination measurement point (i.e., the egress router), computing the relevant performance metrics. This technique requires control traffic from source to destination to compute the metrics as discussed before. Such control traffic determines the amount of bandwidth (resources) required by the system. For the sake of efficiency and scalability it should be minimised. Nevertheless, this method reports exact values of the network metrics that we can use to map to the IPAT distribution.

Once the *real* validity is assessed, if it is below ν, the event is registered, the distribution discarded, and a *SLA Violation* event is triggered. Otherwise we insert the distribution on the RDS.

This *Training* is only invoked when the distribution comparison is not accurate (i.e., when network conditions are unknown).

C. Distribution comparison

The metric we chose to compare between two distributions is the divergence, which describes the degree of similitude (dissimilitude) between two distributions. The higher the divergence the more different the distributions (and so does the traffic profile).

Since RDS is composed by a set of distributions, the divergence cannot be directly computed. Hence we define:

Definition 3: Degree of Matching D_M, between a RDS $\mathcal{D}$ and another distribution Q is defined as:

$$D_M(\mathcal{D},Q) = \quad \min\{d(p,Q)\} \quad p = \mathcal{D}^1 \dots \mathcal{D}^{|\mathcal{D}|} \qquad (1)$$

where $d(p,Q)$ is a predefined divergence algorithm between the distributions p and Q as presented in Section III.

Then Q and $\mathcal{D}$ are considered similar if $D_M(\mathcal{D},Q) \leq \delta$, where δ is our divergence threshold. The critical point here is that different distributions do not mean different qualities, since the traffic profile can change over time, even with the same quality level. Therefore, when the distributions are considered different the system must learn the degree of quality of the new distribution. The *Training* procedure is then invoked with Q. Because training consumes system resources, as described previously, there is a trade-off here: the lower δ, the more resources (queries) will be needed. On the other hand, the higher δ, the lower amount of resources will be required, at the cost of losing some accuracy on the SLA assessment. Refer to [7] for a full description of the effects of changing δ.

D. Distribution Validity $\mathcal{V}$

Our final goal is to assess whether the network is honouring the SLA between the ingress and the egress points of the network. In this work, as a proof of concept, we assume a simple linear SLA compliance policy.

$\mathcal{V}$ is defined in the range $[0,1]$ indicating the quality of service experienced for the flow with respect to the SLA. To compute this value we consider the usual metrics (OWD, IPDV and PLR).

Another point to consider is that, depending on the type of traffic, the QoS constraints might considerably differ. Hence, we define $\omega_O, \omega_I, \omega_P$ as weights specified for each particular metric, where $\omega_O + \omega_I + \omega_P = 1$.

Expression 2 computes $\mathcal{V}$, which is the degree of validity of the time interval.

$$\mathcal{V} = Q^O(\overline{OWD}) \cdot \omega_O + Q^I(|\overline{IPDV}|) \cdot \omega_I + Q^P(PLR) \cdot \omega_P \quad (2)$$

Where $Q^*(x)$ determines the exponential quality degrading function defined as:

$$Q^*(x) = \begin{cases} 1, & x \leq \mathcal{X} \\ \lambda e^{-\lambda(x-\mathcal{X})}, & \mathcal{X} < x < \mathcal{M} \\ 0, & otherwise \end{cases} \qquad (3)$$

Where $\mathcal{X}$ is the metric dependent threshold of quality degradation specified by the SLA, and $\mathcal{M}$ the upper feasible bound for the quality of that particular metric. Finally, λ in $(0,1)$ is the decaying factor for the exponential quality degradation function.

Then $\mathcal{V} \geq \nu$ the network behaviour is considered stable. The closer is ν to 1, the stricter our system will be to SLA violations.

V. Tests and Testbeds

Our validation is performed by using two different testbeds as follows.

A. Synthetic traffic under controlled testbed conditions

We set up a testbed, where we performed several tests using synthetic traffic generation, under a tightly controlled environment. We configured two end nodes in order to generate and collect traffic. On the core of the testbed we installed two servers with *Traffic Control* and `NetEM` emulator capabilities present in most recent Linux kernels. We then can change the network conditions according to our needs and experience a wide range of controlled network disruptions.

The set of emulated network conditions are:

1) *Good Network Conditions:* no SLA disruptions and good network behaviour all over the test.
2) *Mild Network Disruptions:* moderated increase of OWD with periods of high IPDV and some packet losses. Some traffic disruptions but only with few SLA violations per test.
3) *Medium Network Disruptions:* similar to the mild network disruptions but with limited buffers on the routers which leads to moderate periods of packet losses.
4) *Severe Network Disruptions:* random losses from 1% to 10% with variable OWD. Severe SLA violations in regular intervals on the test.

We performed tests with *Periodic*, *Poissonian* and *Synthetic Real Traffic* [8] traffic profiles with all the above network conditions.

B. Synthetic traffic over the European Research network

In this testbed we performed a set of more than 500 experimental tests during 2006 and 2007 using twelve different testbeds across Europe. The testbeds cover a total of 5 countries and 4 different access technologies (LAN, xDSL, UMTS and WiFi) with an overlay architecture over the Géant research network.

We evaluated the performance of our system by actively generating UDP traffic on the network with different properties. Specifically, we focus on three different sets of tests. The first one simulates a low rate, small size packets with a used bandwidth of $64Kbps$. We label this traces as (synthetic) `VoIP`.

The second type of traffic is a periodic flow with average packet rate of ~ 96 packets per second, with MTU size packets amounting to a total of $1Mbps$ of UDP traffic. We call this trace `UDP1`. Finally, the third kind of traffic is a average sized, high rate UDP flow with $\sim 1.4Mbps$. We call this test `UDP2`.

VI. Evaluation

The validation of the proposal is issued by evaluating the tests described in the previous section.

We focus the study in the system's accuracy, that is, in the SLA violation detection rate, measured in terms of false negatives (i.e., not detected SLA violations), notice that false positives cannot happen in our methodology, since we assess specifically all the found violations. We compare the proposed algorithm with the case of perfect knowledge about the SLA violations. We also analyse the amount of resources required by the system; such resources are counted in terms of reduction ratio of the required bandwidth used by the control traffic. Therefore, we compare the cost of reporting per packet information with our solution, which only demands information when there is a change in the traffic reception profile.

During all the analysis we use the same parameters across the tests for the estimation. In particular, we set up, as a proof of concept, the following values: divergence threshold of $\delta = 3\%$, bin width of $w = 3ms$, and an acquisition time interval of $t = 175ms$.

A. Methodology

Analysing all the information obtained from the tests is complex. To ease the comprehension of the validation process, we use the following methodology:

1) For each test we collect the full trace on both end nodes.
2) We match the packets extracting the network performance metrics as described in [1], using them as reference quality (perfect knowledge).
3) We identify the different SLA violation periods with the reference results acquired above.
4) We apply off-line our algorithm. Here we register: *i)* required control traffic due to *Training*. *ii)* estimated SLA violation periods.
5) Finally, we match the SLA violations with the ones obtained in Step 3.

B. Accuracy and Resources requirements

In order to study the behaviour of our system, here we discuss the achieved accuracy together with the analysis of the required resources in the different testbeds.

1) Synthetic traffic with controlled network: The goal of this synthetic traffic generation is to evaluate the reaction of each algorithm in a controlled environment with the different traffic profiles.

We analyse in Table I the *Accuracy* and the *Resource* utilisation for the different generated traffic. The accuracy is computed for the overall test duration, counting the ratio of detected SLA violations over the total, while the required resources are computed by the ratio of the actual number of queries, over the maximum possible queries per test. Our goal is to achieve high accuracy with low resource consumption.

As it can be observed in the table, the accuracy of the solution is higher for the extreme cases. When there are *Good* network conditions in the network we always estimate correctly, and with very low resource consumption in general. This is because our algorithm assumes correct network behaviour by design. In the case of *Severe* network conditions, where our contribution is more useful, we can detect with

	Accuracy			
	Good	Mild	Medium	Severe
Periodic	1.000	1.000	0.987	1.000
Poisson	1.000	0.250	0.940	0.893
Synthetic	1.000	0.667	1.000	1.000
	Resources			
	Good	Mild	Medium	Severe
Periodic	0.001	0.256	0.385	0.394
Poisson	0.562	0.572	0.657	0.671
Synthetic	0.002	0.002	0.397	0.397

TABLE I
ACCURACY AND RESOURCES FOR $\delta = 0.03$

very good accuracy the SLA disruption periods. On the other hand, in the fuzzy case when there are few SLA violations, the accuracy of the system drops sensibly. The cause of this is the statistical resolution achieved when there are few SLA violations, where missing only one violation is statistically significative. Moreover, in a real deployment, such SLA violations are of no practical interest since they represent very short, sporadic, periods of light congestion, with no high impact on the final network behaviour.

The second consideration is the resources needed by the *Severe*, and some *Medium* network conditions. This is because we always query the ingress node when an unknown IPAT distribution is found. With bad network conditions this situation is common. Hence it forces the system to query for exact metrics. Here the minimum number of queries is bounded by the amount of SLA violations, which in our experimental case is 0.39 as shown in the Table I. In the specific case of Poissonian Traffic, we need more resources than this lower bound. Notice though, that requiring less resources than that would imply non detection of some SLA violations.

2) Synthetic traffic over the European Research network: In this testbed we plan to show the proper accuracy of our proposal in a real network with random quality, unexpected results and unknown cross traffic with different multi-hop paths.

In Figure 1 we show the different accuracy results for each algorithm and traffic profile. The X-axis of the figure has the test number (normalised to 1) and the Y-axis the accuracy. The figure considers all the tests, including the ones without SLA violations.

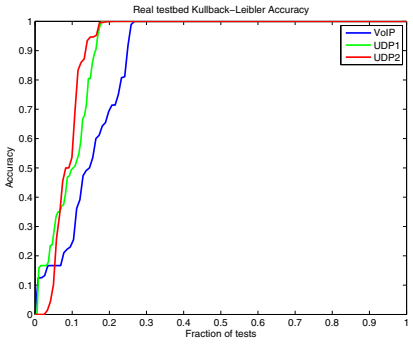

Fig. 1. Accuracy for Synthetic traffic over the European Network

We complement the figure with Table II, which summarises the results of our experiments. We show the aggregated total amount of bins with violations, together with the amount

	Violations	*Detected*	*Accuracy*	*Resources*
VoIP	7216	6096	0.845	0.198
UDP1	62264	60108	0.965	0.551
UDP2	24863	22265	0.896	0.338

TABLE II
VIOLATION DETECTION UNDER A REAL NETWORK, $\delta = 0.03$

our algorithm could detect. In the third column we highlight the overall accuracy and finally, the last column, details the average amount of resources needed for the reporting.

We manually checked that most of the failures in the SLA estimation are due to isolated bins with violations very close to the SLA agreement boundary with no practical interest, similarly to the case we found in the previous testbed.

In terms if resources, the average resource usage of the whole system is below 25%. It is lower when considering `VoIP` traffic (i.e., around 4%).

VII. CONCLUSIONS

We have presented a novel approach to on-line SLA assessment, where differently of previous research, our work separates and reduces the performance metric computation and the interaction between the edge nodes of the network. This is accomplished by: *i)* a smart algorithm for gathering the distribution of Inter-Packet Arrival Time (IPAT); *ii)* a divergence algorithms to compare the distributions; and *iii)* a robust Training methodology that delivers a very competitive solution regarding SLA violation detection.

We validated our methodology with a set of different tests, which involved a controlled and European-wide testbeds, using synthetic and real traffic. The experimental results show that we can reduce the required resources considerably, with a low effect on the final accuracy of the system.

As lines left for further research, an interesting upgrade of the system would be to infer the real metrics of the network by comparing ingress and egress packet arrival times (Inter Packet Generation Time with Inter Packet Arrival Time).

REFERENCES

[1] René Serral-Gracià, Pere Barlet-Ros, and Jordi Domingo-Pascual. Coping with Distributed Monitoring of QoS-enabled Heterogeneous Networks. In *4th International Telecommunication Networking Workshop on QoS in Multiservice IP Networks*, pages 142–147, 2008.

[2] René Serral-Gracià, Albert Cabellos-Aparicio, and Jordi Domingo-Pascual. Network performance assessment using adaptive traffic sampling. *IFIP Networking*, LNCS 4982:252–263, May 2008.

[3] Joel Sommers, Paul Barford, Nick G. Duffeld, and Amos Ron. Accurate and Efficient SLA Compliance Monitoring. In *Proceedings of ACM SIGCOMM*, pages 109–120, 2007.

[4] D.A. Vivanco and A.P. Jayasumana. A Measurement-Based Modeling Approach for Network-Induced Packet Delay. *Local Computer Networks (LCN). 32nd IEEE Conference on*, pages 175–182, 2007.

[5] S. Kullback and R.A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[6] Andrew McCallum, Don Towsley, and Yu Gu. Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation. In *Internet Measurement Conference (IMC)*, pages 345–350, 2005.

[7] René Serral-Gracià et al. Parameter evaluation for distributed SLA assessment. In *R. Rep. http://personals.ac.upc.edu/rserral/research/techreports/distance-full.pdf*, 2008.

[8] P. Owezarski, P. Berthou, Y. Labit, and D. Gauchard. LaasNetExp: a generic polymorphic platform for network emulation and experiments. *Proceedings of the 4th International Conference on Testbeds and research infrastructures for the development of networks & communities*, 2008.