Sampling Time-Dependent Parameters in High-Speed Network Monitoring

Edurne Izkue
Universidad Pública de Navarra
Campus Arrosadia
31006 Pamplona, Spain
+34 948 169853
edurne.izkue@unavarra.es

Eduardo Magaña
Universidad Pública de Navarra
Campus Arrosadia
31006 Pamplona, Spain
+34 948 169853
eduardo.magana@unavarra.es

ABSTRACT

Nowadays network bandwidth is increasing continuously for end-users and network providers. Network monitoring tools have to be able to support these high-speed networks, processing a high number of packets per second. For this reason, network monitoring tools have to be improved using software or hardware techniques. In this paper we use sampling techniques, as a software technique that can provide the capacity of monitoring networks with high bandwidth, keeping a low-cost hardware platform. A new technique called mixed sampling is proposed and compared with other techniques proposed in the literature. We focus on time-dependent statistics not studied before with sampling. These statistics will require applying estimation techniques for better results.

Categories and Subject Descriptors

C.2 COMPUTER-COMMUNICATION NETWORKS,
C.2.3 Network Operations: Network monitoring

General Terms

Algorithms, Measurement, Performance.

Keywords

Network monitoring, sampling, estimation.

1. INTRODUCTION

Network monitoring techniques have to be adapted to new high-speed access and core networks. The Internet growth explosion and new applications consuming more bandwidth mean that network monitoring infrastructure has to be adapted. There are multiple proposals in the literature to optimize network monitoring tools from different points of view, mainly around hardware and software options.

This work was supported by the Spanish Ministry of Education and Science (Project PINTA TEC2004-06437-C05-03).

In hardware, we can use specific network monitoring cards like Endace DAG [1] or Scampi COMBO6 [2]. They provide specific functionalities like larger buffers or a programmable processor that is able to make some preprocessing. This means freeing up the main CPU. In software, we can find algorithms to optimize filtering tasks, for example, organizing multiple filtering rules in a hierarchical way [3]. In the middle of both, interrupt mitigation tries to reduce the number of interruptions per second. Usually one interrupt warns about a packet waiting for processing, but we can group packets together and launch one interrupt for the group of packets [4]. The overhead introduced by an interrupt is relatively high, so in this way we can optimize the use of CPU resources. Other kind of improvement would be reducing the number of copies between the network card and the monitoring application. Techniques like having a shared memory between the driver at kernel space and the application at user space can reduce the number of packet copies [5].

Another way to achieve higher monitoring speeds is to use packet-based sampling. This technique can provide good results if applied to certain statistics like packet size distribution, inter-arrival time distribution, etc. These measurements are mainly related to those that can be collected over certain time interval. In that interval, there are several occurrences of the event that we want to measure and we sample some of the events. After some post-processing we can obtain a good approximation of the distribution of that event without collecting all those events. Therefore, these statistics are normally related with distributions or density function of some parameter [6][7][8].

In [6], two sampling mechanisms are compared: time-driven and event-driven. In the time-driven technique, a timer is configured to provide a packet each certain interval time. In event-driven technique, a packet of each *m* packets is considered so it can reduce the hardware resources to *1/m*. The paper focuses in packet size and inter-arrival time distributions measurements.

A simple model of the number of concurrent flows that must be accommodated in a router memory is presented in [7]. It uses sampling over original traffic flows, and it takes into account the possible flows present in unsampled packets. In [9] the authors propose a correlated sampling strategy that is able to select an arbitrarily small number of the best representatives of a set of flows. Larger flows are referentially sampled over small ones, exploiting the fact that a large fraction of usage is contained in a small fraction of flows (larger ones). Non-uniform sampling and size dependant sampling is used in [10].

Up to our knowledge there are no studies of sampling in other kind of monitoring parameters less related with distributions (stationary parameter) and more related with classical network measurements like bits per second, packets per second, etc (time-dependent parameter). It is usual in the literature to find packet sampling applied to packet size distribution, inter-packet delay distribution and flow size distribution over certain time interval. With only a percentage of packets sampled we can infer the distribution applicable to all packets inside that interval. However the application of sampling to other measurements like bits per second or packets per second has new problems to approach. Applying sampling for these time-dependent parameters is more complex because we have to use some kind of estimation for the packets not taken into account. If we want to measure bits per second each second with only some sampled packets we have to estimate the size of not sampled packets in order to have an approximation to the real figure of bits per second. In that way, it is not a simple collection of realizations of the same event as before. We have a time series and the underlying process can be or not stationary. Studies from previous work can not be extended easily to multiple time intervals repeatedly for time series estimation, In this paper we will apply sampling to this kind of traffic parameters and we would apply a new technique between time-driven and event-driven sampling: mixed-driven.

The paper is organized as following. Section II introduces the concepts of sampling in network monitoring. Section III describes the scenario and network traces. In Section IV we present the main results. Finally, conclusions and references end the paper.

2. SAMPLING IN NETWORK MONITORING

2.1 Time-dependent parameters

As explained before, typical studies in packet sampling have been applied to stationary parameters like the calculation of packet size or inter-arrival time distributions in certain interval time. However, the application of sampling in time-dependent parameters has not been studied in deep.

With time-driven or event-driven sampling, we collect only some of the packets in the network. It decreases the real knowledge about network state. If for example we want the parameter bits per second, we can collect 1 packet for each interval time of 500ms. Then we have two possibilities:

- Count the packets unsampled: we will need to estimate the packet size of the packets not sampled, but we know the number of them.
- Not count the packets unsampled: we will need to estimate both, the packet size and number of packets not sampled.

For our study we will suppose the first case in which we know the number of packets not sampled. This is not a hard requirement because for example in event-driven sampling, it is requisite to count the packets received in order to choose the one to sample. Usually, this functionality can be provided by the network card.

In any case, we only sample a characteristic of one packet and we need to estimate the same characteristic for the rest of packets not sampled. For example, if we are considering the statistic bits per second, we will have to add the sizes of all packets received in one second. In the case of sampling, we will only know the size of some of the packets, having to estimate the size of the others.

2.2 Sampling methods

Studies like [6][7] show that the event-driven sampling gives better results. However, both event-driven and time-driven have their drawbacks.

In event-driven sampling, if there is low network load, the time interval between samples is going to be bigger and the accuracy of the results will be worst. Opposite, in networks with high load conditions, the number of samples will be high. The sampling factor, *m*, means that one packet is sampled from a block of m packets. We get one sampling value each m packets.

In the case of time-driven sampling, if there is low network load we will sample almost all packets. However, if the network load is high, we will have several packets per interval and we will only sample some of them, loosing again accuracy.

The mixed-driven sampling procedure proposed in this paper joins both event and time-driven methods as can be seen in Figure 1. That figure shows several packets in a timeline and some of them (filled up packets) are the chosen to be sampled. In event-driven sampling one of each three is sampled. In time-driven the first packet of the interval T is sampled. In mixed-driven we combine both techniques, sampling the first packet that would be sampled for any of both techniques.

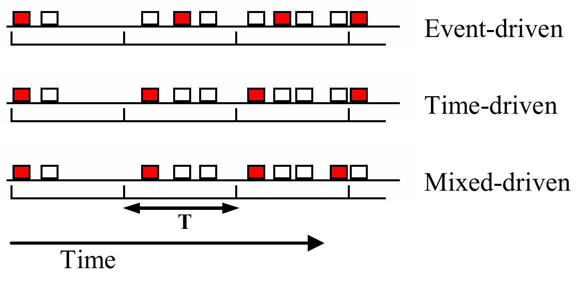

Figure 1: Sampling methods

So, mixed-driven sampling behaves as a time-driven sampling in low load network conditions and as an event-driven sampling in high load network conditions. In both cases, mixed-sampling will give accurate results, adapted to network conditions.

2.3 Techniques for sample selection

The sampling methods specify the procedure to choose each packet to be sampled from a number of them. However, we can sample the first one, the last one or any inside the interval time where we have to sample. Such interval will be referred in this paper as block. In time-driven sampling, the block will be constant in size, but in event-driven or mixed-driven it will be changing. Then, inside a block we can choose the packet to sample applying one of the techniques showed in Figure 2 as proposed in [6]:

- Systematic: choosing always the same first, second,..., or last packet in the block as the sample.
- Stratified random: choose a random packet for each block. In one block could be the first, in other the 3rd, etc.
- Simple random: consider a superset of *M* blocks and choose *M* random packets of the complete interval of *M* blocks.

• Variable-length systematic: choose the length of the block in a random way and inside the block choose the packet to sample in a systematic way.

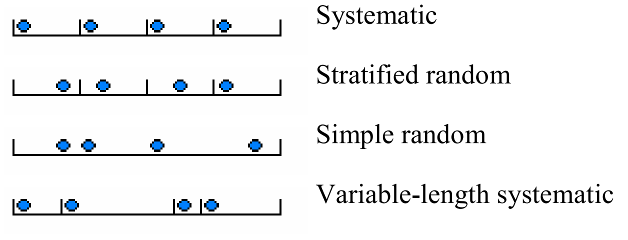

Figure 2: Sample election techniques

In our study we will compare the results from those four sample election techniques. The random values will be obtained from uniform distributions and exponential distributions with mean *m*, reducing packet processing to *1/m* of the original packet rate.

2.4 Parameter estimation

For applying sampling methods to time-dependent parameters, we need some way to estimate the parameter in study for those packets unsampled. For example, if we sample one of each three packets to obtain the network load, we will know the size of the sampled packet but we will need some way to estimate the size of not sampled packets.

We propose the following possibilities for parameter estimation:

• Linear estimation: the parameter measured with the sampled packet is kept for the not measured packets. For example, if we measure packet size, we will assume identical size for all the packets of the same block.

• Mean estimation: knowing some mean value of the parameter to measure, we suppose all not sampled packets with that mean parameter. This estimation needs a previous work of characterization of network parameters to obtain a generic mean value.

• Dynamic estimation: the generic mean value is obtained dynamically over all the parameter values of the sampled packets. At the beginning of the measure the estimation will be very bad, but it will improve with the number of sampled packets.

2.5 Error metric

In order to decide which is the best combination of sampling methods, election techniques and parameter estimation, we need to compare values of the parameter to measure considering all values (not sampling) and applying sampling. Besides, we need some way to resume the results of a time-dependent parameter over some interval time. We could use mean square error or something similar. However, sampling theory has considered better error measurements. In [11] an alternative metric is presented. This metric ϕ is independent of the size of the distributions to compare and it is defined as:

$$\phi = \sqrt{\frac{\chi^2}{n}} \quad \text{where } n = \sum_{i=1}^{B}(E_i + O_i) \quad \text{with } \chi^2 = \sum_{i=1}^{B}\frac{(O_i - E_i)^2}{E_i}$$

E_i is the real value of the parameter considering all packets, and O_i is the value of the parameter considering sampling methods. B will be the number of values to compare. In our case, if we are measuring the network load in bits per second, the value of bits each second will be the value to compare, and the number of values will be the number of seconds considered for the comparative.

The value of this coefficient ϕ will be 0 if the trace of real values coincides exactly with the trace from sampled packets. Bigger values of ϕ will indicate worst results in the sampled trace. We will use this coefficient to compare the results of different sampling methods.

3. MEASUREMENT SCENARIO

The analysis has been made using real traffic traces obtained from the Internet link of the Public University of Navarra with around 4000 computers. This university is connected to the academic spanish network called RedIris through a dedicated POS/SDH link of 155 Mbps; however the real traffic is in the order of 10-18 Mbps.

The capture of the packet trace has been made with a Linux machine using libpcap library, between January 23 and 29, 2006. The study presented here only refers to an hour: 12-14h, January 23, 2006. All the study has been made off-line using this trace, but the results are applicable to a scenario with real-time traffic. We will concentrate on the network load parameter in bits per second (bps). In figure 3 the bps real parameter is plotted with 1 second averaging interval. The results can be extended to other more fine grained statistics, like load per user or service, or other parameter statistics. To reduce the overhead of results, we will only use the systematic technique as sample election technique.

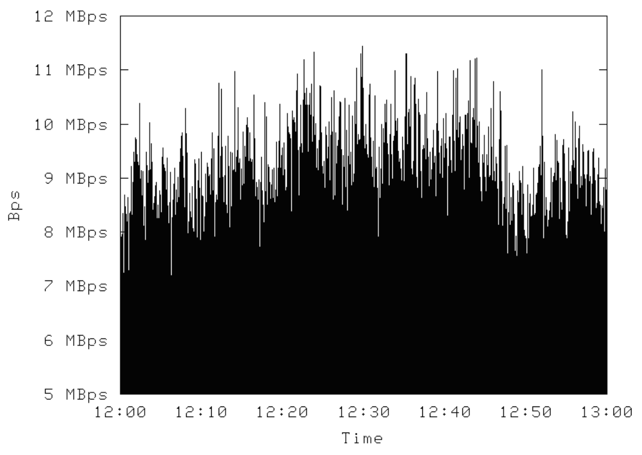

Figure 3: Network load in bps for January 23, 2006, 12-13h

4. RESULTS

As we said before, a very important aspect of packet sampling is the parameter estimation. In the case of bps we will need to estimate the packet size for those unsampled packets. To observe the variability in the packet size, Figure 4 shows the packet size density function for the hour of Figure 3. Basically we find packets of the minimum size (64 bytes) and maximum size (1500 bytes). Because of this high variability in packet size, proposed mean estimation will not provide good results at first. However we will check that it depends on interval aggregation.

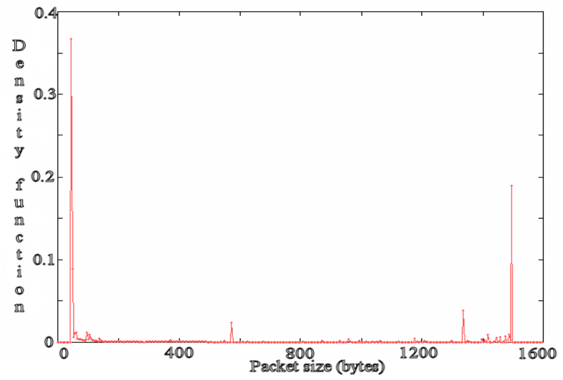

Figure 4: Packet size density function.

For the case of event-driven sampling, we can see in Figure 5 the real parameter bits per second (upper part, 0-10Mbps) and the difference (lower part, under 0 Mbps) obtained in a sampled trace with m=2. Again the granularity of the figure is one second. The difference between real and sampled measurements is the error that we obtain in the sampling process. The error introduced is not significant for the great majority of applications, but in order to compare the different techniques it is preferable to use the ϕ coefficient.

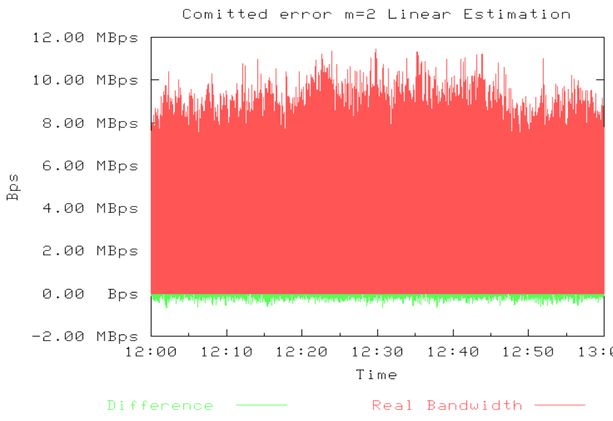

Figure 5: Bps in event-driven sampling and error introduced

The ϕ coefficient for event-driven and different parameter estimators is shown in Figure 6. For low m, the linear estimation is better, but for bigger values of m, any other mean or dynamic estimation is better. For the mean estimation, we show two possible values obtained from all the packets of the hour considered (540 bytes) or all the packets in the trace (1 week, 531 bytes).

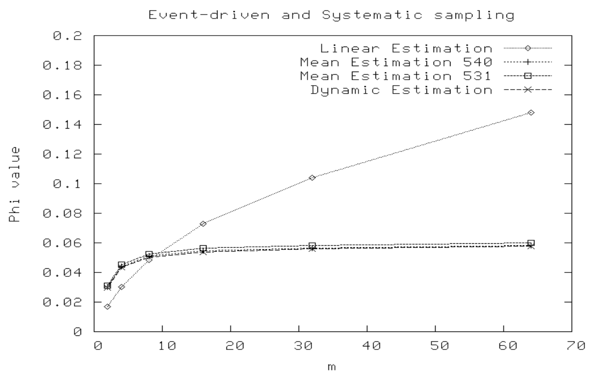

Figure 6: Event-driven with different estimation techniques

For small values of m up to 9, the mean/dynamic estimation is better. This is true because considering a large number of packets m will imply that the mean value will be a better reference for the sum of all packet sizes (m*mean). Those estimations gives a ϕ near constant with m larger than 9, because there is not improvement in the mean estimation.

In order to compare event-driven and time-driven sampling, we will have to compare the sampling factor m of event-driven with interval sizes t of time-driven sampling. They both will have to give the same number of samples for our hour under study. The mapping is shown in Table 1.

Table 1: event-driven (m) and time-driven (t) mapping

m	t
2	1.038 ms
8	4.155 ms
16	8.311 ms

In Figure 7 we can see different parameter estimation for the case of time-driven sampling. If we compare it with Figure 6, event-driven gives better results. Dynamic estimation do as well as mean estimation for event-driven and time-driven.

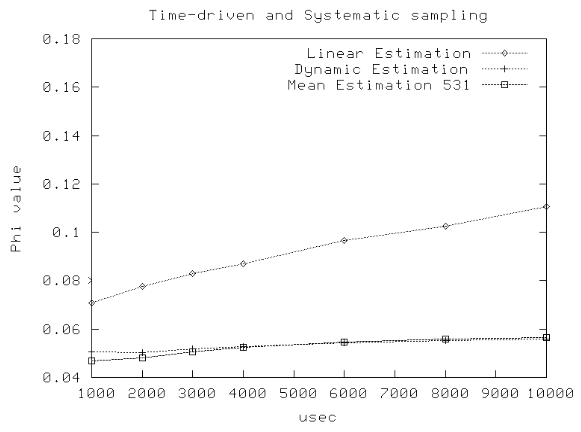

Figure 7: Time-driven with different estimation techniques

In Figure 8 we compare the three sampling methods: event-driven, time-driven and mixed-driven, relating the ϕ coefficient with the number of samples (a number of packets or a time mapped as shown in table 1, for example, a value of 2 means m = 2 or t = 1.038 ms). Time-driven is clearly the worst as concluded in [6] for time independent parameters. However, for the first samples the event-driven shows bad results. Indeed, in that time interval there is big packet size variability that disturbs the packet size estimation. In the same scenario, the proposed mixed-event method outperforms event-driven in this first part. If we analyze other hours from the original trace we obtain similar results whenever we find high variability in packet sizes, something that is very usual in Internet traffic [12]. For the same volume of sampled packets, mixed-driven technique gives best results.

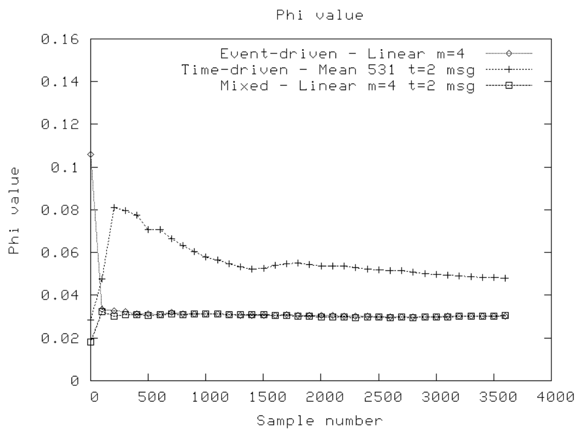

Figure 8: Sampling methods comparative with the number of sample

5. CONCLUSIONS

In this paper we introduce the concept of sampling monitoring on time-dependent parameters, and a new method called mixed-sampling. It is more difficult to apply sampling on these kinds of time-dependent parameters because we need estimation for packets not sampled. Different solutions are proposed, and the results show how dynamic estimation produces best results even for big sampling rate (m) and systematic sample selection.

With the high variability of Internet traffic, mixed-driven technique gives best results for the same volume of sampled packets.

The comparison with other techniques proposed in the literature is not possible due to the novelty of our proposal in sampling application to time-dependent parameters.

6. REFERENCES

[1] Endace Measurement Systems. http://www.endace.com

[2] A Scaleable Monitoring Platform for the Internet (SCAMPI). IST European Project. http://www.ist-scampi.org/

[3] A. Begel, S. McCanne and S.L. Graham. BPF+: Exploiting Global Data-flow Optimization in a Generalized Packet Filter Architecture. In *Proceedings of ACM SIGCOMMM Symposium on Communications Architectures and Protocols,* Harvard University, Cambridge, Massachusetts, September 1999.

[4] I. Kim, J. Moon, and H. Y. Yeom, Timer-Based Interrupt Mitigation for High Performance Packet Processing, *Proceedings of 5th International Conference on High-Performance Computing in the Asia-Pacific Region*, 2001.

[5] Lucas Deri. Improving Passive Packet Capture: Beyond Device Polling. *15th NMRG*, Bremen, Germany, January 2004.

[6] K. Claffy, G. Polyzos, and H-W. Braun. Application of Sampling Methodologies to Network Traffic Characterization. *Proceedings of SIGCOMM '93*, pp. 194-203, San Francisco, September 1993.

[7] N.G. Duffield, C. Lund, M. Thorup. Estimating flow distributions from sampled flow statistics. *ACM Sigcomm 2003*, Karlsruhe, Germany, August 25-29, 2003.

[8] C. Estan and K. Keys and D. Moore and G. Varghese. Building a Better NetFlow. *Proceedings of ACM SIGCOMM 2004*.

[9] Nick Duffield, Carsten Lund, Mikkel Thorup. Flow Sampling under Hard Resource Constraints. *Proceedings the ACM IFIP Conference on Measurement and Modeling of Computer Systems SIGMETRICS/Performance 2004*, pp.85-96, June 2004.

[10] N. Hohn, D. Veitch. Inverting Sampled Traffic. *ACM SIGCOMM Internet Measurement Conference 2003*, Miami Beach, October 27-29, 2003.

[11] Kedar Dhandhere, Hyang-Ah Kim and Tim Jia-Yu Pan. The Application and Effect of Sampling Methods on Collecting Network Traffic Statistics. *Technical Report Carnegie Mellon*, April 28, 2001.

[12] F. Yegenoglu, F. Faris, O. Qadan. A model for representing wide area Internet packet behavior. *Proceeding of the IEEE InternationalPerformance, Computing, and Communications Conference, 2000*. IPCCC '00. pp.167-173, Feb 2000.