A Modular Architecture for Deploying Self-adaptive Traffic Sampling

João Marco C. Silva, Paulo Carvalho, and Solange Rito Lima

Centro Algoritmi, Universidade do Minho, Portugal
{joaomarco,pmc,solange}@di.uminho.pt

Abstract. Traffic sampling is seen as a mandatory solution to cope with the huge amount of traffic traversing network devices. Despite the substantial research work in the area, improving the versatility of adjusting sampling to the wide variety of foreseeable measurement scenarios has not been targeted so far. This motivates the development of an encompassing measurement model based on traffic sampling able to support a large range of network management activities, in a scalable way. The design of this model involves identifying sampling techniques through its components rather than a closed unit, allowing to address issues such as flexibility, estimation accuracy, data overhead and computational weight within a narrower and simpler scope. This paper concretises these ideas presenting a modular and self-configurable measurement architecture based on sampling, a framework implementing sampling inherent pieces, and provides first results when deploying the proposed concepts in real traffic scenarios.

1 Introduction

Performing network measurement tasks in today's networks is a continuous challenge attending to the massive traffic volumes involved, to the wide range of possible monitoring objectives to fulfill, sometimes in a near real-time basis and requiring minimal interference with the normal network operation. Aiming at efficient network measurements, traffic sampling techniques are broadly deployed in strategic network nodes, generically called measurement points (MPs). Their main objective is to select a subset of packets which will then be used to estimate network parameters, avoiding processing all network traffic [12], with the potential cost of affecting measurement accuracy [4].

Despite the substantial research work on packet sampling [9] [8] [13], choosing the best sampling technique depends on traffic characteristics or statistics needed by applications [3]. Moreover, most proposals are focused on specific network measurement tasks, aiming at increasing the accuracy estimation of a single network metric or a small set of metrics, which, in turn, may increase the consumption of computational resources (e.g., CPU, memory and storage capacity).

In this way, the lack of an encompassing traffic sampling architecture able to map a large range of network management measurement needs in a scalable and autonomous way, yet attending to existing computational resources constraints, is evident. Knowing that distinct sampling techniques lead to different computational weight and accuracy levels for each metric estimation, this paper proposes a modular and self-adaptive architecture able to accommodate the selection and configuration of sampling techniques according to the requirements of the network task and resources available.

A. Sperotto et al. (Eds.): AIMS 2014, LNCS 8508, pp. 179–183, 2014.

2 Measurement Architecture

A self-adaptive sampling-based measurement architecture is envisioned as comprising three planes, as illustrated in Figure 1. The *management plane* includes tasks deployed directly in MPs or in external management entities (such as in Software-defined Networking approaches). Based on specific requirements of each network task, measurement needs are identified, one or more MPs are selected, and the most suitable sampling technique is chosen and configured. This also involves identifying an information model able to define managed objects in the network, as suggested in [2].

The management plane is also responsible for providing the self-adaptive behavior of the model. Adaptiveness is ruled by a function which, in runtime, balances estimations accuracy and the corresponding computational weight. Taking a set of thresholds for CPU load, memory and data storage consumption, combined with the expected relative error of the metric estimation, the function is able to determine whether to maintain the current sampling technique (and settings) or to introduce a lighter and/or more accurate technique (based on the ongoing requirements and computational constraints).

The *sampling plane* consists mainly of a modular sampling framework able to assist the deployment of current and future sampling approaches, which fragments the sampling techniques into well-defined components according to sampling *granularity*, *selection scheme* and *selection trigger*. Then, each component is further divided into a set of approaches that, once combined, may allow deploying sampling policies in a flexible and scalable way, as illustrated in Figure 1. In brief, these components are: (i) *granularity* - identifies the atomicity of the element under analysis in the sampling

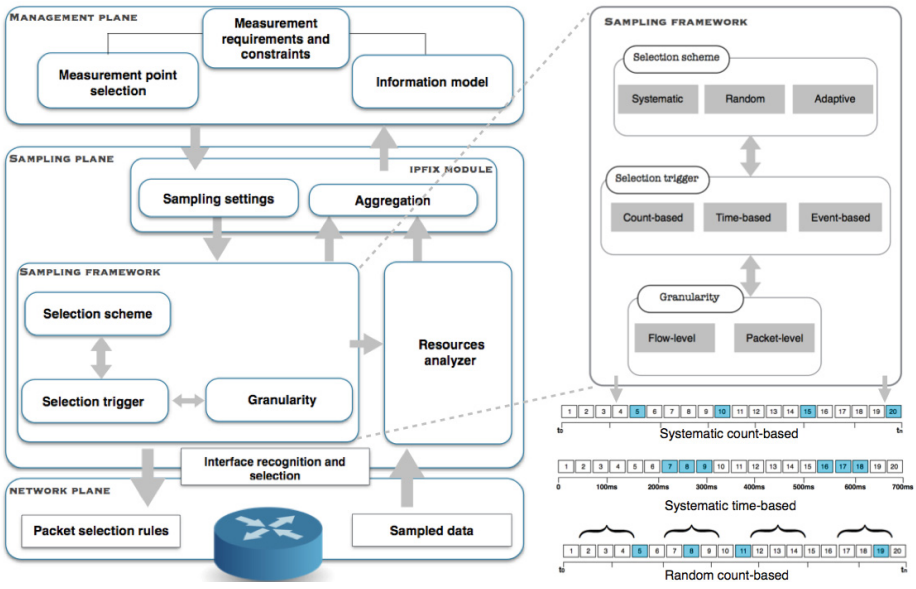


Fig. 1. Architecture description

process: in *flow-level* approach, the sampling process is only applied to packets belonging to a flow or to a set of flows of interest; in a *packet-level* approach, packets are eligible as single independent entities; (ii) *selection scheme* - identifies the function defining which traffic packets will be selected and collected; this scheme may follow a *deterministic*, a *random* or an *adaptive* function; and (iii) *selection trigger* determines the spatial and temporal sample boundaries, may use a *time-based* approach, a *count-based* approach or an *event-based* approach.

To support adaptiveness, a profiler module (*resources analyzer*) monitors the resources consumption in runtime, reporting the current status to the management plane, which will decide on maintaining the sampling policy or setting a new one regarding the measurement needs and resource constraints. A new policy may correspond to a new sampling technique or just a change in the configuration parameters, *e.g.* reducing the sample frequency.

An IPFIX [1] module is responsible for the communication between the management plane and the sampling plane, receiving the sampling settings from the management plane. This module also receives the sampled packets from the network plane, processing them and aggregating relevant fields according to the network task. The aggregation and exporting processes follow IETF guidelines [10] [6] and include results from the resource analyzer, which will then be used for self-adaptation.

At *network plane*, traffic is collected from network interfaces by applying the sample rules defined in the sampling plane. Unprocessed sampled packets are subsequently reported to the sampling plane to be processed, simplifying the network plane.

3 Ongoing Works and Results

According to the measurement architecture, presented in Figure 1, the sampling framework and the resource analyzer were developed in Java using *libpcap*[1], and deployed in a low-cost, open computing device currently used in measurement architectures [11]. The methodology of tests resorts to a quantitative comparison between the computational burden of multiple sampling techniques and policies in presence of similar workloads. The purpose of this comparison is to provide an initial understanding with regards to the relationship between computational requirements and accuracy as afforded by the various sampling techniques. This will facilitate the design of an efficient adaptive module, based on suitable thresholds for specific measurement needs and resource constraints.

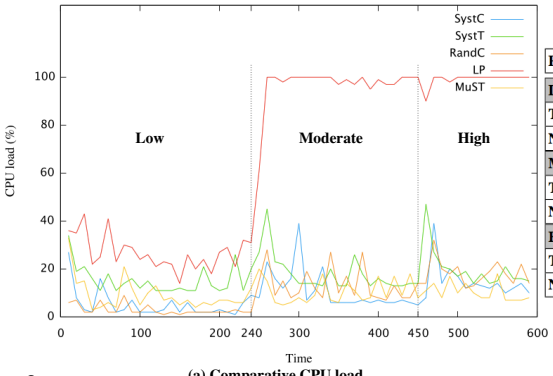
Traffic scenarios consist of three workload periods (low, moderate and high) in the network backbone of University of Minho, Portugal along a typical workday, as shown in Table 1. Due to privacy policies only *https* traffic was collected, then submitted to different sampling techniques deployed in the sampling framework, *i.e.*, SystC - Systematic count-based [12], SystT - Systematic time-based [12], RandC - Random count-based [12], LP - Adaptive linear prediction [5] and MuST - Multiadaptive sampling [7].

As shown in Figure 2(a), SystC and MuST require lower CPU consumption, being MuST less demanding during the most critical work scenario. Regarding memory usage, Figure 2(b) demonstrates similar behavior across all techniques and workload scenarios. The volume of data involved in the sampling process (see Figure 2(c)) is higher

[1] `http://www.tcpdump.org/`

for the SystT technique, contrarily to SystC and RandC. Assuming that the computational resource consumption of systematic techniques is proportional to the sampling frequency, Figure 2(d) presents the mean resource consumption for distinct sampling frequencies for SystC technique when applied to the high workload scenario.

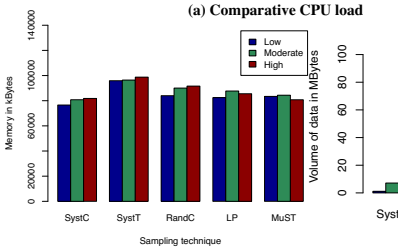
Despite the importance of reducing the computational burden of traffic sampling, sampling techniques must still be able to represent the network behavior accurately. Table 1 presents the accuracy results per technique when estimating two metrics useful for traffic characterization, such as mean throughput and number of flows. The results show that the higher storage requirement of SystT delivers a better accuracy in flow identification. Regarding throughput, the relative mean error (RME) is low for all scenarios and techniques (less than 10%). Exceptions are: (i) MuST technique applied to high workload, achieving a significant low error (less than 1%); and (ii) RandC technique applied to moderate workload, with a relative error above 10%.



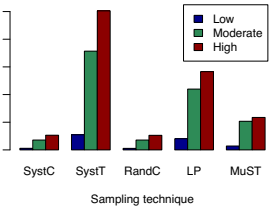
(a) Comparative CPU load

Table 1. Accuracy in metric estimation

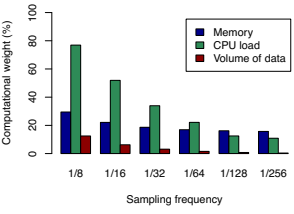
Parameter	SystC	SystT	RandC	LP	MuST
Low workload $\approx 5Mbps$ ‖ Total of flows = 2187					
Throughput RME	0.018	0.049	0.059	0.011	0.021
Number of flows	486	1106	515	869	454
Moderate workload $\approx 25Mbps$ ‖ Total of flows = 16069					
Throughput RME	0.046	0.059	0.11	0.042	0.036
Number of flows	3885	8963	4081	6092	4689
High workload $\approx 70Mbps$ ‖ Total of flows = 33577					
Throughput RME	0.047	0.068	0.054	0.065	0.004
Number of flows	6817	16719	6835	10455	4916



(b) Comparative memory usage



(c) Comparative volume of data



(d) SystC resources usage

Fig. 2. Overall results

These preliminary results evince the relevance of tuning traffic sampling (choosing technique, configuration parameters and thresholds), in order to meet distinct measurement requirements and constraints. The present research work proposing a modular and self-adaptive measurement architecture will allow enlarging the scope and efficiency of network measurement tasks through traffic sampling.

Acknowledgements. This work has been supported by FCT - *Fundação para a Ciência e Tecnologia* in the scope of the project: PEst-OE/EEI/UI0319/2014.

References

1. Boschi, E., Mark, L., Quittek, J., Stiemerling, M., Aitken, P.: IP Flow Information Export (IPFIX) Implementation Guidelines - RFC5153. Tech. rep., IETF (2008), https://datatracker.ietf.org/doc/rfc5153/
2. Dietz, T., Claise, B., Quittek, J.: Definitions of Managed Objects for Packet Sampling - RFC6727. Tech. rep., IETF (2013), https://datatracker.ietf.org/doc/rfc6727/
3. Duffield, N.: Sampling for Passive Internet Measurement: A Review. Statistical Science 19(3), 472–498 (2004)
4. Estan, C., Varghese, G.: New directions in traffic measurement and accounting. SIGCOMM Comput. Commun. Rev. 32(4), 323–336 (2002), http://dl.acm.org/citation.cfm?id=964725.633056, http://doi.acm.org/10.1145/964725.633056
5. Hernandez, E.A., Chidester, M.C., George, A.D.: Adaptive Sampling for Network Management. Journal of Network and Systems Management 9(4), 409–434 (2001), http://dx.doi.org/10.1023/A:1012980307500
6. Muenz, G., Claise, B., Aitken, P.: Configuration Data Model for the IP Flow Information Export (IPFIX) and Packet Sampling (PSAMP) Protocols - RFC6728. Tech. rep., IETF RFC 6728 (2012), http://datatracker.ietf.org/doc/rfc6728/?include_text=1
7. Silva, J.M.C., Carvalho, P., Rito Lima, S.: A multiadaptive sampling technique for cost-effective network measurements. Computer Networks (2013), http://www.sciencedirect.com/science/article/pii/S1389128613002491
8. Sommers, J., Barford, P., Duffield, N., Ron, A.: Improving accuracy in end-to-end packet loss measurement. In: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications - SIGCOMM 2005, vol. 35, p. 157. ACM Press, New York (2005), http://dl.acm.org/citation.cfm?id=1080091.1080111
9. Tammaro, D., Valenti, S., Rossi, D., Pescapé, A.: Exploiting packet-sampling measurements for traffic characterization and classification. International Journal of Network Management 22(6), 451–476 (2012), http://doi.wiley.com/10.1002/nem.1802
10. Trammell, B., Wagner, A., Claise, B.: Flow Aggregation for the IP Flow Information Export (IPFIX) Protocol - RFC7015. Tech. rep., IETF (2013)
11. Young, H.: Archipelago measurement infrastructure, http://www.caida.org/projects/ark/
12. Zseby, T., Molina, M., Duffield, N.: Sampling and Filtering Techniques for IP Packet Selection RFC 5475. Tech. rep. (2009), http://datatracker.ietf.org/doc/rfc5475/
13. Zseby, T., Hirsch, T., Claise, B.: Packet Sampling for Flow Accounting: Challenges and Limitations. In: Claypool, M., Uhlig, S. (eds.) PAM 2008. LNCS, vol. 4979, pp. 61–71. Springer, Heidelberg (2008), http://dx.doi.org/10.1007/978-3-540-79232-1_7