Cheapo: An Algorithm for runtime adaption of time intervals applied in 5G Networks

Marios Touloupou, Evgenia Kapassa, Athanasios Kiourtis, Dimosthenis Kyriazis
Department of Digital Systems
University of Piraeus
Piraeus, Greece
{mtouloup, ekapassa, kiourtis, dimos}@unipi.gr

Abstract— **The explosion of mobile devices and content, server virtualization, and advent of cloud services are among the trends driving the networking industry to re-examine traditional network architectures. Software-Defined Networking (SDN) is an emerging architecture that is dynamic, manageable, cost-effective, and adaptable, making it ideal for the high-bandwidth, dynamic nature of today's applications. Currently, it continues to be crucial for businesses to monitor their networks in order to be productive and avoid serious threats from network failures and server downtime, demanding accuracy and timeliness. In this paper, we focus on this trade-off between monitoring accuracy and timeliness. We propose Cheapo – an adaptive algorithm for monitoring frameworks in SDN, which provides an API for flow statistics collection at different aggregation levels. The proposed approach provides highly accurate information without producing unnecessary network traffic.**

Keywords— NFV/SDN; 5G Networks; Monitoring; Network Services; Cost-efficient Monitoring

I. INTRODUCTION

As networks evolve and become more complex and advanced, the need for network monitoring is remaining constant to drive decisions for network management and optimization. As stated in [1], it continues to be crucial for businesses to monitor their networks in order to be productive and avoid serious threats from network failures and server downtime. As more and more businesses deploy networking to support various business activities, and as traffic generated by human-operated devices becomes dwarfed by massive growth in data flowing to and from IoT devices, network environments will become much more fluid. At the same time, the importance of networks will grow tremendously [24] — in fact, any digitally transformed business will need to achieve excellence in networking that today is achieved mostly in verticals that revolve around data processing and manipulation.

Generally, network monitoring can analyze different facts such as performance in real-time [25], meaning that if a failure or issue is detected, alerts are triggered in runtime. This rapid relay of information means that tools and mechanisms are able to act in real-time to adapt the network resources accordingly in order to take corrective actions, thus minimizing potential downtime. In addition, network monitoring eliminates the need for a physical system administrator and manual checks [22]. This can result for a company into savings for both time and expenses, since the problem is addressed into an effective way. Another major benefit is the reporting generated from network monitoring. Such reports can help in the identification of patterns and trends in system performance, as well as in the demonstration of the requirements into upgrades or replacements. What is more, in such cases, performance baselines can also be easily established. To this end, network monitoring frameworks can assist in the identification of specific areas of a network that are facing problems [23]. In that case, it becomes easy to pinpoint the issue, resulting into cost and time savings in cases that the problem needs to be addressed.

In recent years, Software Defined Networking (SDN) has emerged with the promise to facilitate network programmability and to ease the management tasks. Accurate and timely statistics are essential for multiple network management tasks, like load balancing, traffic engineering, Service Level Agreement (SLA) enforcement, accounting and intrusion detection [3]. Furthermore, management applications may need to monitor network resources at different aggregation levels. In SDN architectures, monitoring information is a key factor for management and orchestration. The challenge is to minimize the footprint of monitoring frameworks and the overhead of monitoring data collection. However, decisions regarding adaptable and dynamic monitoring should be made without any potential miss of information that could affect runtime decisions.

To address these challenges, in this paper we propose *Cheapo*, an adaptive algorithm that is applicable to monitoring frameworks in SDNs. Cheapo has been used and validated in a current well-established monitoring framework, Prometheus [10] is used in "SONATA NFV: Agile Service Development and Orchestration in 5G Virtualized Networks" project [4]. The purpose of this project is to support the management of 5G services under the SDN/NFV paradigm. Cheapo, comes to introduce smart decision logic on probe level (i.e. monitoring data collection level). In other words, through Cheapo it will be possible to take decisions (such as reducing the period of a measurement) or calculating the time interval, on the probe level, in order to reduce unnecessary traffic and make the network more cost-efficient.

The rest of this paper is organized as follows. Section II presents the related work on the field of monitoring in Internet Protocol (IP) networks as well as in SDNs. Section III introduces the proposed solution with initial evaluation results, followed by explanatory figures describing its architecture, while Section IV describes our conclusions and future plans.

II. Related Work

Currently, there are multiple flow based network monitoring tools for SDNs. This section provides a short description of some approaches developed and performed into SDNs, which can also be included into cloud computing services.

OpenFlow [6] is a protocol used in SDNs for communication between a network device and a controller server. This particular architecture separates packet forwarding and high-level routing decisions, whereby packet forwarding is performed by the switch hardware and high-level routing decisions are performed by a controller server. What is more, it should be noted that OpenFlow has emerged as the de facto standard for communication between the controller and the switches into an SDN-based infrastructure. cAdvisor [9] (i.e. container Advisor) provides container users an understanding of the resource usage and performance characteristics of their running containers. It is a running daemon that collects, aggregates, processes, and exports information about running containers. More specifically, it is able to keep resource isolation parameters for each container, as well as historical resource usage, histograms of complete historical resource usage and network statistics. JFlow [7] is a Juniper Networks proprietary flow monitoring implementation. Juniper Networks® J Series Services Routers and SRX Series Services Gateways generate summarized flow records for sampled packets from the Packet Forwarding Engine (PFE). Such flow records are exported in an RFC or NetFlow-compliant standard packet format to an external flow information collector. JFlow is interoperable with any NetFlow supported flow collector, and as a result of that the external flow collector can be any third-party software that collects data exported from Juniper Networks devices. In the data plane functionality, OpenSketch [8] provides a simple three-stage pipeline (including hashing, filtering, and counting), which can be implemented with commodity switch components and support many measurement tasks. On the other hand, in the control plane functionality, OpenSketch provides a measurement library that automatically configures the pipeline and allocates resources for different measurement tasks. Prometheus [10] is an open-source system monitoring and alerting toolkit originally built at SoundCloud, that joined the Cloud Native Computing Foundation in 2016 as the second hosted project after Kubernetes [26]. Among the main features of Prometheus are that it provides a multi-dimensional data model (time series identified by metric name and key/value pairs), including a flexible query language to leverage this dimensionality. Furthermore, there exist no reliance on distributed storage, since single server nodes are autonomous, while time series collection happens via a pull model over HTTP. It should be added that with regards to the pushing time series, they are supported through an intermediary gateway, while targets are discovered through service discovery or static configuration, including multiple modes of graphing and dashboarding support.

In general, there has been an *eternal shrinkage* [20] between the accuracy of statistical data collection and the use of resources for monitoring in networks generally. What is more, monitoring in SDNs should also help to offset general resource costs and measurement accuracy as discussed by the authors in [13]. Variable-rate adaptive sampling techniques have been proposed in different contexts [28] to improve resource consumption, while providing satisfactory levels of accuracy of the collected data. Such techniques for saving resources while attaining a higher percentage of precision have been extensively discussed in the literature on environmental sensor networks [14], [15], [16], [17], [18], [19]. The main focus of these sampling techniques is to efficiently collect data using the sensor while trying to minimize the power consumption of the latter, which is often considered as a rare source for the sensors. Adaptive sampling techniques have also been studied in the context of traditional IP networks. However, as far as we know, adaptive sampling for SDN tracking has not been explored yet.

One should have in mind that the exchange between resource use and accuracy [12] have different expressivity, whereas each one of them is suitable for actions with different spatial and temporal properties. In other words, a given measurement job can be achieved with different accuracy using different prototypes. From our point of view, if SDNs supported all three measurement artefacts [29], an interesting challenge for SDN would be to manage different resources in multiple parallel tasks. Typically, the resource allocated to a particular measurement task (e.g. TCAM for meters, bandwidth limits for transmitting measurements to the controller) limits the accuracy of the results.

Recently, the authors in [11] proposed an adaptive SDN-based monitoring method, focusing on the detection of abnormalities. In short, they manage to switch the aggregation level of packet forwarding rules between different grading levels of traffic measurement.

In our case, we propose Cheapo, an adaptive variable rate sampling algorithm that makes a time scale adjustment, that will be built on top of SONATA and Prometheus Monitoring Framework. In a few words, into a SDN's point of presence (POP), VNFs [27] are running in the form of services. In order for the monitoring framework to correctly perform its task, VNFs must collect their data at any way and push them towards another agent.

After our research, we realized that a static frequency time interval gives high accuracy data but produce unnecessary traffic at the same time. In that case, Cheapo comes to replace the static variable with an adoptable algorithm that will continuously adapt the time interval according to the network state, in order to keep a balance between the accuracy of the data and the traffic inside the network.

III. PROPOSED APPROACH

A. Baseline Approach

The proposed approach is not coupled to a specific monitoring framework but can be applied to any monitoring mechanism. It aims at adapting during runtime the monitoring time intervals in order to ensure that the data collected and transmitted to other architectural components for runtime decision making are data that actually drive decisions and not all raw data. The internal architecture of SONATA Monitoring Framework enhanced with the proposed algorithm (i.e. Frequency Analyzer) is cited in the following figure (Fig. 1).

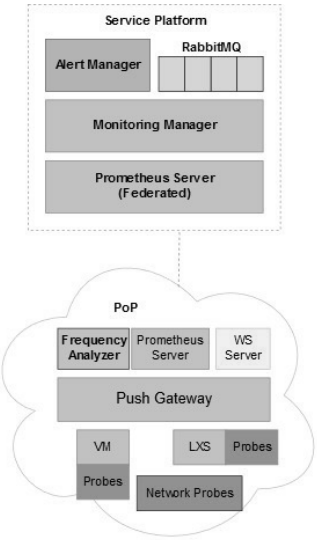

Fig. 1 Monitoring Framework high-level architecture

B. The Downside of a Static Threshold

As we can realize, probes, collect data from the running services and are then pushing its data towards the *Push Gateway*. The time interval for the probes to push its data towards the *Push Gateway* is usually a static variable/rule set by the service developer or the monitoring framework itself. For instance, if a metric value riches a specific threshold, the probe pushes data towards the *Push Gateway*.

The aforementioned approach has several disadvantages. For example, if thresholds are set too high, critical information might be missed or identified late resulting to potential service degradations or outages. On the other hand, if these thresholds are set too low, the volume of alerts can be overwhelming. In the proposed approach, we aim at removing the "static" approach in monitoring through a runtime adaptable scheduling algorithm that proposes time intervals based on the obtained monitoring information.

C. An Adaptive Monitoring Algorithm

As we previously mentioned, the proposed approach is not coupled to a specific monitoring framework. In other words, we propose that probes are due to collect data with linear increase of the time. Supposing that every 1 sec, probes are storing the data collected to a local storage and they compare the values with the data collected on the next 1 sec. In the case that the difference of their values is more than a given threshold, then we can realize that there was a change in our network (i.e. "bad" change, we may expect for a bandwidth value of 20Mbps but only 10Mbps return at the probe). In such a case, we should allow the monitor manager to be informed about that change. Consequently, probes are sending their values to the *Push Gateway* taking note of the time that the specific data change occurred. Afterwards, probes must *divide* the time the change happened with a specific number α, resulting into the so-called *Timeout*. By the time that the probes will start re-capturing the data, they will now have information that they will collect and compare with a linear time interval until that specific *Timeout*. After the *Timeout* period ends, probes are due to collect and compare data with exponential time interval. Fig. 2 shows the time's change after a "bad" change to the network.

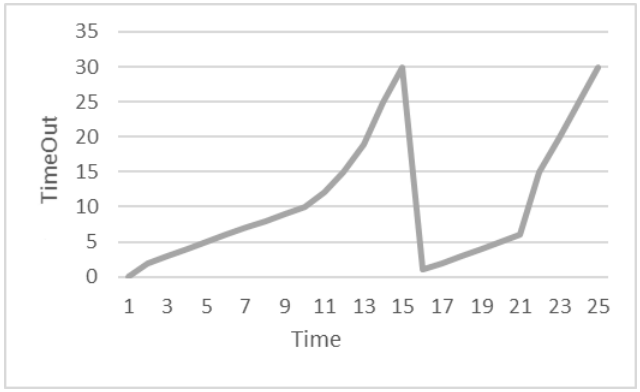

Fig. 2. Timeout Adjustment (Time/α)

As it is seen in Fig. 2, the X axis shows the linear increase of the time that probes collect data. The latter starts from the 1st sec and increases linearly, until it riches the 10th sec, and then continues with exponential increase of the time. In parallel, the algorithm compares the Delta of the gathered data at the time *t* and *t+1*. At the 15th sec., a significant change (i.e. big Delta) on the value of the parameter collected by the probes was occurred. We assume that instead of the expected 20Mbps Bandwidth, was returned a value of 10Mbps at the probe. This means, that an error has occurred (i.e. "bad" change) concerning the network's behavior. Consequently, probes will transmit their data towards the push gateway, setting the new *Timeout* at (15 / α). In that case, the equation returns the value of 5. As a result, the probe knows that for a specific period of time, data must be collected with a linear increase of time, with a duration of 5 sec. Afterwards, it must continue the data collection with an exponential increase of time, as shown in Fig. 2, until it riches the 25th sec – max timeout set by the start of the process.

However, there is also the case of a "good" change (i.e. we may expect for a Bandwidth value of 10Mbps but 20Mbps might return at the probe). In such a case, probes are using the same tactic of collecting the data. Shortly, they start collecting and comparing data with linear time interval until they rich the *Timeout*. In the case that the difference of their values is less

than a given threshold, then we realize that there was a change in our network (i.e. "good" change). In that case, we should also let the monitor manager know about that change. The main difference is that we can realize that the network status is in "good" shape. Consequently, after the probes provide their data to the monitoring manager, they *multiply* the time the change happened with a specific number β (Fig. 3). As a result, the next *Timeout* that the probes will be collecting data with a linear increase of the time will be more than the previous one. Afterwards, the same approach is used for the next decisions taken by the probes. It should be clear that in that case, the probes are collecting and transmitting their data with a chance of continuously changing the value of the *Timeout* value.

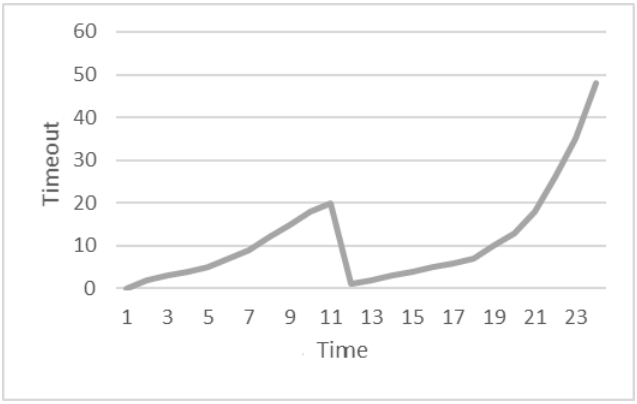

Fig. 3. Timeout Adjustment (Time*β)

A similar approach is described in Fig. 3. In that case the X axis shows the linear increase of the time that probes collect data. The latter starts from the 1st sec and increases linearly until it riches the 5th sec. Consequently, at the 5th sec, a change of a meter occurs. The probe will transmit its data towards the push gateway, increasing the *Timeout* by (Current time * β). In this scenario, the equation returns the value of 18. That means that the probe will continue the data collection following a linear increase of the time until the new *Timeout* (i.e. 18th sec.). In the case that there is no change by this time, probe will continue the data collection with exponential increase of the time interval until it reaches the max timeout variable. Afterwards, the whole process starts from the beginning using the very first timeouts set by the monitoring framework.

IV. Coclusions and future work

In this paper, we have introduced Cheapo – a scheduling algorithm for monitoring frameworks in SDN. Shortly, Cheapo has been described as an adaptable algorithm which can be affiliated by any monitoring framework in order to prevent any unnecessary traffic to the network, but can also provide accuracy of the monitoring data and make the network cost-efficient at the same time.

In general, we have identified the rationale behind this *Timeout* adjustment is that we maintain a higher pushing frequency for data that significantly contribute to link utilization, while we maintain a lower pushing frequency for data that do not significantly contribute towards link utilization at that moment.

As a future work, and in the frame of 5GTANGO EU-funded project [21] the described approach will be adapted and further enhanced in order to facilitate decision making with respect to the data that is transmitted, addressing security and privacy constraints.

Acknowledgment

This work has been partially supported by the 5GTANGO project, funded by the European Commission under Grant number H2020ICT-2016-2 761493 through the Horizon 2020 and 5G-PPP programs (http://5gtango.eu).

References

[1] "The Importance of Monitoring Your Networks", https://www.opsview.com/resources/network/blog/importance-monitoring-your-networks

[2] "The Importance of Network Monitoring", https://itnow.net/the-importance-of-network-monitoring/

[3] S.R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, "Payless: A low cost network monitoring framework for software defined networks", In Network Operations and Management Symposium (NOMS), pp. 1-9, 2014 IEEE

[4] "SONATA NFV: Agile Service Development and Orchestration in 5G Virtualized Networks", http://www.sonata-nfv.eu/

[5] P. Trakadas, P. Karkazis, HC. Leligou, T. Zahariadis, S. Kolometsos, "Implementation of a Monitoring Framework to Support the Management of 5G Services" [under press]

[6] "Open Datapath", https://www.opennetworking.org/sdn-resources/openflow

[7] A. C. Myers, "JFlow: Practical mostly-static information flow control," in Proceedings of the 26th ACM SIGPLAN-SIGACT symposium on Principles of programming languages. ACM, 1999, pp. 228–241

[8] M. Yu, L. Jose, and R. Miao, "Software defined traffic measurement with opensketch," in Proceedings 10th USENIX Symposium on Networked Systems Design and Implementation, NSDI, vol. 13, 2013.

[9] "Cadvisor", https://github.com/google/cadvisor

[10] "From metrics to insight", https://prometheus.io/

[11] Y. Zhang, "An adaptive flow counting method for anomaly detection in sdn," in Proceedings of the ninth ACM conference on Emerging networking experiments and technologies. ACM, 2013, pp. 25–30

[12] M. Moshref, M. Yu, & R. Govindan, Resource/accuracy tradeoffs in software-defined measurement. In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking (pp. 73-78). ACM, 2013, August.

[13] M. Moshref, M. Yu, and R. Govindan, "Resource/Accuracy Tradeoffs in Software-Defined Measurement," in Proceedings of HotSDN 2013, August 2013, to appear

[14] A. Jain and E. Y. Chang, "Adaptive sampling for sensor networks," in Proceeedings of the 1st international workshop on Data management for sensor networks: in conjunction with VLDB 2004. ACM, 2004, pp. 10–16.

[15] B. Gedik, L. Liu, and P. Yu, "Asap: An adaptive sampling approach to data collection in sensor networks," Parallel and Distributed Systems, IEEE Transactions on, vol. 18, no. 12, pp. 1766–1783, 2007.

[16] A. D. Marbini and L. E. Sacks, "Adaptive sampling mechanisms in sensor networks," in London Communications Symposium, 2003.

[17] J. Kho, A. Rogers, and N. R. Jennings, "Decentralized control of adaptive sampling in wireless sensor networks," ACM Transactions on Sensor Networks (TOSN), vol. 5, no. 3, p. 19, 2009.

[18] C. Alippi, G. Anastasi, M. Di Francesco, and M. Roveri, "An adaptive sampling algorithm for effective energy management in wireless sensor networks with energy-hungry sensors," Instrumentation and Measurement, IEEE Transactions on, vol. 59, no. 2, pp. 335–344, 2010.

[19] R. Willett, A. Martin, and R. Nowak, "Backcasting: adaptive sampling for sensor networks," in Information Processing in Sensor Networks,

2004. IPSN 2004. Third International Symposium on, 2004, pp. 124–133.

[20] R. S. Chowdhury, M. F. Bari, R. Ahmed, & R. Boutaba, Payless: A low cost network monitoring framework for software defined networks. In Network Operations and Management Symposium (NOMS), 2014 IEEE (pp. 1-9). IEEE, May 2014

[21] "5G DEVELOPMENT AND VALIDATION PLATFORM FOR GLOBAL INDUSTRY-SPECIFIC NETWORK SERVICES AND APPS" http://5gtango.eu/

[22] "Basics of Network Monitoring" http://www.solarwinds.com/basics-of-network-monitoring

[23] "Common Home Network Problems" https://www.lifewire.com/top-home-networking-problems-and-mistakes-817736

[24] "Networks 2020" https://www.avaya.com/en/documents/networks_2020_idc_report_42291317.pdf?t=0

[25] "Why Network Monitoring?" https://www.spiceworks.com/it-articles/why-network-monitoring/

[26] "Production-Grade Container Orchestration" https://kubernetes.io/

[27] "NFV vs. VNF: What's the difference?" http://searchsdn.techtarget.com/answer/NFV-vs-VNF-Whats-the-difference

[28] J. Wang, Ji, X., Zhao, S., Xie, X., & J. Kuang. Context-based adaptive arithmetic coding in time and frequency domain for the lossless compression of audio coding parameters at variable rate. EURASIP Journal on Audio, Speech, and Music Processing, 2013(1), 9.

[29] A. Yassine., H. Rahimi, & S. Shirmohammadi,. Software defined network traffic measurement: Current trends and challenges. IEEE Instrumentation & Measurement Magazine, 18(2), 42-50, 2015.