Computer Networks 63 (2014) 68–83

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

# An instrumentation framework for the critical task of measurement collection in the future Internet

Olivier Mehani *, Guillaume Jourjon, Thierry Rakotoarivelo, Max Ott

*NICTA, 13 Garden Street, 2015 Eveleigh, NSW, Australia*

ARTICLE INFO

*Article history:*
Received 1 July 2012
Received in revised form 30 December 2013
Accepted 8 January 2014
Available online 15 January 2014

*Keywords:*
Network measurements
Instrumentation system
Software-Defined Measurement
Performance evaluation
OML

ABSTRACT

Experimental research on future Internet technologies involves observing multiple metrics at various distributed points of the networks under study. Collecting these measurements is often a tedious, repetitive and error prone task, be it in a testbed or in an uncontrolled field experiment. The relevant experimental data is usually scattered across multiple hosts in potentially different formats, and sometimes buried amongst a trove of other measurements, irrelevant to the current study. Collecting, selecting and formatting the useful measurements is a time-consuming and error-prone manual operation.

In this paper, we present a conceptual Software-Defined Measurement (SDM) framework to facilitate this task. It includes a common representation for any type of experimental data, as well as the elements to process and collect the measurement samples and their associated metadata. We then present an implementation of this concept, which we built as a major extension and refactoring of the existing Orbit Measurement Library (OML). We outline its API, and how it can be used to instrument an experiment in only a few lines of code. We also evaluate the current implementation, and demonstrate that it efficiently allows measurement collection without interfering with the systems under observation.



## 1. Introduction

In his seminal 2004 paper [1], Paxson presented several good-practice guidelines on the collection, annotation and storage of experimental data and identified their critical role in sound Internet experiments. Since multiple categories of observations are collected using different tools with varying accuracy, precision, and disruption characteristics, the creation of good quality data under these guidelines often requires tedious manual and *ad hoc* post-processing. Moreover, as these tools rarely share common storage formats, further non-negligible data manipulation tasks are required before any analysis, or later reuse in other scientific studies, can be done.

The problem of collecting and reporting data becomes paramount for networking experiments and distributed applications, as they usually involve multiple entities. Furthermore, as research in future Internet technologies moves towards Software-Defined Networking (SDN) [2] and everything-as-a-service [3], such data collection processes needs to operate in highly dynamic environments where experimental parameters and performance metrics, as well as their sources and destinations—in essence the full reporting chain—may change rapidly. Yet, the observation of these environments often relies on special-purpose tools with often limited flexibility. Nonetheless, some recent works have demonstrated the value of an ability to correlate and analyse different types of measurements from various sources in, *e.g.*, investigating the root cause of observed network issues [4,5].

Additionally, sampling and reporting have a cost. As future Internet systems become increasingly more mobile and heterogeneous, the collection of measurement data

* Corresponding author. Tel.: +61 293762167.
*E-mail addresses:* olivier.mehani@nicta.com.au (O. Mehani), guillaume.jourjon@nicta.com.au (G. Jourjon), thierry.rakotoarivelo@nicta.com.au (T. Rakotoarivelo), max.ott@nicta.com.au (M. Ott).

1389-1286/$ - see front matter © 2014 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.bjp.2014.01.007

might require significant network resources at a given time. While various workaround solutions are possible, such as local buffering of measurements, adjustment of their sampling rates, or aggregation of multiple samples into summary statistics (*e.g.*, [6]), they require some decision process on the measurement probe side to adequately adapt to the currently observed conditions of the collection chain.

A common framework for data collection and reporting is clearly needed to support measurement in future Internet research. Such a framework would provide a generic foundation for *Software-Defined Measurements* (SDM). Moreover, we believe it should cater for the following requirements, which extends the good-practice guidelines from Paxson's paper [1].

(Req. 1) known, ideally minimal, systematic bias in the data collection (*i.e.*, no side effect introduced by the measurement);
(Req. 2) robust timestamping across nodes (*i.e.*, time comparisons between samples from different sources should be meaningful);
(Req. 3) rich and documented storage format (*i.e.*, supporting the best practices of [1]);
(Req. 4) ability to integrate legacy measurement tools and formats (*i.e.*, leverage existing and well-understood utilities rather than force a clean slate);
(Req. 5) domain-agnostic support (*i.e.*, not only network or system metrics);
(Req. 6) usability in distributed systems (*i.e.*, data coming from multiple points can be processed, and/or stored, on many others);
(Req. 7) reusable measurements (*i.e.*, samples collected for one purpose should be accessible for other analyses without a second observation);
(Req. 8) dynamically reconfigurable (*i.e.*, feedback loops should allow the reporting chain to adjust its own parameters based on currently observed conditions).

The Orbit Measurement Library (OML) [7] was released in 2005 for the sole purpose of instrumenting wireless experiments and reporting measurements into a single central database. We took over the development of OML from its original team and have evolved it into a generic framework no longer limited to network characterisation. More precisely, we have made extensive changes to this library (>90% of the code base) to cater for most of the previously mentioned requirements (namely, Req. 1–6). We believe that the resulting OML2[1] software is a good candidate for an SDM framework.

The objective of this article is manifold. First we propose a generic architecture towards which we believe an SDM framework should tend. We also provide a comprehensive overview of the OML design towards implementing this SDM architecture. Second, we offer an experimental characterisation of the bias that OML might introduce to systems under study, thus evaluating its compliance to Req. 1. This allows us to derive a set of guidelines to ensure that they do not reach a statistically significant level.

The remainder of this article is organised as follows. Section 2 presents work related to measurement and collection frameworks for experimental studies. In Section 3, we present our proposed conceptual framework, as well as its implementation into OML. We also review the API of this suite, and highlight the steps needed to create an instrumented application. We then evaluate OML's "observer effect" in Section 4 and derive usage guidelines to minimise or remove it when present. Finally, Section 5 concludes this article and highlights future directions towards a full implementation of the proposed framework.

## 2. Related work

A distributed monitoring framework usually comprises three generic elements: probes performing the actual measurement data collection, formats and protocols allowing for storage and exchange of the measurements, and tools to process, forward and store the samples. In this section, we review the state of art for these different elements. A summary of this discussion in regards to each tool's compliance with SDM requirements defined in Section 1 can be found in Table 1.

### 2.1. Data collection tools

The networking community has been developing and using several types of stand-alone measurement tools dedicated to a specific task, such as `tcpdump`(1), D-ITG [9,10] or Iperf [8]. The former has been shown to accurately report at capture rates up to gigabits per second [26] while the latter allows researchers to generate a traffic load to evaluate the capacity of a network or the resilience of a system. In particular the authors of [27] showed that Iperf generated the highest load on network paths compared to other traffic generators. High performance or versatile hardware solutions have also been developed, such as DAG,[2] but usually store the data locally and thus do not comply with Req. 6.

As we noted in the introduction, one common problem with these tools is that they do not share output formats, and post-processing is required before being able to cross-analyse their data, thus hindering Req. 4 and 7. The problem of data collection from distributed nodes (Req. 6) is also not addressed at this level. Additionally, there is little or no study characterising potential biases or impact of using these tools on the system under study (Req. 1). In particular, we show in Section 4 that under certain conditions, using OML to report Iperf's results rather than its standard CSV output significantly increases the achievable throughput.

Several solutions exist to support more generic data collection (Req. 5), allowing for instrumentation and metrics collection from various networking applications and devices. A few system monitoring tools such as Zabbix [12]

[1] In the remainder of this article, we simply use "OML" to refer to this suite, available at http://oml.mytestbed.net.

[2] http://www.endace.com.