Computer Networks 107 (2016) 104–120

Contents lists available at ScienceDirect

Computer Networks

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

Multidimensional cloud latency monitoring and evaluation

Ondrej Tomanek*, Pavol Mulinka, Lukas Kencl

Department of Telecommunications Engineering, Czech Technical University In Prague, Technicka 2, Prague, Czech Republic

ARTICLE INFO

Article history:
Received 27 November 2015
Revised 10 June 2016
Accepted 11 June 2016
Available online 14 June 2016

Keywords:
Multidimensional
Distributed
Cross-layer
Cloud computing
Network latency
Monitoring

ABSTRACT

Measuring or evaluating performance of a Cloud service is a non-trivial and highly ambiguous task. We focus on Cloud-service latency from the user's point of view, and, to this end, utilize the multidimensional latency measurements obtained using an in-house designed active-probing platform, CLAudit, deployed across PlanetLab and Microsoft Azure datacenters. The multiple geographic Vantage Points, multiple protocol layers and multiple datacenter locations of CLAudit measurements allow us to pinpoint with great precision if, where and what kind of a particular latency-generating event has happened. We analyze and interpret measurements over two one-month time-intervals, one in 2013 and one in 2016. As these traces are large, an automated interpretation has been appended to the data-capture process. In summary, we demonstrate the utility of the multidimensional approach and document the differences in the measured Cloud-services latency over time. Our measurements data is publicly available and we encourage the research community to use it for verification and further studies.

1. Introduction

The increasing reliance on Cloud Computing within the field of information and communication technologies, together with the lack of technical information disclosed, poses serious challenges to Cloud tenants and end-users, left on their own to monitor what they pay for. One-dimensional monitoring approach, i.e. deploying a single Vantage Point talking to an arbitrary Cloud target via a single protocol, only provides limited insights. For example, one might not be able to discover the cause of a network failure or identify path segments impacting the Cloud-service performance. These issues are amplified by the ever-increasing Internet path diversity, by the complexity of traffic patterns and by the number of devices involved.

In our work we focus on *multidimensional* monitoring of Cloud-service *latency*. We choose latency because of its direct impact on overall performance, and, consequently, on the end-user experience. Latency is difficult to model, predict or control, but measuring latency may provide a lot of useful information. Latency data can be relatively easily derived from communication flows. Methods of capture, analysis and interpretation of such measurements, such as the one presented in this paper, ought to be then especially useful to Cloud tenants, who may thus be able to overturn the aforementioned information deficit, even without any privileged access to the Cloud infrastructure.

The problem we thus address is rigorous monitoring and advantageous interpretation of Cloud-Service latency behavior. We build on our two previous works that focused on architecting a multidimensional Cloud Latency Auditing platform (CLAudit) [1], used to identify issues like Data Center (DC) failures or routing pathologies. Detection of such anomalous events is automated by inter-relating the measurements time-series across different dimensions [2].

In this paper, we present new insights derived from two comparable blocks of CLAudit measurements obtained in 2013 and 2016, when deployed across the PlanetLab [3] network and Microsoft Azure [4] DCs. We also extend the data post-processing pipeline with automated interpretation using an Interpretation tree, Impact Tree and an Interpretation Database. The resultant two-stage post-processing was ran offline on the comparable 2013 and 2016 data subsets and we present interpretation results together with what these have evolved into.

The main contributions of this work are:

- A comprehensive set of publicly available latency measurements of Cloud Services within multiple DCs, captured at multiple protocol layers, across a set of global Vantage Points;
- A detailed analysis of the collected measurements, including a comparison of the Cloud-service quality between the two periods;
- Interpretation of the detected suspicious events within the said time series, including breakdown of the events' likely root causes.

The presented insights into Vantage-Point to Data-Center latency are, to some extent, explainable by the already well

* Corresponding author.
E-mail address: ondrej.tomanek@fel.cvut.cz, tomanon1@fel.cvut.cz (O. Tomanek).

http://dx.doi.org/10.1016/j.comnet.2016.06.011
1389-1286/

investigated Internet service latency. Our work adds on top a study of impact caused by technologies specific to public Cloud Computing, such as by DC middleware (which terminates different protocol layers at different locations) or by data storage in remote databases. The measurements and interpretation show improvements in most aspects of the Cloud-Service experience over time. The impact of infrastructure improvements on the side of the Cloud Service Provider (CSP) is clearly observable, most notably in the improved performance of its core backend networking (reduced rate of incidents from 2.7% to 0.1%), and in the general reduction of the DC-centered ("global-impact") events–incidents.

The rest of this paper is organized as follows: Section 2 presents the related work. We then describe the multidimensional monitoring architecture and setup in Section 3, followed by insights derived from the collected measurements in Section 4. Measurements post-processing pipeline is described in Section 5 and the interpretation of results in Section 6. We conclude the work by summarizing implications and suggesting ways to continue in Section 7.

2. Related work

2.1. Network latency foundation

Network latency may be understood as a sum of delays along the communication path. The majority of the past latency-studying and tackling efforts has come from the Internet traffic-engineering domain, striving to ensure sufficient traffic QoS (Quality-of-Service). Results exist in the form of theoretical concepts, industrial solutions, RFCs and standards. A theoretical foundation of latency engineering was given by Queuing theory [5,6] and Network calculus [7,8]. Specific solutions such as guaranteed service networks or end-to-end jitter bounds were discussed in [9,10] and [11]. Because of their specificity and limited applicability to today's computer networks, overprovisioning [12] remains the most popular way of achieving QoS.

Many negative latency-related observations have been published, e.g. that significant portion of Internet traffic suffers from routing pathologies or that variations in end-to-end latency indicate long congestion periods [13–15]. More bad news include latency unpredictability [16,17] and the deep-seated tail latency phenomenon [18–21].

2.2. Cloud network latency

Latency is of the utmost importance to the Cloud, but the complexity and diversity of this environment prevent the conventional Internet measurement techniques to be easily adjusted to fit Cloud Computing needs. The problem partly being the scale, because the larger the scale the greater the impact of latency variability and the need for reliably low latency [22,23]. The ever-increasing interdependence of traffic patterns; context dependency and various stochastic factors render the task of capturing latency analytically intractable [24]. There was some success in approximating latency of specific environments (Cloudlet-to-Cloud and cellular-to-Cloud latency can be approximated by Rayleigh distribution [25]), but Cloud Computing service latency both inside and outside the DC lacks a good fit so far.

Miscellaneous Cloud measurements and analyses were conducted [26], often on platforms and tools designed in academia (like *Fathom* [27] or *Flowping* [28], often using PlanetLab [29]). Deriving traffic characteristics of flows inside a DC was the focus of [30] or [31]. Specific measurements concerning Cloud performance include [24] and [32]. End-user-perceived Cloud-application performance measurements were discussed for example in [33–37]. General network delay tomography and specific blackbox latency predictions are the topic of [38] and [39,40], respectively. Observations from multiple Vantage Points have been used previously [41–43], but these do not measure back-end or latency at multiple protocol layers. Active probing of Cloud resources [44] confirms need for sophisticated measurement, but is availability-oriented and does not document trends.

Commercial and research testbed latency-tackling implementations focus on reducing latency in a certain part of the Cloud. WAN acceleration heuristics [45] or careful path selection algorithms represent WAN-centric optimization. Both industry and academia have independently converged towards moving the Cloud closer to the end-user and offloading strategic responsibilities from remote DCs. Concepts such as Femto cells, Cloudlets or Fog Computing are starting to be successfully deployed, in mobile clouds foremost [46,47]. Innovations inside the DC include data path management [48–50], transport optimization, adjusting TCP behavior within the DC network or proposing entire new protocols (*DCTCP* [51], *D3* [52], *D2TCP* [53], *PDQ* [54], *pFabric* [55] and *delay-based TCP* [56]).

2.3. Suspicious event detection

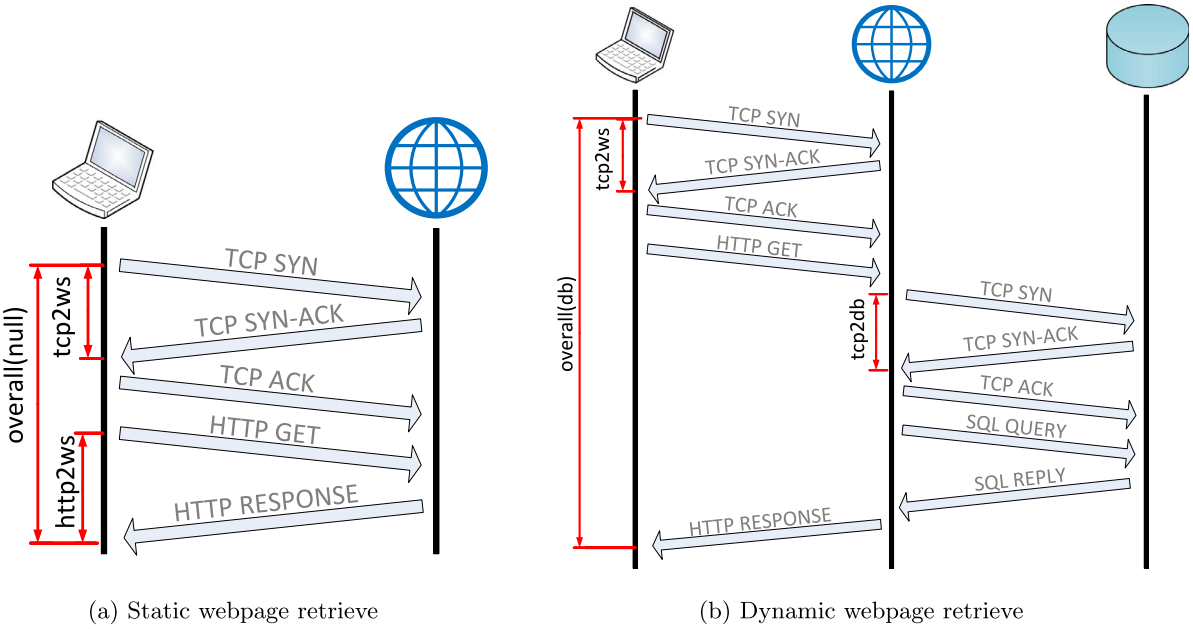
Anomaly detection is a well-known concept in telecommunications and networking. Previous Cloud and network studies focused on passive monitoring of: volumes of transferred data [57], CDN deployed on PlanetLab - *PlanetSeer* [58], or on active monitoring of ICMP and HTTP service availability [44]. Their common goal was to measure and aggregate suspicious events and anomalous behavior, whereas we focus on a framework for distinguishing suspicious events based on their geographic location and affected OSI layer.

Techniques were proposed for anomaly detection using a traffic-flow pattern analysis [59–61]. We focus on the user perspective specifically, studying Cloud latency measurements at multiple OSI layers and across multiple geo-dispersed Vantage Points. Cloud monitoring survey [62] summarizes the state of the art solutions in the field of Cloud monitoring, but only a little attention is given to detection of suspicious events in latency measurements. Latency as a metric is used for performance evaluation [63,64] and benchmarking [65], but not for suspicious event detection. Time-series studies [66–68] focused on similarity search using wavelets and statistical methods. In contrast, as we lack the definition of "normal" Cloud behavior, we are searching for deviations from its empirically derived parameter values.

Compared to the commercially-provided global monitoring software (like *Renesys* [69] or *ThousandEyes* [70]) we use advanced metrics, leverage co-incidences across all dimensions of measurement time series and provide automated interpretation of events, offloading much investigation from the end user. Furthermore, in contrast with commercial software, our measurements are publicly available to the research community for verification and further studies.

3. CLAudit measurement platform

CLAudit alias Cloud Latency Auditing Platform is a system for collecting and evaluating multidimensional measurements. By *measurements* we mean RTTs of individual protocol exchanges, processing times and overall latency, as shown on webpage retrieval processes in Fig. 1. By *multidimensional* we mean measurements capable of being looked at from point of view of Vantage Points, Data Centers and/or protocol layers (see examples in Figs. 3 and 4). This section provides a brief overview of CLAudit components and deployment together with examples of data it collects. For a detailed description, see [1].



(a) Static webpage retrieve (b) Dynamic webpage retrieve

Fig. 1. Static and dynamic webpage retrieves with time intervals included in discussed datasets denoted.

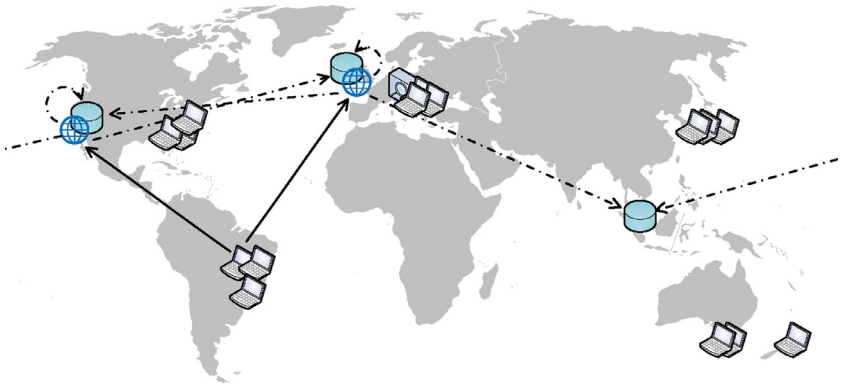


Fig. 2. *CLAudit deployment subset.* Part of the CLAudit infrastructure used for year-to-year comparisons. VPs are represented by laptops, front-end servers by globe icons and back-end servers by cylinder icons. Continuous and dashed curves depict VP to front-end and front-end to back-end measurements, respectively.

3.1. Components

Vantage points. Remotely-controlled VPs emulate real client appliances for interacting with Cloud-hosted applications and services. They collect, at various protocol layers, latency measurements of what end users perceive when utilizing Cloud. VPs are geographically dispersed to obtain end-user perspectives from around the globe. To maintain comparability, VPs are homogeneous in terms of OS and software (i686 Fedora Core 8 PlanetLab hosts). For the sake of redundancy and validation, VPs are deployed in triplets in every region (two VPs in a single campus and the third, backup VP in different campus nearby).

Cloud front-end. Servers that serve client requests (in our context respond to VP requests). Front-ends are geographically dispersed across Microsoft DCs, emulating a globally-distributed Cloud application. They are implemented as Azure PaaS Web Apps inside Free Tier (a single shared instance without backup per DC). The application container combines static and dynamic webpages.

Cloud back-end. Servers that provide data when front-end servers need them to compose the client response (e.g. dynamic webpage in our context). The back-end is, by definition, not involved in every interaction, although Cloud applications often consist of both the front-end and the back-end. In our setup, several back-ends are co-located with front-ends within a single DC, whereas other back-ends reside in DCs elsewhere. That is to take into account scenarios like remote storage, georedundancy, etc. Each back-end server hosts an Azure SQL database inside Standard Tier at S1 performance level, allowing for 90 concurrent requests.

Monitor. Monitor is a single *central* entity, that gathers data from all VPs; instructs and adjusts measurement; publishes, archives and analyzes the past and near-real-time data. Monitor is hosted on a high-end server on premises of the Czech Technical University in Prague.

The CLAudit deployment subset used for year-to-year comparisons is depicted in Fig. 2 and enumerated in Table 1.

3.2. Measurement setup

The request-response nature of many existing protocols allows for measuring RTTs and deriving latency. That is just what CLAudit does - in a continuous, simultaneous, large-scale distributed manner. Several measurement types, described in Table 2, are used for active probing between every *(VP, front-end)* and *(front-end, back-end)* pairs. There is neither a front-end, nor a back-end selection algorithm, as all the pairwise combinations are measured. We thus get an idea of Internet, intra-DC and inter-DC latencies.

Table 1
CLAudit deployment subset. Part of the CLAudit infrastructure used for year-to-year comparisons. Note that US and Japan VPs were relocated.

Component	Location
Vantage points (primary)	Melbourne, Belo Horizonte, Prague, Osaka/Hiroshima, Seattle/Atlanta
Vantage points (secondary)	Melbourne, Belo Horizonte, Prague, Osaka/Hiroshima, Seattle/Atlanta
Vantage points (backup)	Durham, São Paulo, Brno, Nagoya and Auckland
Front-ends	California, Dublin
Back-ends	California, Dublin, Singapore
Monitor	Prague

A measurement iteration of a single VP consists of a batch of 5 requests of every type and up to 5 respective responses received, repeating every 3 or 4 min[1]. We do not intend to conduct a stress test - neither PlanetLab's, nor Azure's acceptable user policy is thus violated. Partly because there are just a few packets generated and partly because these are spread over time due to unequal distances between VPs and DCs. Monetary cost of the experiment remains thus low.

RTTs are measured in milliseconds, rounded down to nearest integer. All measurement types have a reasonable 5 s timeout set, effectively treating timeouts and failures to respond the same way. Out of the 5 RTTs, only minimum and median values are stored while dropping the rest - we thus record 2 random variables. Such a measurement setup allows to reveal behavioral patterns lasting in order of tens of minutes. Shorter events cannot effectively be distinguished.

The *overall* type measures the end-to-end latency that drives the user experience. The *overall* RTT could have been approximated by a sum of particular RTTs and a little processing delay on servers. The motivation behind *overall* is to validate the measurements via an independent measurement done using different software utilities and to trigger the interpretation.

3.3. Datasets

Since 2013 the platform has been collecting data - several hundreds of samples per *(measurement_type, VP, front-end, back-end)* tuple. For documenting the possible trends we use two datasets, whose timeframes range from May 10th 2013 to June 4th 2013 and from January 10th 2016 to February 4th 2016. The main reason is the relatively good stability of PlanetLab nodes during that time. Out of the *(primary, secondary)* VP pairs, only the Australian one experienced outage from 12th to 13th of May 2013, rendering a piece of data irrecoverable. During the course of measurements, the platform has recorded a few 5000 ms samples, resulting either from RTTs over 5000 ms or failures to get any response. Both are treated the same way due to the 5 s timeout set. The fractions and their respective causes are presented in Table 3. The observed improvement is caused both by improvement on the CSP side as well as CLAudit engineering.

[1] 3 min in 2013–2015 and 4 min from 2016 on

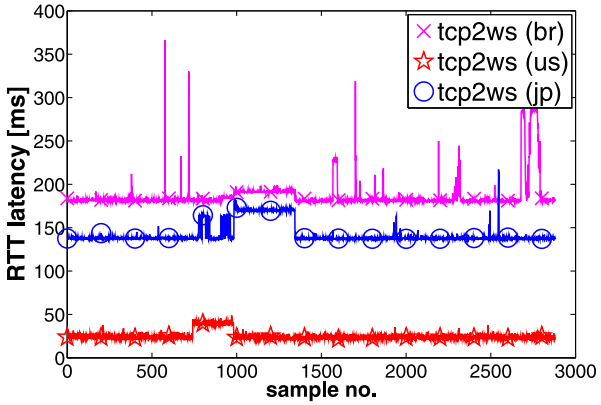


Fig. 3. *VP dimension example (2013).* A single week's *tcp2ws* data time series, as recorded by VPs (Brazil, USA and Japan) TCP-handshaking with California front-end. A normal Cloud network behavior is observed.

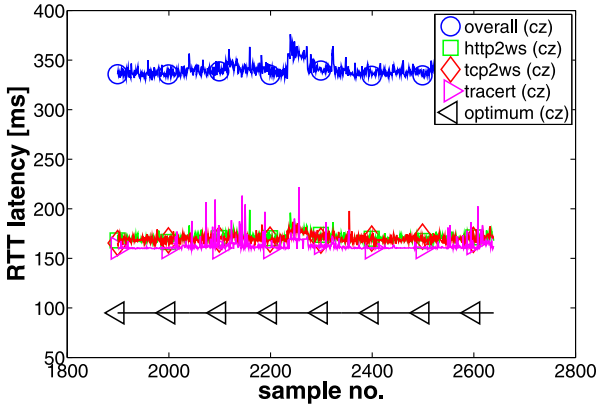


Fig. 4. *Protocol dimension example (2013).* Roughly two days of data from various protocol layers of front-end server in California, as recorded by Czech VP. The constant precalculated light's RTT denotes the optimal RTT.

3.3.1. VP dimension example

The example in Fig. 3 shows a *VP dimension* of median *tcp2ws* measurements, conducted against front-end webserver in California DC in 2013.

Latency observed by US VP looks very stable. It experiences low jitter and only minor spikes. The RTTs were roughly two times the optimal RTTs (according to Table 4). The more distant VPs not

Table 2
Measurement types included in discussed datasets.

Measurement type	Description	Back-end involved
tcp2ws	Time elapsed between VP's TCP SYN sent and TCP SYN ACK from web server received	No
tracert	Time elapsed between VP's ICMP Echo Request sent and ICMP Time Exceeded from farthest reachable network hop received. In 2013, ICMP Time Exceeded from a host containing IATA-like airport code of target DC location was used instead	No
tcp2db	Time elapsed between front-end web server's TCP SYN sent and TCP SYN ACK from back-end database server received	Yes
http2ws	Time elapsed between VP's HTTP request sent and static web HTTP response from front-end web server received	No
overall(db)	Time elapsed between VP's TCP SYN sent and HTTP response from front-end web server incorporating back-end database data received	Yes
overall(null)	Time elapsed between VP's TCP SYN sent and HTTP response from front-end web server received	No

Table 3
Fraction of `5000 ms` samples in 2013 and 2016, together with a cause and respective fractions among front-end-only measurements (in gray) and back-end measurements (involving a back-end DC).

Year	2013				2016			
Fraction	6.97% (66 128 samples)				0.16% (1 143 samples)			
Cause	PlanetLab		Other		PlanetLab		Other	
Fraction	63.28%		36.72%		24.85%		75.15%	
Back-end DC	None	California	Dublin	Singapore	None	California	Dublin	Singapore
Fraction of other	35.60%	0.55%	0.27%	0.29%	35.26%	11.99%	12.69%	15.22%

Table 4
Distance & light RTT. VP to DC distances [km] and RTTs [ms] a real-environment light achieves.

Distance & light RTT		Data center			
		California		Ireland	
		Dist[km]	RTT[ms]	Dist[km]	RTT[ms]
Vantage point	Melbourne	12644	127.81	17214	174.00
	Belo Horizonte	10404	105.16	8869	89.65
	Prague	9395	94.96	1470	14.86
	Osaka	8662	87.56	9579	96.82
	Hiroshima	8844	89.40	9498	96.01
	Seattle	1094	11.06	7300	73.79
	Atlanta	3440	34.77	6322	63.91

surprisingly experience more severe fluctuations and spikes. This measurement shows more or less normal behavior with VPs from more distant locations and inferior network infrastructures experiencing negative effects. Here we can mostly observe local effects with a single suspicious increase simultaneously observed by US and Brazilian VP, indicating a broader issue.

3.3.2. Protocol dimension example

The example in Fig. 4 shows the *protocol-layer dimension* - all the measurement types relevant to a front-end server in California DC, as measured during two days by the Czech VP in 2013.

Light's RTT denotes the optimum (as discussed in Section 4). *tracert* RTTs are slightly shorter than *tcp2ws* and *http2ws*, because the packets did not traverse inside the DC to the ultimate front-end server. *tcp2ws* and *http2ws* behave almost identically, except for occasional spikes in HTTP. *overall* validates a sum of involved RTTs. Specifically, a single round trip of TCP followed by a single round trip of HTTP together with a processing delay add up to the overall user-perceived latency.

3.4. Data publishing

The CTU in Prague hosts a web page for near-real-time data publishing and archivation. The data are available to the research community as well as general public under ODC-By license. The project website is http://claudit.feld.cvut.cz.

4. Insights into measurements

This section presents insights derived from the datasets. Included are *mean latency per kilometer* metric, availability, latency tails and multimodality observations together with possible trends over time. All these are subject to the fundamental *Speed of Light Barrier*.

Speed of light barrier

With evolving technologies clearly improving queuing, processing and transmission latency [71], understanding the propagation latency becomes ever more important. The propagation latency is, above all, dictated by the speed of light and becomes more significant with an increasing distance. Table 4 shows distances between VP and DC sites, calculated using the Great-circle distance - the shortest distance between two points on the WGS84 ellipsoid calculated using *geod* library, measured along the surface and ignoring differences in elevation [72]. Note that the distances are approximate as CSPs often do not disclose an exact location of the DC. Also note that, for various reasons, the actual data path is likely different from the path suggested by the Great-circle distance. Table 4 also shows the optimal round trip latency, achieved by a signal at a ∼66% speed of light in vacuum (e.g. light in fiber). We use this approximation as a lower bound according to the current understanding of the laws of physics.

4.1. Mean latency per kilometer

Table 5 summarizes the datasets using 50th percentile, chosen for its robustness. To help assess the Cloud connectivity latency on a data path in a particular region of the world, we define a universal metric called *mean latency per kilometer*, which is an amount of latency accumulated by packet's propagation over one kilometer distance and as such gives an idea about Cloud path performance in a given region. The more VPs in the DC region, the greater the accuracy. The real-environment light's *mean latency per kilometer* is ∼5.05 ms. Using that as a reference, in Table 5 we also listed the actual *mean latencies per kilometer* from which the distance from optimum can be easily seen. The best achieved latency was a little over the light's, observed on the path between Australia and Bay Area. The likely explanations include, first, both sites being situated on an opposite coast of ocean where the data path encounters less hops and, second, the underseas cable approximating the Great-circle distance well [73]. The worst mean latencies (over three times the light's) were observed on the intercontinental paths, where the data paths include many hops and, because of reasons like terrain and borders, are rarely conducted along the path suggested by the Great circle.

Mean latency per kilometer has mostly decreased between 2013 and 2016, resulting in a faster user-to-DC connection. Locating the improved network segments is difficult and would likely require a new set of measurements within partitioned paths. Hence, we can only speculate whom the bits of improvement should be attributed to, be it the CSP, the ISP or the PlanetLab. A consistent decrease

Table 5
Summarized RTTs. Cell rows correspond to the HTTP, TCP and ICMP (going from top). Cell columns correspond to the median of 2013 minima, median of 2016 minima, median of 2013 medians and median of 2016 medians, respectively.

Summarized RTT			Data center							
			median_rtt [ms] (mean_latency_per_km [μs])							
			California				Ireland			
			min13	min16	med13	med16	min13	min16	med13	med16
Vantage point	Melbourne	HTTP	174 (6.88)	163 (6.45)	176 (6.96)	164 (6.49)	311 (9.03)	296 (8.60)	313 (9.09)	297 (8.63)
		TCP	171 (6.76)	161 (6.37)	172 (6.80)	162 (6.41)	308 (8.95)	295 (8.57)	309 (8.98)	296 (8.60)
		ICMP	169 (6.68)	160 (6.33)	169 (6.68)	161 (6.37)	–	294 (8.54)	–	294 (8.54)
	Belo Horizonte	HTTP	183 (8.79)	207 (9.95)	185 (8.89)	209 (10.04)	231 (13.02)	256 (14.43)	233 (13.14)	257 (14.49)
		TCP	180 (8.65)	205 (9.85)	181 (8.70)	207 (9.95)	228 (12.85)	254 (14.32)	229 (12.91)	255 (14.38)
		ICMP	178 (8.55)	204 (9.80)	179 (8.60)	206 (9.90)	227 (12.80)	254 (14.32)	–	255 (14.38)
	Prague	HTTP	166 (8.83)	163 (8.67)	169 (8.99)	164 (8.73)	40 (13.61)	37 (12.59)	42 (14.29)	37 (12.59)
		TCP	163 (8.67)	159 (8.46)	168 (8.94)	161 (8.57)	38 (12.93)	34 (11.56)	42 (14.29)	34 (11.56)
		ICMP	160 (8.52)	158 (8.41)	160 (8.52)	159 (8.46)	34 (11.56)	33 (11.22)	–	34 (11.56)
	Osaka (2013) Hiroshima (2016)	HTTP	140 (8.08)	135 (7.63)	145 (8.37)	136 (7.69)	278 (14.51)	252 (13.27)	329 (17.17)	266 (14.00)
		TCP	136 (7.85)	132 (7.46)	137 (7.91)	132 (7.46)	272 (14.20)	249 (13.11)	274 (14.30)	262 (13.79)
		ICMP	133 (7.68)	132 (7.46)	134 (7.73)	133 (7.52)	–	270 (14.21)	–	271 (14.27)
	Seattle (2013) Atlanta (2016)	HTTP	24 (10.97)	54 (7.85)	28 (12.80)	54 (7.85)	160 (10.96)	99 (7.83)	228 (15.62)	99 (7.83)
		TCP	21 (9.60)	51 (7.41)	23 (10.51)	51 (7.41)	155 (10.62)	97 (7.67)	157 (10.75)	97 (7.67)
		ICMP	18 (8.23)	50 (7.27)	18 (8.23)	50 (7.27)	152 (10.41)	95 (7.51)	–	96 (7.59)

in mean minimum latency, observed by all-but-Brazilian VPs at all protocol layers, indicates an improvement in a Cloud connection in general. In the case of Brazilian VP, a >20 ms increase was detected at all front-ends at all protocol layers. This multidimensional observation indicates a degradation somewhere between the Belo Horizonte access connection and Microsoft's South American connection in between 2013 and 2016.

Since Azure DCs are known not to let ICMP packets in, these only reach the DC edge. This allows us to dissect the latency improvements made inside from those made outside. By subtracting mean ICMP latency improvements from mean TCP/HTTP latency improvements and comparing the results we observed a decrease in both the HTTP and the TCP latency across all DCs, implying a better responsiveness of the web. Improvements in both the DC connectivity and the protocol latencies indicate that Microsoft is paying attention to the latency and is evolving its Cloud Computing infrastructure with the latency in mind.

4.2. Latency tail

Cloud-like large-scale infrastructures can ignore neither tail latency nor latency variability. The analysis revealed the presence of both "troublemakers" in 2013 and 2016, too. The exact sources of the latency variability are hard to identify, whereas general sources of tail latency include network sources like congestion and application sources like power saving optimizations. We only attribute to the CSP the ones that were observed from multiple VPs, even though it is certainly possible for the CSP to influence a good deal of latency tail and variability introduced outside DCs (e.g. by improving exterior routing and peering at strategic Internet exchange points).

A summary of TCP CDFs is presented in Fig. 5 (TCP has become the dominant traffic type for Cloud applications). Tails tend to show up at various percentiles and sometimes add tens of milliseconds to about tenth of the traffic, which is beyond tolerance for certain classes of the applications.

Australia and New Zealand VPs observed a lot of variability by having minimum and median latency apart for the most part (Fig. 6). At Brazil and Japan VPs, variability is observed through irregular CDF shapes.

Table 6
Availability of Azure back-end – per measurement type and per DC.

Type	2013	2016	Back-end	2013	2016
tcp2db	99.37%	99.96%	California	99.74%	99.94%
			Dublin	99.88%	99.95%
			Singapore	99.85%	99.94%

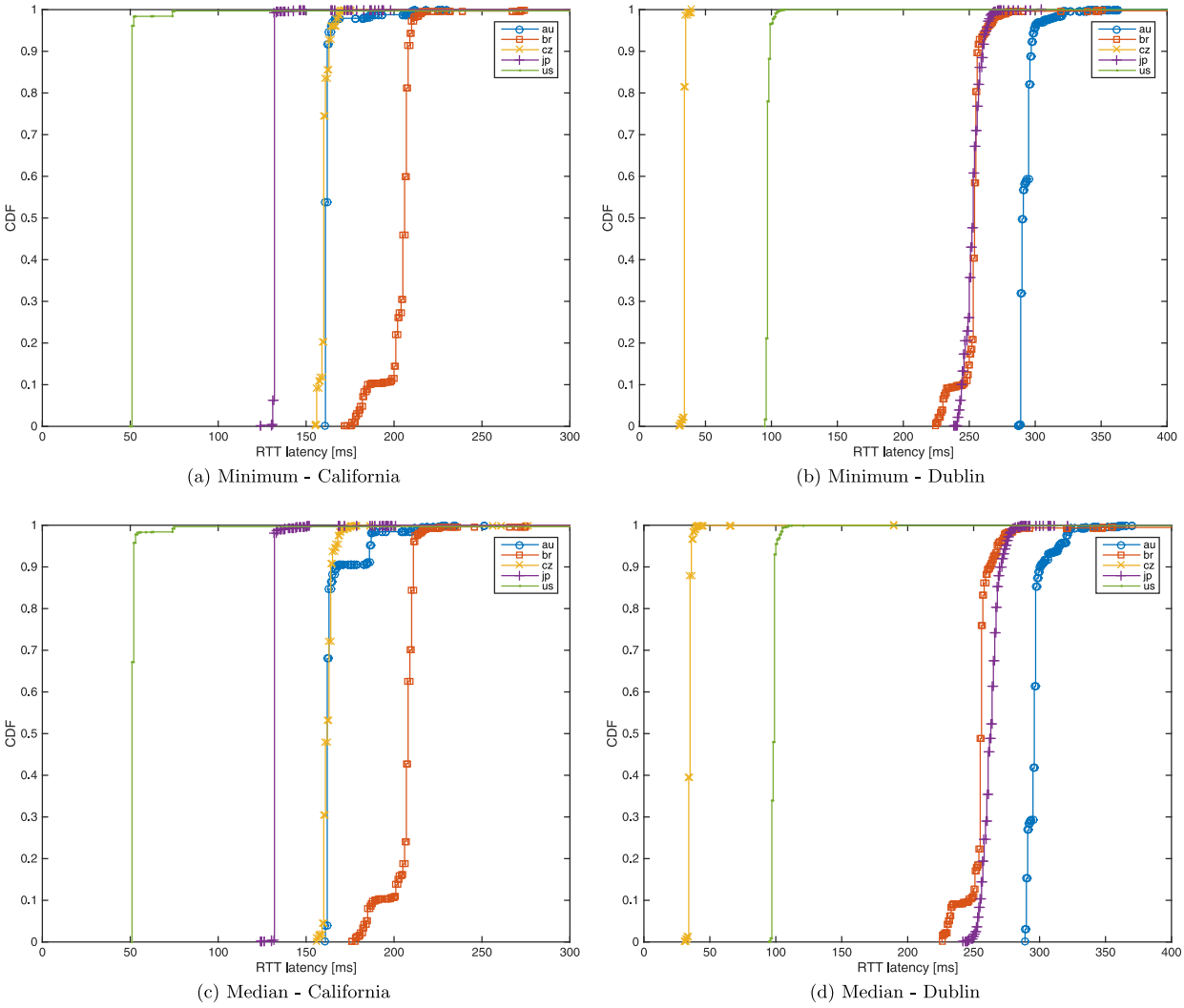
4.3. Multimodal distribution

A common reason for discrepancies and significant jumps in CDFs (Fig. 5) is a multimodal distribution of latency, which negatively affects applications. Analysis of the datasets identified three situations when bimodal distribution occurs - a persistent increase of latency (VP-to-front-end measurements), periodic effect (front-end-to-back-end measurements) and minimum-median discrepancy (VP-to-frontend measurements). The first two anomalies are described in [1].

An example of the third is presented in Fig. 6. Both the primary and secondary VPs (hosted in Melbourne) as well as the backup VP (hosted in Auckland) observed bimodal latency at all protocol layers when measuring Dublin DC services. As this concerns minimum and median measurements combined, it can be viewed as a bimodal mixture of two distributions, each having a different mode. The problem is likely on the path(s) from Australia and New Zealand to Dublin DC, as no other VPs observed such effect. Also, the Australia/New Zealand path to California DC did not display any similar problem. Among possible causes is taking alternative paths of different speed, but it is important to note that in general there is no correlation of hopcount to distance and latency [16], which is confirmed by our ICMP data.

4.4. Availability

DC availability and its change over time are other important metrics. Given our measurement setup, we consider an individual measurement successful should any of the 5 probes within a single batch succeed. The success rate than denotes the availability. We only use back-end data for availability analysis as front-end data are heavily influenced by PlanetLab instabilities.



(a) Minimum - California (b) Minimum - Dublin

(c) Median - California (d) Median - Dublin

Fig. 5. CDFs of minimum and median TCP RTTs as measured between each *(VP, DC)* pair. Discrepancies, bimodality, tail, minor effects and differences between the related supposedly-identical curves can be observed. Note that US and Japanese VPs were relocated between years due to Planetlab instabilities.

The results are listed in Table 6. Observed are availability improvements across all measured back-ends, the current availability tops 99.9% mark at individual DCs as well as TCP protocol layer.

4.5. Summary

Our end-to-end year-to-year Azure measurement insights can be summarized into the following:

- Cloud Computing service latency varies up to over three times the optimum;
- Circumstantial evidence of a substantial Cloud Computing latency decrease exists;
- TCP/IP protocol interbehavior is less coherent in the Cloud environment;
- Tail latency, latency variability and latency multimodality are still present on the end-to-end paths and partly explain why Cloud is not a predictable platform for all applications yet;
- Azure back-ends display a good availability over time, both protocol-wise and DC-wise.

5. Post-processing definition

This section describes the post-processing of latency measurements, i.e. *detection* and *interpretation* of suspicious events (Fig. 7). The process is a variation of unsupervised machine learning, where we possess no ground-truth data about DC or network events, but search for indications and in the interpretation phase use the time series' multidimensionality to estimate event causes.

5.1. Detection

We detect occurrences of suspicious events within the latency measurement time-series $X = \{x_i\}$. As suspicious events have various characteristics [1], we use a *Service Unavailability* check, followed by three different metrics, all operating over a non-overlapping sliding window $W = \{w_j\}$ (we have shown in [2] that an overlap in the sliding window does not bring any benefit). Each window of size S contains a subsequence of latency measurement samples x_i . When referring to the k-th element of j-th window we use the notation $w_j(k)$, $k \in \{1, 2, ..., S\}$. Metric values are compared to empirically determined thresholds. Metrics profile normal behavior and mark rest as suspicious using various formulas from the domain of descriptive statistics: *latency threshold, coefficient of variation* and *histogram* (see Fig. 8). Boolean outputs from metric calculations $Y = \{y_j\}$ may be combined into the resulting logical output $R = \{r_j\}$. Metrics are discussed briefly in the following subsections, for additional explanation see our previous work [2].

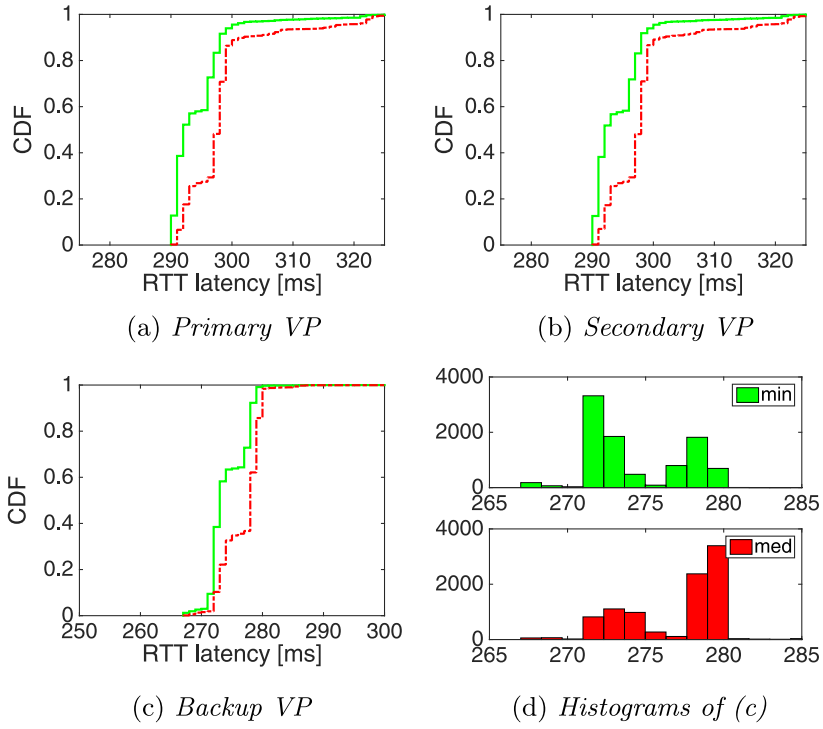


Fig. 6. *Example of a bimodal latency distribution.* Minimum and median HTTP RTTs apart for the most part, as observed from Australia and New Zealand VPs when measuring Dublin DC. Bimodality was observed at all protocol layers.

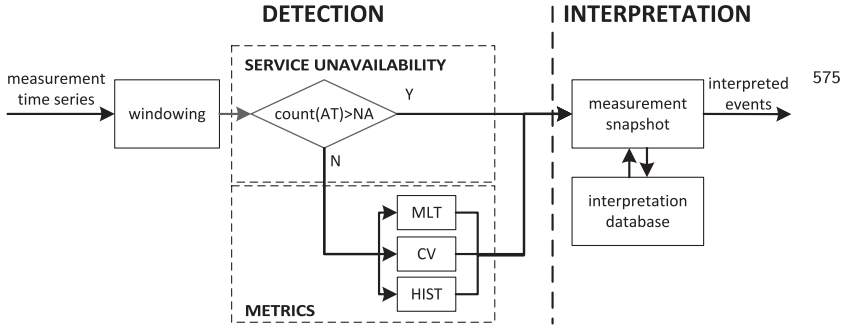


Fig. 7. *Post-processing algorithm.* An overview of *detection* and *interpretation* phases that transform latency measurements into interpreted events.

5.1.1. Service unavailability

We define measurement timeout MT as 5000 ms. We mark a window as suspicious if at least ϵ_{NAT} (Not Available Threshold) of its samples are missing:

$$y_j \iff ((count\,(x_i > MT)\ in\ w_j) > \epsilon_{NAT}). \quad (1)$$

We do not include these windows in the (subsequent) metrics computation.

5.1.2. MLT (maximum latency threshold) metric

The main intuition is that under steady-state conditions defined by the technical and physical specifications of the path between the VP and the DC server, the measured latency should reside close to its minimum value x_{min}. However, due to the generally unstable environment, a fraction of measurements might exceed $x_{min}*\epsilon_{MLT}$ (where ϵ_{MLT} is an input constant coefficient). Hence, we use two conditions to detect suspicious events: First, a condition to mark x_i as suspicious:

$$x'_i \iff (\ x_i > x_{min} * \epsilon_{MLT}\ ), \quad (2)$$

resulting in boolean representation of X, denoted $X' = \{x'_i\}$, and windows $W' = \{w'_j\}$. Second, a condition that marks the window w'_j as suspicious if a fraction of suspicious values (i.e. $x'_i == 1$) is greater than fixed threshold $N \in \{0, 1, \ldots, S-1\}$:

$$y_j \iff \left(\sum_{k=1}^{S} w'_j(k) > N\right). \quad (3)$$

As minimum latency evolves over time, the x_{min} value should be adjusted accordingly. Auxiliary mechanisms should be implemented so that the algorithm can continuously detect suspicious events, e.g. an aging function, periodically resetting the x_{min} value. Considering the time span of our datasets, we did not implement such a mechanism.

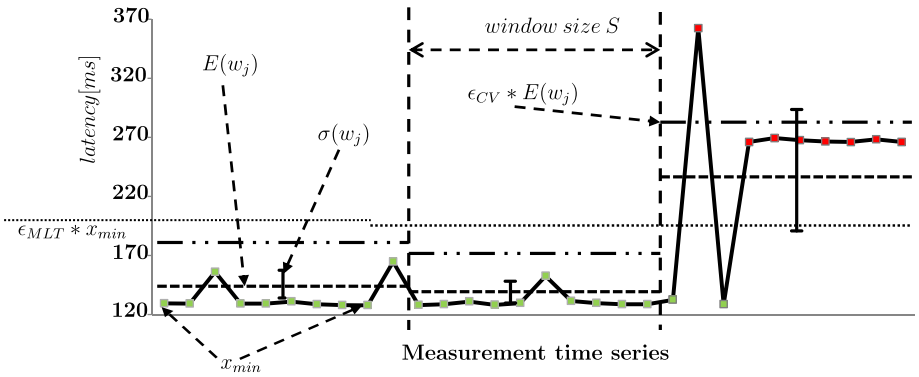
5.1.3. CV (coefficient of variation) metric

The premise here is that an increased coefficient of variation over the measurements window $cv(w_j) = \frac{\sigma(w_j)}{E(w_j)}$ indicates instability. The tolerable extent is given by a (constant) threshold ϵ_{CV}:

$$y_j \iff (\ cv(w_j) > \epsilon_{CV}\ ). \quad (4)$$

5.1.4. HIST (Histogram) metric

We assume that measurements over a short time exhibit DC-specific characteristics, unless conditions change, which indicates



a)

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
MAXIMUM LATENCY THRESHOLD METRIC																														
$x'_i[boolean]$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1	1	1
$y_j[boolean]$	0										0										1									

b)

CV METRIC			
$\sigma(w_j)[ms]$	12.7	7.1	65.8
$E(w_j)[ms]$	135.4	131.7	249.4
cv	0.094	0.054	0.264
$y_j[boolean]$	0	0	1

c)

HISTOGRAM METRIC																														
H_{TEST}																														
$B = 10$	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
p_{LOW}, p_{HIGH}	128, 157										128, 157										128.3, 151.8									
H_{TRAIN}	1	7	0	0	0	0	0	0	1	1	1	7	0	0	0	0	0	0	1	1	1	6	0	1	0	0	0	0	1	1
H_{TEST}	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	1	0	0	0	1	1	0	0	0	0	0	0	8
$y_j[boolean]$	N/A										0										1									

Fig. 8. Example of the MLT, CV and HIST metrics computed over a latency measurements sample subset.

a suspicious event. Histograms of W are searched for deviations from historic measurements (H_{train}). The histograms have a constant number of elements $B = 10$. A demarcation range sets the value of histogram edge elements. We use a range set by low and high percentiles computed from an initial training set of measurements. We emphasize the more recent measurements using a low-pass filter, where we sum up the historic histogram H_{train}, weighed by the parameter α, with the most recent non-suspicious measurements H_{test}, weighed by $(1-\alpha)$, $\alpha \in (0, 1)$. The outcome is then compared to the current H_{test} using root mean square error ($RMSE$). A (constant) threshold ϵ_{HIST} then decides whether to mark a window suspicious:

$$y_j \iff (RMSE(H_{train}, H_{test}) > \epsilon_{HIST}). \quad (5)$$

5.2. Interpretation

Given the blackbox nature of the CLAudit measurements, we can only make educated estimates as to the interpretation of the measured data and of the detected events. However, by exploiting data multidimensionality, we are able to confidently identify and separate various causes of latency events, such as a network failure or a DC issue. In the following, we list these events together with the expected footprint in the multidimensional detection array, as determined by the detection metrics computation. The process of results interpretation consists of creation of a *Measurement snapshot* of all measurements in a given window, followed by its comparison to an *Interpretation database*.

5.2.1. Measurements snapshot

Detection metrics outputs are combined in 2D arrays called *snapshots*, which also contain information about the measurement type, time stamp, VP and back-end location. Each snapshot corresponds to a single w_j in the latency measurements of a single front-end server. Outputs from the metrics are then combined into the resulting output $R = \{r_j\}$, which is then compared to the *Interpretation database*.

5.2.2. Interpretation and likely root cause

Interpretation database (Table 7) is a multidimensional array, consisting of interpretable suspicious event series and their likely

Table 7
Interpretation database. Part of the database with entries classified as having local impact (as denoted by a blue line in the Impact Tree in Fig. 9). Identical rows follow that classify entries as having regional or global impact.

Triggering event	Local VP	VPs	DC	DB	Underlying events					Impact scope			Likely root cause
					overall	http2ws	tcp2db	tcp2ws	tracert	non-DB	DB		
										VP	DC	DB	
overall(null)	1	1	1	n/a	n/a	n/a	n/a	n/a	n/a	Local	Local	Local	False positive or VP failure
	Both	1	1	n/a	n/a	N	n/a	Y	N	Local	Local	Local	Front-end TCP handshake
	Both	1	1	n/a	n/a	Y	n/a	Y	Y	Local	Local	Local	Path/routing
	Both	1	1	n/a	n/a	Y	n/a	Y	N	Local	Local	Local	Data connection
	Both	1	1	n/a	n/a	Y	n/a	N	N	Local	Local	Local	HTTP service
	Both	1	1	n/a	n/a	Y/N	n/a	N	Y	Local	Local	Local	Weak positive
	Both	3	1	n/a	n/a	Y/N	n/a	Y/N	Y/N	Regional	Local	Local	...
	...	...	...	...	...	...	...	...	...	...	...	...	...
overall(db)	1	1	1	n/a	n/a	n/a	n/a	n/a	n/a	Local	Local	Local	False positive or VP failure
	Both	1	1	1	N	N	Y	N	N	Local	Local	Local	Back-end TCP handshake
	Both	1	1	1	Y/N	Y/N	Y	Y/N	Y/N	Local	Local	Local	Same as overall(null) + back-end TCP handshake
	Both	1	1	n/a	Y/N	Y/N	N	Y/N	Y/N	Local	Local	Local	Same as overall(null)
	Both	3	1	1	Y/N	Y/N	Y/N	Y/N	Y/N	Regional	Local	Local	...
	...	...	...	...	...	...	...	...	...	...	...	...	...
http2ws	1	1	1	n/a	n/a	n/a	n/a	n/a	n/a	Local	Local	Local	False positive or VP failure
	Both	1	1	n/a	n/a	n/a	n/a	N	N	Local	Local	Local	HTTP service
	Both	1	1	n/a	n/a	n/a	n/a	Y	Y	Local	Local	Local	Path/routing
	Both	1	1	n/a	n/a	n/a	n/a	Y	N	Local	Local	Local	Data connection
	Both	1	1	n/a	n/a	n/a	n/a	N	Y	Local	Local	Local	Weak positive
	Both	3	1	n/a	n/a	n/a	n/a	Y/N	Y/N	Regional	Local	Local	...
	...	...	...	...	...	...	...	...	...	...	...	...	...

root causes constructed from an *Interpretation Tree* (Fig. 9(a)). Interpretable suspicious events are denoted as *triggering* and the suspicious events used for their interpretation as *underlying*. Likely root causes include *weak positives*, i.e. suspicious events not supported by the corresponding OSI layer hierarchy measurement detections (e.g. a *tcp2ws* event is detected but the "upper" *http2ws* is not).

In the following we use notation from Table 2. We created an *Interpretation Tree*, which interprets suspicious events into likely root causes using a top-to-bottom OSI model approach by designation of *triggering events*: {*http2ws, overall(null), overall(db)*}, and analysis of dependencies of detected events in *underlying* protocols: {*tracert, tcp2ws, tcp2db, http2ws*}, where {*http2ws, overall(null)*} can be either *triggering* or *underlying* for {*overall(null), overall(db)*}.

Fig. 9(a) shows the simplified Interpretation Tree for the case of a Suspicious Event detection from a single VP. The primary VP's detection is validated by the secondary VP's measurements. The multidimensionality of the Interpretation database provides the ability to point out the affected area in cases where multiple VPs detect suspicious front-end or back-end event and *Impact Scope* in Table 7 changes to regional (multiple adjacent VPs) or global (all VPs), according to Fig. 9(b).

Description of the likely root causes created from an Interpretation Tree (Fig. 9(a)) and measurement descriptions (Table 2) can be found in Table 8.

5.2.3. Examples

Figs. 10 and 11 show different examples of event interpretation. Fig. 10 shows suspicious event detection in multiple measurement types (protocol layers), indicating a local issue - *local impact*, an instance of the *http2ws - data connection issue* event from the interpretation database in Fig. 9(a). In contrast, Fig. 11 shows a suspicious event detection using multiple geographically dispersed VPs - *global impact*, indicating a DC issue - *HTTP service root cause* (see Fig. 9(b)). The examples show how interpretation of a triggering event is spread across different dimensions, where we investigate the affected location and the underlying events.

Table 8
Likely root cause description.

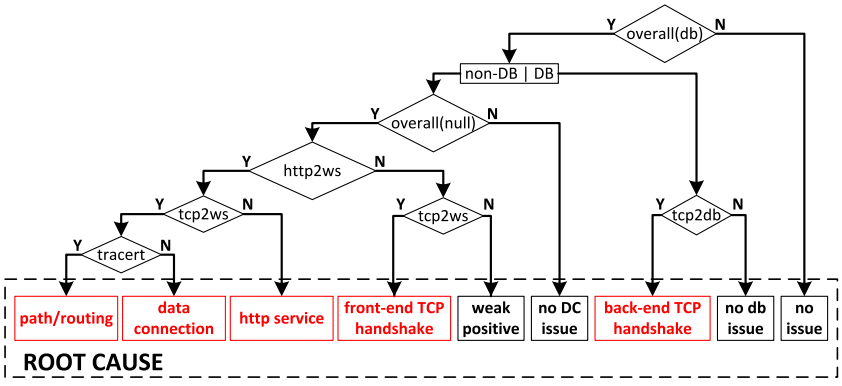
Likely root cause	Description
Path/routing	Issue affecting all measured OSI layers
Data connection	Issue affecting TCP/IP traffic between VP and front-end server
HTTP service	HTTP service issue
Front-end TCP handshake	TCP handshake issue between front-end server and VP
Back-end TCP handshake	TCP handshake issue between front-end and back-end server
Weak positive	Event not supported by corresponding OSI layer hierarchy measurement detections
False positive	Discrepancy in VP pair detection
VP failure	Measurement failure on one of the VPs from VP pair

6. Measurement post-processing evaluation

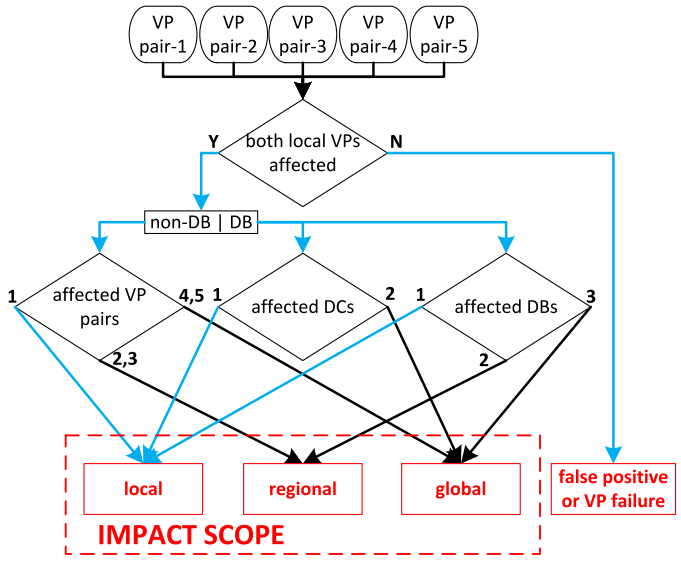
Evaluation consists of *Detection*, where we analyze characteristics of our findings, and *Interpretation*, where we analyze relations among the results from all the measurement types. The parameter optimization process was described in our previous work [2], added here is the *Service Unavailability* check.

6.1. Data processing setup

The initial 20% of measurements were only used for initial adjustment of the metric parameters, i.e. the selection of x_{min} and creation of a representative H_{train} and demarcation range. We chose 20%, as our experiments have shown that the metric parameters stabilize at around 10% and we are left with enough representative data for the interpretation. Due to lower occurrence of missing samples, we process the minimum latency values only. Dataset size information can be found in Table 9. Compared to our previous work [2] we improved the synchronization by timestamping packets. We use one VP from each location in our analysis and interpretation. Compared to our previous work we use $R \geq 1$ - i.e. detection made by one metric is sufficient to mark a window as



(a) *Interpretation Tree*



(b) *Impact Tree*

Fig. 9. *Interpretation diagram.* a) Interpretation Tree for the case of a suspicious event detected by a pair of VPs in a single location; b) Impact Tree for the classification of impact scope of the detected likely root cause. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 9
Dataset statistics.

Dataset statistics	2013	2016
Days	26	26
Measurements	76	76
Samples	948,480	711,360
Windows, total	94,848	71,136
Windows, no missing sample	80.5%	99.4%
Windows, 1–4 missing samples	14.7%	0.6%
Windows, 5–10 missing samples	4.8%	0.1%
Windows, training only (20%)	19,000	14,212
Windows, evaluated	75,848	56,924

suspicious, and we show that a larger number of detected suspicious events can be filtered via an improved data interpretation.

6.2. Detection

In this subsection we justify parameter setting of *Service Unavailability* and analyze and compare statistical results of the detected suspicious events.

A *Service Unavailability* check was added to the algorithm to address the missing samples. In Table 9 we show the difference between missing samples in windows in both datasets, where $\epsilon_{NAT} \in \langle 0, 10\rangle$, $MT = 5000$ ms.

In the computation of the *MLT* metric we use a condition, that a minimum of 5 samples in each window must be larger than the threshold in order to mark the window suspicious. As missing samples have the highest latency, the ϵ_{NAT} value of the *Service Unavailability* check must be set to 5, otherwise outcome of the metrics computation would be biased.

6.2.1. Suspicious event analysis

Metrics relevance. We verify and compare whether each metric justifies its own existence, i.e. individually marks distinct windows. In Fig. 12(a) and (c) we observe a decrease in windows marked by *Service Unavailability* over the years and a significant decrease in windows marked as suspicious. A significant fraction of windows marked as suspicious in 2013 was caused by condition marking a window as suspicious if at least one metric marks it. This loose condition allows us to analyze a broader range of suspicious events and filter only those that conform to the interpretation rules in later stage of *Detection and Interpretation*. Fractions of individual metric detections among the years in Fig. 12(b) and (d) remain comparable. In Table 10 we see that in 2016 *MLT* and *CV* metrics have the highest marking rate in back-end measurements *overall(db), tcp2db* whereas HIST metric has the highest

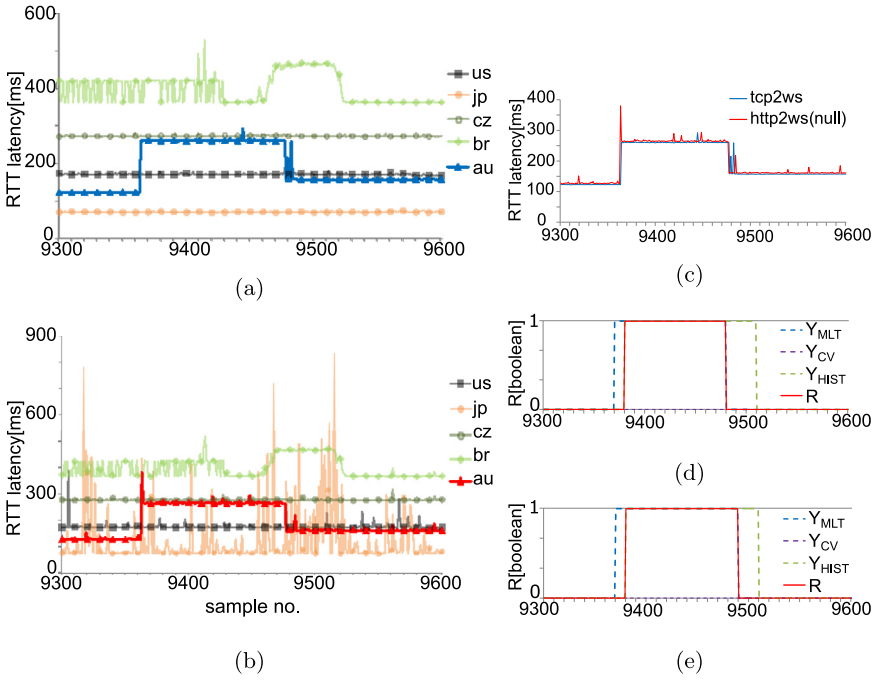


Fig. 10. *Example of an isolated (local impact scope) suspicious event.* a) *tcp2ws* and b) *http2ws* from all VPs targeting the same front-end server; c) *tcp2ws* and *http2ws* from the Australian VP marked as suspicious; d) *tcp2ws*: metrics and (the combined result) *R*; e) *http2ws*: metrics and *R*.

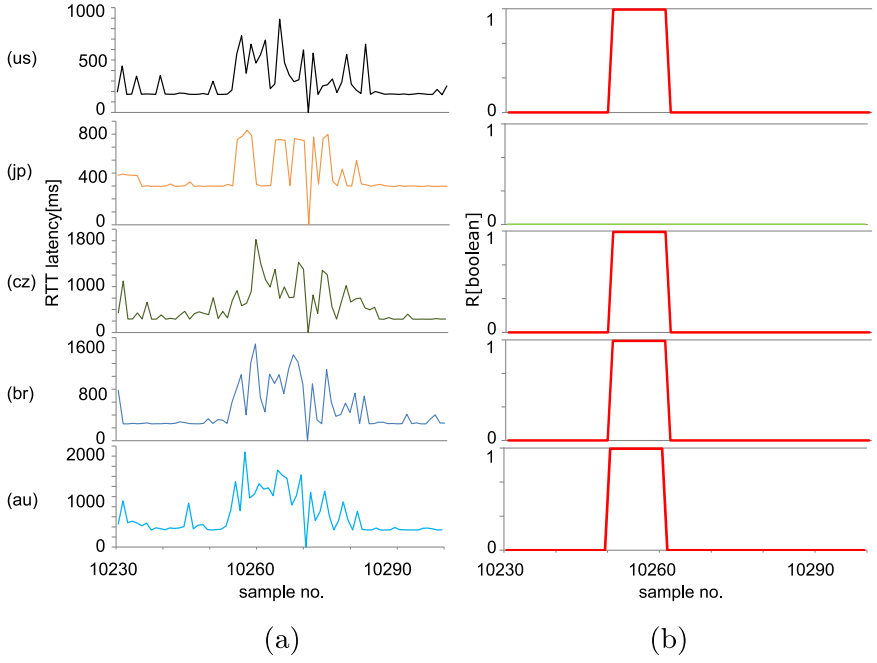


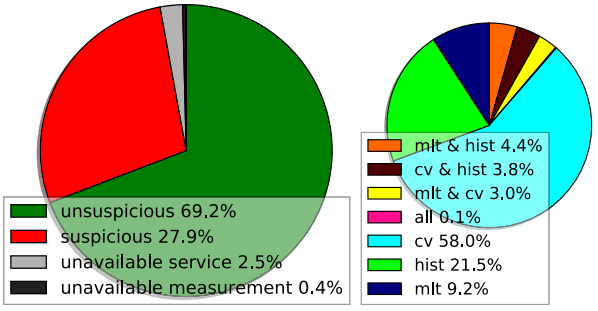
Fig. 11. *Example of a concurrent suspicious event.* a) *http2ws* from all VPs targeting the same front-end server and back-end server combination, b) combined results $R \geq 1$ of the metric detections for each VP (green line indicates a window not marked as suspicious as no metric marked it). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 10
Metric hit rate per measurement type. Table shows fractions of detected measurement types per metric.

Metric hit rate	http2ws		overall(null)		overall(db)		tcp2ws		tracert		tcp2db	
	2013	2016	2013	2016	2013	2016	2013	2016	2013	2016	2013	2016
MLT	1.6%	0%	0.1%	0%	48.9%	0.2%	4.2%	0%	0%	0%	45.1%	99.8%
CV	13.2%	1.8%	11.7%	2.5%	45.4%	36.1%	4.8%	1.8%	13.6%	3.4%	11.3%	54%
HIST	5.3%	12.5%	6.1%	8.4%	9.1%	5%	7.7%	37.9%	16.0%	33.6%	55.7%	3%
≥1 METRIC	10.1%	6.2%	9.3%	4.9%	37.9%	22.5%	5.7%	16.7%	12.9%	15.8%	24.0%	34%
SERVICE UNAVAILABILITY	0.2%	15.6%	0.2%	15.6%	1.2%	43.8%	0.5%	12.5%	95%	12.5%	3%	0%
COMBINED	9.3%	6%	8.6%	5%	34.9%	22.7%	5.3%	17%	19.6%	16%	22.3%	33.7%

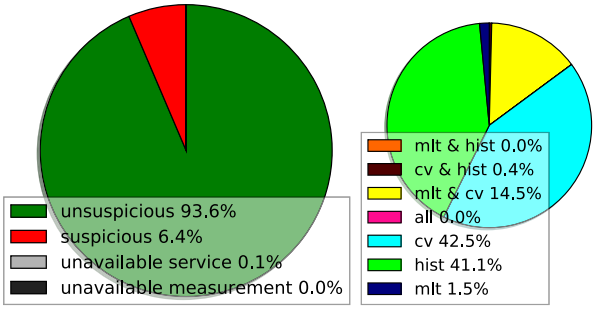
Table 11
Likely root cause summary. Table shows likely root causes and corresponding scope of impact of measurement snapshots interpreted according to triggering events - *http2ws, overall(null), overall(db)*. In 2013 we interpreted 6631 snapshots out of 9980 and in 2016 we interpreted 1070 out of 7490. Description of the likely root causes can be found in Table 8

Likely root cause			2013		2016	
Likely root cause	Back-end TCP handshake	Impact scope	Out of suspicious snapshots	Out of all snapshots	Out of suspicious snapshots	Out of all snapshots
Path/routing	N	Local	3.3%	2.2%	18.3%	2.6%
Path/routing	N	Regional	1.5%	1.0%	0.0%	0.0%
Path/routing	N	Global	0.2%	0.2%	0.4%	0.1%
Data connection	N	Local	0.7%	0.5%	8.8%	1.3%
HTTP service	N	Local	6.7%	4.5%	0.9%	0.1%
HTTP service	N	Regional	2.5%	1.7%	0.0%	0.0%
Path/routing	Y	Local	1.0%	0.7%	1.3%	0.2%
Path/routing	Y	Regional	0.4%	0.3%	0.0%	0.0%
Path/routing	Y	Global	0.2%	0.1%	0.6%	0.1%
Data connection	Y	Local	0.1%	0.1%	0.6%	0.1%
HTTP service	Y	Local	1.2%	0.8%	0.0%	0.0%
HTTP service	Y	Regional	0.2%	0.1%	0.0%	0.0%
Back-end TCP handshake	N/A	Local	21.0%	13.9%	3.1%	0.4%
Back-end TCP handshake	N/A	Regional	27.0%	17.9%	0.2%	0.0%
Back-end TCP handshake	N/A	Global	4.1%	2.7%	0.7%	0.1%
Weak positive	Y/N	Local	11.7%	7.8%	58.9%	8.4%
Weak positive	Y/N	Regional	16.9%	11.2%	5.3%	0.8%
Weak positive	Y/N	Global	1.2%	0.8%	0.7%	0.1%
Sum				66.5%		14.3%



(a) *2013* : all windows #windows=75,848

(b) *2013* : suspicious windows #windows=21,188



(c) *2016* : all windows #windows=56,924

(d) *2016* : suspicious windows #windows=3,618

Fig. 12. *Suspicious event numbers. all windows* show fractions of windows marked as suspicious by at least one metric, unsuspicious windows and unavailable windows with missing ≥ 5 samples due to unavailable measurement or unavailable service - i.e. no response from CSP. *Suspicioious windows* show detailed view of windows marked as suspicious.

rate in DC measurements *http2ws, tcp2ws, tracert*. In 2013 all metrics have the highest ratios in back-end measurements. Susceptibility of the metrics to missing samples, given their definition, goes from highest (*MLT*, which compares to minimum) to lowest (*HIST*, which compares to historic histograms). In 2016 missing samples only had little effect on overall dataset, whereas in 2013 the *tracert* measurements were affected due to the different measurement approach from 2016. In 2013 the back-end measurements - i.e. *tcp2db*, had a long-term character with alteration of their mean latency which resulted in same marked windows by *HIST* and *MLT* metrics. Diversity of measurement type ratios among metrics justifies their existence by marking partly distinct set of suspicious events.

Persistence of detected suspicious events was measured per each measurement type (Fig. 13). We search for events that are a deviation from normal behavior. Most of the detected events have a short-term character, i.e. $duration \leq 30\tilde{}min = 1 * w_i$, except back-end measurements. Long-lasting continuous suspicious events, i.e. $duration \geq 300\tilde{}min = 10 * w_i$ in 2013 (400 min in 2016), are an undesired algorithm feature, but with minor occurrence in combined results, . Results in Fig. 13 complement the comparison of measurement type distributions in *Metric hit rate* (Table 10). Differences in persistence (Fig. 13(d)–(f)) show that the metrics differ in their susceptibility to continuously mark latency deviations as suspicious events from their inception. The metrics differ not only in susceptibility to mark a window as suspicious based on the statistical nature of the measurements but also in disposition to detect continuous suspicious events. We also observe complementary information to previously-mentioned cause of *Service Unavailability* in 2013. In Fig. 13(f) we see that, as expected, the *HIST* metric is by definition more susceptible to continuous detection of suspicious events than other metrics.

Concurrent detections from geographically dispersed VPs. Fig. 14 shows histograms associating the number of geographically dispersed VPs detecting a suspicious event within the same time window and the measurement type. We distinguished combinations of multiple back-ends for *overall(db)*.

Most of the metric-detected suspicious events in both years are detected at a single VP (indicating a local, regional or a network issue) whereas *Service Unavailability* suspicious events are detected at multiple concurrent VPs, indicating a DC (2013) or back-end (2016) issue. This implies that the proposed geographically-dispersed detection approach is feasible and allows to point out the location of the issue and affected protocols or to create performance statistics of Cloud services per VP location.

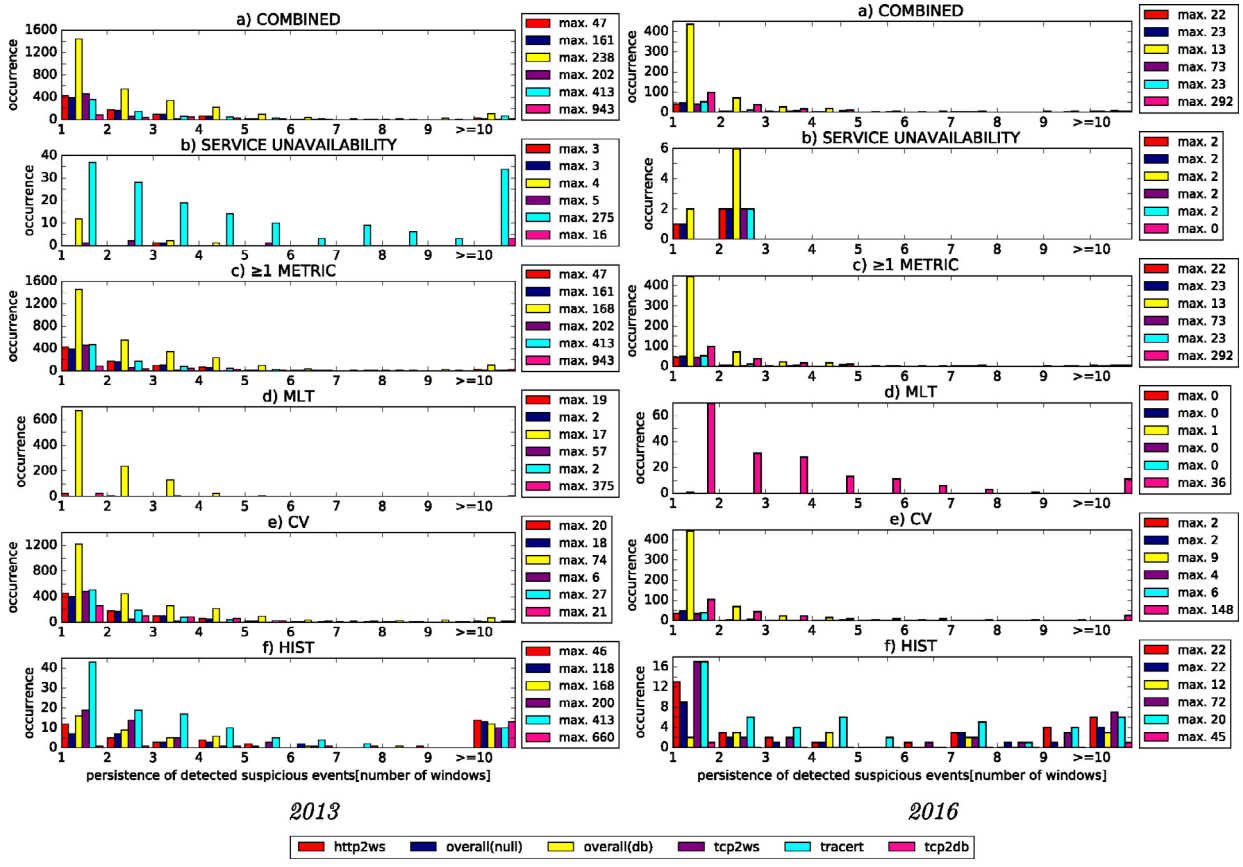


Fig. 13. *Persistence of suspicious events.* Histogram of consequent event windows. Events spanning more than ten windows are grouped in last part of histogram (≥ 10). Maximum per each measurement type can be found in color legend. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

The differences in Fig. 14 show that each metric is prone to a large number of concurrent detections across different types of measurements. The MLT metric detects a large amount of events in back-end measurements - *overall(db), tcp2db*, which is supported by high multitude of higher-persistence values 3–5 in Fig. 13(d) compared to other measurements. The CV metric detects variations in time windows and is more effective in detecting concurrent events in measurements prone to variability - *http2ws*, but is susceptible to intermittent course of event detection, which can be seen in large multitude of small persistence values 1-2 in Fig. 13(e), but it is able to detect concurrent events from VPs regardless of measurement type or behavior (see Fig. 14). The HIST metric takes a different approach from MLT and CV. This results in a dissimilar persistence of events (Fig. 13(f)) and a collocation of values in Fig. 14 compared to other metrics. The *Service Unavailability* check shows a large number of concurrent detections from multiple VPs in *tracert* in 2013 (Fig. 14), other measurements contain windows marked by *Service Unavailability* mainly from a single VP. By combining results of the different metrics we would lose information about specific detections made by each of them, therefore we would rather collect all detections from all metrics and filter out the weak positives.

6.3. Interpretation

In this section we present interpretation results for triggering events : *http2ws, overall(null), overall(db)*. Overall, the statistics improved significantly, as indicated among others by the lower number of concurrent detections from multiple VPs (Table 12). The combined statistics in Table 11 summarize performance of DCs in both years. In 2016, we observe a significant decrease in the number of detected events. Considering the absolute values in Table 11, we observed a significant decrease in regional weak positives and in interpretable events over the years. The methodology of weak-positive filtering gives us the ability to utilize divergency of metric-detected events by prioritizing those events for analysis that contain sufficient amount of information to be interpreted. A strict following of the methodology affects categorized likely root causes with regional and global impact scope, as when one measurement from one VP does not adhere to the methodology, the event is marked as weak positive. Especially given the much lower number of global issues detected in 2016 we conclude that the overall DC performance has improved over the years.

Table 12
Impact scope of interpreted suspicious events. Table shows fractions of concurrent VP detections of triggering events. In other words, how many VPs detect an event at the same time. In 2013 we interpreted 6631 snapshots out of 9980 and in 2016 we interpreted 1070 out of 7490

# Vantage points	2013	2016
1	45.8%	92.0%
2	32.1%	4.4%
3	16.4%	1.3%
4	3.8%	0.7%
5	1.9%	1.6%

We interpret results from Table 11 as follows:

1. Stability of database measurements improved significantly in 2016, as incidence of the 'back-end TCP handshake' event de-

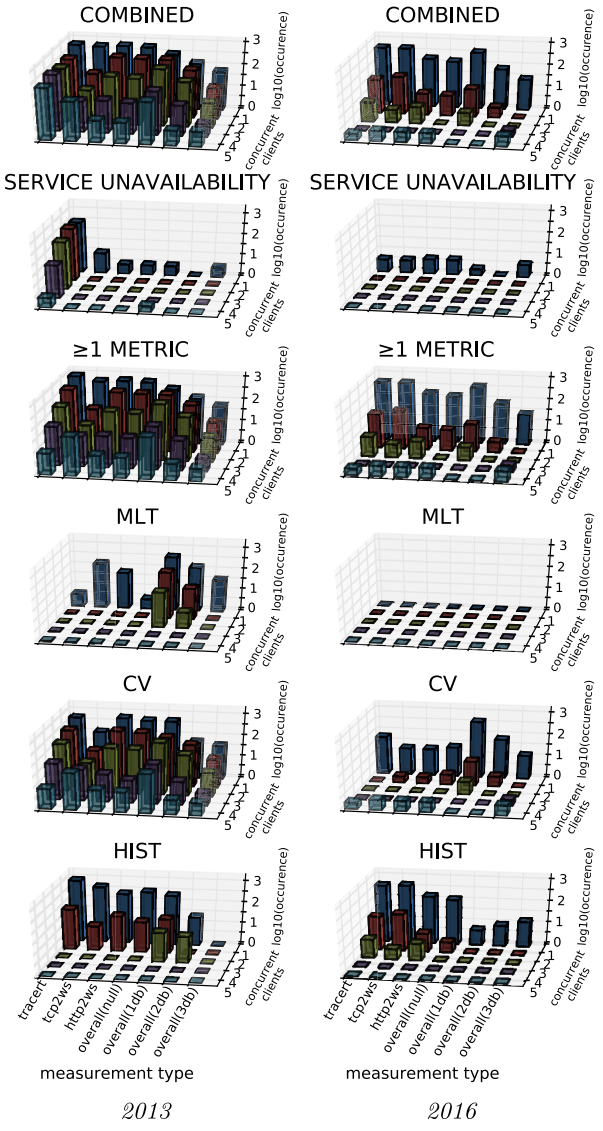


Fig. 14. *Concurrent detections from geographically-dispersed VPs in the same time window.* Measurement-type axis has separate *overall(db)* measurement types to distinguish between single detection from clients on a same back-end server and combination of multiple back-end servers. *tcp2db* is not included as it is not measured from each VP.

creased to near zero values across all impact scopes. This also affected all other likely root causes triggered by the *overall(db)* event: {path/routing, data connection, HTTP service, back-end TCP handshake}, resulting in decrease of their incidence.

2. The increased incidence of the path/routing likely root cause with local impact scope suggests an increased instability in Internet connectivity. This is an issue which should be addressed in cooperation with local ISPs.
3. Overall, the weak positive statistics improved between 2013 and 2016, except for a slight increase in weak positives with local impact. This suggests that while the Cloud infrastructure improves, our VP technology, PlanetLab, and its networking, remain as unreliable as ever.
4. On likely root causes including the back-end TCP handshake we observe that in cases of complex communication, back-end server issues occur concurrently across all the interpreted likely root causes: {path/routing, data connection, HTTP service} except for the front-end TCP handshake.
5. We observe a slight increase in incidence of the data connection likely root cause with local impact. We interpret this as an issue with TCP traffic prioritization at a particular VP.
6. A significant decrease in incidence of HTTP root cause across all impact scopes shows that availability and stability of user-facing protocols on the measured front-end servers has improved.

7. Conclusions and future work

This work presents multidimensional latency monitoring of a Cloud service. By operating on a large-scale dataset collected by our measurement platform, it demonstrates merits and limitations of multidimensional evaluation. Results include insights into Windows Azure Cloud-service latency and its possible trends over three years. Importantly, the evaluation provides circumstantial evidence of Cloud-service latency gradual improvement, namely the decrease in regional and global suspicious events between 2013 and 2016, making the Cloud more appealing to emerging environments like the Internet of Things.

The presented distributed measurements and interpretation can be utilized for improvement of Cloud services. They may justify existence of offload infrastructures such as Edge Computing or Cloudlets, or be used to implement feedback control, allowing for real-time network reconfiguration, computing and storage resource re-distribution, and consecutive control of latency. Suspicious event detection may indicate network vulnerability, cloud-service overload or a cybersecurity incident. The multidimensional latency analysis may lead to better geographic resource allocation on the side of Cloud-service users or tenants, even dynamically re-adjusting in real time. As we also plan to add other CSPs (i.e. a fourth dimension) to the measurements, real-time CSP benchmarking of Cloud-service latency can be deployed and again used for troubleshooting, comparison or optimal real-time resource allocation among different CSPs on the side of users or tenants.

The platform could be enhanced to greater precision by including many more Vantage Points or CSP deployments, however, careful consideration would have to be given to scalability of both the measurements and interpretation. Furthermore, advanced data mining techniques may be utilized to deepen the insights over the simple methods of descriptive statistics that have been used in this work for the automated time-series analysis. Unsupervised learning may be used to define normal Cloud behavior and design the best technique for detection of a variety of suspicious events, incorporating co-incidence between detected events by multiple Vantage Points or on multiple layers.

Last but not least, all the measurements data are freely available online to the research community, for verification and experimentation. Both the archived and near-real-time data can be found at http://claudit.feld.cvut.cz.

Acknowledgments

We kindly thank CESNET z.s.p.o for providing access to the PlanetLab infrastructure and Microsoft corporation for providing the Windows Azure Academic license. This work is supported by the Czech Educational and Scientific Network (CESNET), grant No. 497/2013. This work was also supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS15/153/OHK3/2T/13.

References

[1] O. Tomanek, L. Kencl, Claudit: planetary-scale cloud latency auditing platform, Cloud Networking (CloudNet), IEEE, 2013.
[2] P. Mulinka, L. Kencl, Learning from cloud latency measurements, Communication Workshop (ICCW), ICC 2015, IEEE, 2015.

[3] Planetlab, http://www.planet-lab.org/.
[4] Microsoft azure, https://azure.microsoft.com/.
[5] R. Jain, The Art of Computer Systems Performance Analysis, John Wiley & Sons Chichester, 1991.
[6] J.-Y. Le Boudec, Performance Evaluation Of Computer And Communication Systems, EPFL Press, 2011.
[7] J.-Y. Le Boudec, P. Thiran, Network Calculus: A Theory of Deterministic Queuing Systems for the Internet, Springer, 2001.
[8] Y. Jiang, Y. Liu, Stochastic Network Calculus, Springer, 2008.
[9] J. Bennett, K. Benson, A. Charny, W. Courtney, J.-Y. Le Boudec, Delay jitter bounds and packet scale rate guarantee for expedited forwarding, IEEE/ACM Trans. Netw. 10 (2002) 529–540.
[10] C. Demichelis, P. Chimento, IP packet delay variation metric for IP performance metrics, IETF (2002).
[11] J.-Y. Le Boudec, Application of network calculus to guaranteed service networks, IEEE Trans. Inf. Theory 44 (1988) 1087–1096.
[12] M. Menth, R. Martin, J. Charzinski, Capacity overprovisioning for networks with resilience requirements, SIGCOMM, ACM, 2006.
[13] V. Paxson, S. Floyd, Wide area traffic: the failure of Poisson modeling, IEEE/ACM Trans. Netw. 3 (1995) 226–244.
[14] V. Paxson, Empirically derived analytic models of wide-area TCP connections, IEEE/ACM Trans. Netw. 2 (1994) 316–336.
[15] V.E. Paxson, Measurements and analysis of end-to-end Internet dynamics, Berkeley, 1997 Ph.D. thesis.
[16] R. Kay, Pragmatic network latency engineering fundamental facts and analysis, 2009White Paper.
[17] J. Aikat, J. Kaur, F.D. Smith, K. Jeffay, Variability in TCP round-trip times, ACM SIGCOMM, 2003.
[18] Y. Xu, Z. Musgrave, B. Noble, M. Bailey, Bobtail: avoiding long tails in the cloud., NSDI, USENIX, 2013.
[19] J. Li, N.K. Sharma, D.R. Ports, S.D. Gribble, Tales of the tail: Hardware, OS, and application-level sources of tail latency, SoCC, ACM, 2014.
[20] J. Dean, L.A. Barroso, The tail at scale, Commun. ACM 56 (2013) 74–80.
[21] A.B. Downey, Evidence for long-tailed distributions in the Internet, SIGCOMM Workshop on Internet Measurement, ACM, 2001.
[22] L.A. Barroso, Warehouse-scale computing: entering the teenage decade, ACM SIGARCH, 2011.
[23] L.A. Barroso, J. Clidaras, U. Hölzle, The Datacenter as a Computer: An introduction to the design of warehouse-scale machines, Synthesis Lectures on Computer Architecture, Morgan & Claypool Publishers, 2013.
[24] Y.A. Wang, C. Huang, J. Li, K.W. Ross, Estimating the performance of hypothetical cloud service deployments: a measurement-based approach, INFOCOM, IEEE, 2011.
[25] M. Kwon, A tutorial on network latency and its measurements, IGI Global, 2015.
[26] K.-T. Chen, Y.-C. Chang, P.-H. Tseng, C.-Y. Huang, C.-L. Lei, Measuring the latency of cloud gaming systems, in: ACM International Conference on Multimedia, 2011.
[27] M. Dhawan, J. Samuel, R. Teixeira, C. Kreibich, M. Allman, N. Weaver, V. Paxson, Fathom: a browser-based network measurement platform, ACM IMC, 2012.
[28] O. Vondrous, P. Macejko, Z. Kocur, FlowPing-the new tool for throughput and stress testing, Adv. Electr. Electron. Eng. 13 (5) (2015) 516–521.
[29] N. Spring, L. Peterson, A. Bavier, V. Pai, Using Planetlab for network research: myths, realities, and best practices, ACM SIGOPS, 2006.
[30] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, R. Chaiken, The nature of datacenter traffic: measurements & analysis, ACM IMC, 2009.
[31] T. Benson, A. Akella, D.A. Maltz, Network traffic characteristics of data centers in the wild, ACM IMC, 2010.
[32] A. Pathak, Y.A. Wang, C. Huang, A. Greenberg, Y.C. Hu, R. Kern, J. Li, K.W. Ross, Measuring and evaluating TCP splitting for cloud services, PAM, Springer, 2010.
[33] M. Marshak, H. Levy, Evaluating web user perceived latency using server side measurements, Elsevier Comput. Commun. 26 (2003) 872–887.
[34] A. Abdelkefi, Y. Jiang, B.E. Helvik, G. Biczók, A. Calu, Assessing the service quality of an Internet path through end-to-end measurement, Elsevier Comput. Netw. (2014).
[35] A. Marnerides, A. Schaeffer-Filho, A. Mauthe, Traffic anomaly diagnosis in internet backbone networks: a survey, Elsevier Comput. Netw. (2014).
[36] R. Minnear, Latency: The Achilles heel of cloud computing, 2011. Cloud Computing Journal
[37] J. Danihelka, L. Kencl, Collaborative 3D environments over windows Azure, IEEE Mobile Cloud, 2013.
[38] Y. Tsang, M. Coates, R.D. Nowak, Network delay tomography, IEEE Trans. Signal Process. 51 (2003) 2125–2136.
[39] H.V. Madhyastha, T. Anderson, A. Krishnamurthy, N. Spring, A. Venkataramani, A structural approach to latency prediction, SIGCOMM, ACM, 2006.
[40] K.P. Gummadi, S. Saroiu, S.D. Gribble, King: Estimating latency between arbitrary Internet end hosts, ACM SIGCOMM Workshop on Internet measurement, 2002.
[41] M. Calder, A. Flavel, E. Katz-Bassett, R. Mahajan, J. Padhye, Analyzing the performance of an anycast CDN, ACM IMC, 2015.
[42] Y.-C. Chiu, B. Schlinker, A.B. Radhakrishnan, E. Katz-Bassett, R. Govindan, Are we one hop away from a better internet? ACM IMC, 2015.
[43] K. Johnson, J. Carr, M. Day, M. Kaashoek, The measured performance of content distribution networks, Elsevier Comput. Commun. 24 (2001) 202–206.
[44] Z. Hu, L. Zhu, C. Ardi, E. Katz-Bassett, H.V. Madhyastha, J. Heidemann, M. Yu, The need for end-to-end evaluation of cloud availability, Passive and Active Measurement, Springer, 2014.
[45] S.R. Smoot, N.K. Tan, Private Cloud Computing: Consolidation, Virtualization, and Service-Oriented Infrastructure, Morgan Kaufmann, 2011.
[46] M. Satyanarayanan, P. Bahl, R. Caceres, N. Davies, The case for VM-based cloudlets in mobile computing, Perv. Comput. Mag. IEEE 8 (2009) 14–23.
[47] T. Verbelen, P. Simoens, F. De Turck, B. Dhoedt, Cloudlets: bringing the cloud to the mobile user, ACM workshop on Mobile cloud computing and services, 2012.
[48] J. Perry, A. Ousterhout, H. Balakrishnan, D. Shah, H. Fugal, Fastpass: a centralized zero-queue datacenter network, SIGCOMM, ACM, 2016.
[49] M. Chowdhury, M. Zaharia, J. Ma, M.I. Jordan, I. Stoica, Managing data transfers in computer clusters with orchestra, SIGCOMM, ACM, 2011.
[50] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, M. Yasuda, Less is more: trading a little bandwidth for ultra-low latency in the data center, NSDI, USENIX, 2012a.
[51] M. Alizadeh, A. Greenberg, D.A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, M. Sridharan, Data center TCP (DCTCP), ACM SIGCOMM, 2011b.
[52] C. Wilson, H. Ballani, T. Karagiannis, A. Rowstron, Better never than late: meeting deadlines in datacenter networks, SIGCOMM, ACM, 2011.
[53] B. Vamanan, J. Hasan, T. Vijaykumar, Deadline-aware datacenter TCP (D2TCP), SIGCOMM, ACM, 2012.
[54] C.-Y. Hong, M. Caesar, P. Godfrey, Finishing flows quickly with preemptive scheduling, SIGCOMM, ACM, 2012.
[55] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, S. Shenker, pFabric: Minimal near-optimal datacenter transport, SIGCOMM, ACM, 2013.
[56] C. Lee, K. Jang, S. Moon, Reviving delay-based TCP for data centers, SIGCOMM 2012, ACM, 2012.
[57] A. Lakhina, M. Crovella, C. Diot, Diagnosing network-wide traffic anomalies, SIGCOMM, ACM, 2004.
[58] M. Zhang, C. Zhang, V. Pai, L. Peterson, R. Wang, Planetseer: Internet path failure monitoring and characterization in wide-area services, USENIX OSDI, 2004.
[59] J.D. Brutlag, Aberrant behavior detection in time series for network monitoring, USENIX LISA, 2000.
[60] F. Feather, D. Siewiorek, R. Maxion, Fault detection in an ethernet network using anomaly signature matching, ACM SIGCOMM, 1993.
[61] B. Krishnamurthy, S. Sen, Y. Zhang, Y. Chen, Sketch-based change detection: methods, evaluation, and applications, ACM IMC, 2003.
[62] G. Aceto, A. Botta, W. De Donato, A. Pescapè, Cloud monitoring: a survey, Elsevier Comput. Netw. 57 (2013) 2093–2115.
[63] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, D. Epema, A performance analysis of EC2 cloud computing services for scientific computing, Cloud Computing, Springer, 2010.
[64] Z. Hill, M. Humphrey, A quantitative analysis of high performance computing with Amazon's EC2 infrastructure: The death of the local cluster? Grid, IEEE, 2009.
[65] E. Walker, Benchmarking amazon EC2 for high-performance scientific computing, USENIX LOGIN, 2008.
[66] K.-p. Chan, F. Ada Wai-chee, Efficient time series matching by wavelets, Data Engineering, 1999.
[67] I. Popivanov, R.J. Miller, Similarity search over time-series data using wavelets, Data Engineering, IEEE, 2002.
[68] J. Lin, Y. Li, Finding structural similarity in time series data using bag-of-patterns representation, Computer Science, Springer, 2009.
[69] Dyn research, research.dyn.com/.
[70] ThousandEyes: network monitoring software, https://www.thousandeyes.com/.
[71] Y. Zhu, H. Eran, D. Firestone, C. Guo, M. Lipshteyn, Y. Liron, J. Padhye, S. Raindel, M.H. Yahia, M. Zhang, Congestion Control for Large-Scale RDMA Deployments, ACM SIGCOMM, 2015.
[72] G. Evenden, Proj. 4–cartographic projections library, 1990. Source code and documentation available from trac. osgeo.org/proj.
[73] G. Mahlknecht, Greg's cable map, http://www.cablemap.info/.

Ondrej Tomanek is a third-year Ph.D. student and a junior researcher at the Department of Telecommunications Engineering, FEE, Czech Technical University in Prague (CTU). He obtained his MSc. in Software Engineering at CTU and MBA at Escuda Superior de Marketing y Administracion. His Ph.D. research focuses Cloud Computing Network Architectures and Performance Engineering. Ondrej has experience from both software and systems industries, including an internship at Cisco Systems Inc., San Jose, CA, USA.

Pavol Mulinka is a PhD student on CTU FEE with research topic Cloud Computing Efficiency from Networking Perspective. He received his MSc. in Telecommunications on Slovak University of Technology in Bratislava - Faculty of Electrical Engineering and Information Technology. His Master thesis topic was measurement of glottal periods of the human voice and identification of the speaker. He was a co-researcher on a Cisco Research grant on automation of suspicious event detection on a Cloud measurements. His experience includes network implementation and design for global networking companies, e.g. Verizon, AT&T, HP.

Lukas Kencl is director of the R&D Centre for Mobile Applications (RDC) at the Department of Telecommunications Engineering, FEE, Czech Technical University in Prague (CTU). He obtained a Ph.D. in Communication Networks from EPFL, Switzerland, in 2003. Prior to CTU, he was with Intel Research, Cambridge, UK (2003–2006) and with IBM Research –Zurich (1999–2003). His research focuses on cloud and network services, network architecture and system optimization, and on network privacy and security. He is co-inventor of 5 patents and holder of many industrial awards (Electrolux, IBM Research, Cisco Research, Microsoft Research).