The Need for End-to-End Evaluation of Cloud Availability

Zi Hu[1,2], Liang Zhu[1,2], Calvin Ardi[1,2], Ethan Katz-Bassett[1], Harsha V. Madhyastha[3], John Heidemann[1,2], and Minlan Yu[1]

[1] USC/CS Dept.
[2] USC/ISI
[3] U. of California, Riverside

Abstract. People's computing lives are moving into the cloud, making understanding cloud availability increasingly critical. Prior studies of Internet outages have used ICMP-based pings and traceroutes. While these studies can detect network availability, we show that they can be inaccurate at estimating *cloud* availability. Without care, ICMP probes can *underestimate* availability because ICMP is not as robust as application-level measurements such as HTTP. They can *overestimate* availability if they measure reachability of the cloud's edge, missing failures in the cloud's back-end. We develop methodologies sensitive to five "nines" of reliability, and then we compare ICMP and end-to-end measurements for both cloud VM and storage services. We show case studies where one fails and the other succeeds, and our results highlight the importance of application-level retries to reach high precision. When possible, we recommend end-to-end measurement with application-level protocols to evaluate the availability of cloud services.

1 Introduction

Cloud computing is a distributed computing paradigm that allows users to easily access and configure remote computing resources in a scalable manner. As the cloud grows in importance, it will host more applications and services from the small (such as new and developing web applications) to the large (Amazon, Netflix, etc.).

As we depend on them more and more, services that run in the cloud need to be highly available. Despite this need and news reports highlighting major cloud outages [17], there have been few *systematic*, third party studies of how reliable the cloud actually is. While recent systems might use one [25] or multiple [2] cloud providers to improve reliability, there is a poor understanding of reliable methods to externally and empirically measure cloud reliability.

Many general network availability and measurement studies use ICMP-based methodologies [14,15,12,23,10], sometimes focusing on routing problems [15] or outages in edge networks [12,23]. Studies likely use ICMP because more routers respond to it than to other types of probes [18,19] and because ICMP probes are less likely to elicit complaints [24,18]. However, distrust of ICMP in the network

M. Faloutsos and A. Kuzmanovic (Eds.): PAM 2014, LNCS 8362, pp. 119–130, 2014.

Table 1. Datasets used in this paper

start	duration (days)	target service	(provider)	sources (VPs)	method tries/interval
2013-03-11	+33	VM	(Amazon)	23	3× / 10 min.
2013-03-11	+33	storage	(Amazon, Google, Microsoft)	23	3× / 10 min.
2013-06-18	+17	VM	(Amazon)	54	9× / 11 min.
2013-06-18	+75	storage	(Amazon, Google, Microsoft)	54	9× / 11 min.

operator community [1] calls into question the accuracy and reliability of using only ICMP to measuring availability. While effective for network measurements, ICMP is not perfect, and care must be taken to consider filtering, rate limiting, and depreferential service.

The contribution of this paper is to develop and compare mechanisms to measure cloud reliability. We show that ICMP-based measurements are inaccurate at measuring cloud availability and that *end-to-end measurements* are *necessary* to establish cloud availability.

We first compare ICMP and HTTP to measure cloud reliability at the network and application levels, and then apply them to several cloud VM and storage services. We evaluate the effect of retries and show that ICMP has a higher loss rate than random packet loss alone predicts (Section 3). While ICMP and HTTP nearly always agree, they sometimes disagree. ICMP occasionally experiences a period of loss from some vantage points and thus will *overestimate* cloud outages—a weakness of the methodology. Less frequently, we see that HTTP probing shows outages that last for extended periods from some vantage points; ICMP would *underestimate* these outages because of its failure to reach the provided service. We conclude that, although application-level methods such as HTTP probing incur the cost of provisioning and accessing cloud resources, they are necessary to accurately assess cloud reliability.

2 Methodology

We use two methods to study availability: ICMP probes at the network level, and end-to-end probes with HTTP at the application level. We target both cloud VMs and storage services of three providers. Our work results in four datasets (Table 1), all available on request.

2.1 Outage Causes

We measure outages in *cloud services* by taking observations from many *vantage points* (VPs). Section 2.4 details our VP selection and infrastructure. To understand what these measurements tell us, we must consider potential sources of failure that can occur from the VP to the cloud. These problems may occur near the VP, in the network, near the cloud provider, at the cloud front-end, or inside the cloud infrastructure.

We see several possible failures: (1) DNS lookup failures; (2) routing problems, either near the VP, in the network, or at the provider; (3) random packet loss in the network; (4) rate limiting, either near the VP, in the network, or at the provider; and (5) service outages inside the cloud infrastructure.

While all of these problems can interfere with use of the cloud, some, such as packet loss, are commonplace and the end user is responsible to recover from them. Others affect some measurements differently. Our goal is to understand how the choice of measurement methodologies emphasizes different failures.

2.2 Outage Detection at the Network and Application Level

We measure cloud status every 10 or 11 minutes (see Table 1), sending ICMP and HTTP probes with retries. We record the results from many vantage points. We consider the overall response to be positive if the initial probe or any of the retries succeeds.

For network-level tests, we send an ICMP echo request, considering only positive replies as successful, and lack of a reply or any error code as negative. For end-to-end testing, we retrieve a short file over HTTP with `curl`. A positive response is a HTTP status code of 200 OK; any other HTTP status code is a negative response. We record curl error codes to distinguish some failure cases.

In both cases, if the initial request fails, we try two and eight additional times for the datasets that begin on 2013-03-11 and 2013-06-18, respectively. We then record the result as I or $\overline{\text{I}}$ for ICMP success or failure, and H or $\overline{\text{H}}$ for HTTP. In the 2013-03-11 datasets, we do not do ICMP retries unless HTTP probes fail, in which case we then perform ICMP retries in conjunction with HTTP retries.

To diagnose problems, we observe the probe at the service itself (when possible), and we record ICMP and TCP traceroutes between the VP and service.

Since routing outages near the vantage point will skew our observations, we calibrate our measurements by probing two *control sites* at USC/ISI and University of Washington. We probe these sites with the same method as probing the cloud. We discard cloud measurements when either of these control sites is unavailable.

2.3 Targets: Cloud Storage and VMs

We probe two cloud targets: virtual machines (VMs) and online storage.

Virtual Machines: We test VMs at Amazon only. Google's VM service is not yet public, and Microsoft VMs filter inbound and outbound ICMP traffic.

For Amazon VMs, we instantiate a `micro` VM on Amazon's Elastic Compute Cloud (EC2) running Ubuntu 12.04 in all eight regions (May 2013). We install `lighttpd` HTTP daemon and serve static, 1 kB files. We modify the firewall on each VM to allow all traffic. Each VM is given a public IP address. We probe this IP address directly. We expect both ICMP and HTTP probes to reach our VM at the kernel and application-level.

Storage: We test storage on three providers: Amazon Simple Storage Service (S3), Microsoft Azure, and Google Cloud Storage. Each provider exports an

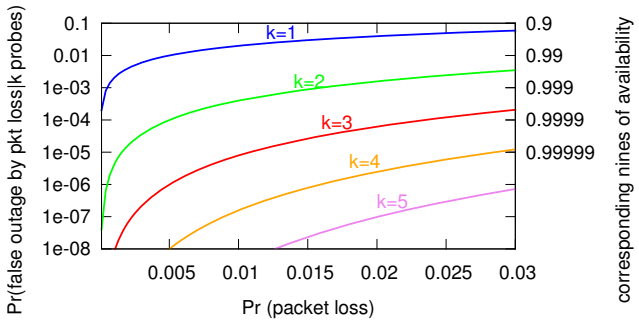

Fig. 1. Probability of false positive caused by random packet loss

HTTP-based storage abstraction. We store 1 kB files on all available regions in each provider.

For ICMP probes to storage, we ping the hostname in the URL of the stored object. We expect that this probe contacts only the front-end for the service. HTTP probes retrieve data from the storage back-end. Providers do not, in general, provide details about their back-end storage architecture, and we expect data to be replicated in each datacenter and often across datacenters. HTTP, however, is an end-to-end test for storage.

2.4 Sources: Vantage Points

We probe each of our targets from *vantage points* in PlanetLab [5], using 23 starting 2013-03-11 and 54 starting 2013-06-18. We limit the number of VPs to reduce cloud costs, and select them from universities around the world. We expect PlanetLab nodes to be well connected, allowing us to focus on cloud availability. We follow best practices in taking measurements from PlanetLab [24].

3 Evaluating the Need for Retries

A range of possible root causes can explain an outage (Section 2.1). To understand what measurement says about the cloud, we must first rule out mundane causes like packet loss.

While packet loss is rare, cloud outages are much rarer, so random packet loss will dominate careless observations. We next show that ICMP requires at least 5 retries, and even HTTP benefits from application-level retries in addition to kernel-level TCP retransmissions.

3.1 A Simple Analytic Model

Packet loss in the network can be correlated (burst losses due to congestion, filtering) and random (queue overflow over medium timescales). We limit distortion from congestive loss by spacing probes 2 s apart, avoiding most short-term

Pr(first try fails)		
Target	**ICMP**	**HTTP**
Amazon/VM	.00585	.00232
Amazon/storage	.00574	.00435
Google/storage	.00631	.00217

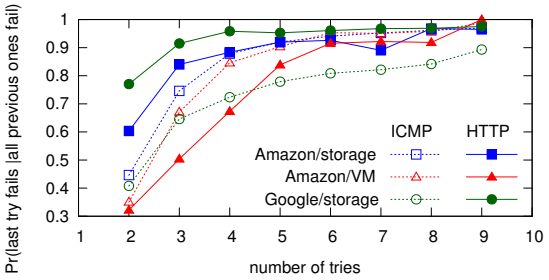

Fig. 2. Comparing loss and reties for each target and method. Left: table of probability *first* try fails. Right: conditional probability kth retry fails, given failure of prior tries. Dataset: VMs (2013-06-18+17), storage (2013-06-18+75).

outages [22,9]. Our probe rate (a few packets per second) is low enough to avoid rate limiting, although, in Section 4.1, we see some cases where *all* packets are dropped.

Having ruled out most sources of correlated loss, we next evaluate random loss. We first establish an analytic model of packet loss, assuming a fixed loss rate that affects packets independently. We then compare that to our experimental observations.

The curved lines in Figure 1 evaluate the probability of falsely inferring an outage caused by random packet loss, as a function of packet loss rate (the x-axis). We assume k tries for each probe and declare the service down when all tries fail. For packet loss p, we model loss of the request or response:

$$\Pr(\text{outage} \mid k \text{ probes}) = (p + (1-p)p)^k \tag{1}$$

Without retries ($k = 1$), the false outage rate approximates the loss rate. Since wide area packet loss can be around 1%, *measurement without retries will show false outages and skew estimates of cloud reliability.* Fortunately, if we assume packet loss is independent, then a few retries drive the false outage rate well below typical cloud outage rates. For example, with three tries and 1% packet loss, probe loss will be around 10^{-5}, or five nines of availability. If we assume network loss rates peak at a few percent, 4–6 tries may be appropriate. Our data starting 2013-06-18 uses 9 retries to rule out random loss.

3.2 ICMP Measurements

We next compare our model against experimental results for ICMP. The dotted lines in Figure 2 show the probability the kth try fails if all previous $k-1$ tries failed. We evaluate this by considering the first k tries from each observation.

With ICMP, we see that retries help. An initial loss is followed by a second loss only 35-45% of the time, so 55–65% of the time the second try succeeds, suggesting that the first try was random loss. This effect diminishes with more retries, generally plateauing around 5 or 6 retries. When we compare long-term

observed loss rates to short-term ICMP retries, we see that losses are generally *more* correlated than predicted by our analytic model. That is, it usually takes more retries to recover from an initial loss than are predicted, but, with enough retries, we often recover from an initial loss.

3.3 Retries and HTTP Probes

For HTTP-based probing, we retry at the application level, but the kernel also does retries for the TCP connection. Our HTTP client (curl) has a 10 s application timeout. The OS (Linux-2.6.32) does 3 SYN transmissions in this time, providing 2 network-level retries "for free" for each application retry.

We see this benefit in the Figure 2's left table, where single-try HTTP losses rates are much lower than ICMP. Kernel-level retries help even with application retries, as seen in Figure 2's right graph where the basic HTTP failure rate for Amazon/storage and Google/storage is half that of ICMP. However, even HTTP benefits from multiple application-level retries before the conditional benefit of additional tries plateaus. We recommend 6 application-level tries even for HTTP probes.

Application-level probes show even higher levels of conditional failure than network-level, with 50% of second HTTP attempts failing on average, presumably because of the additional kernel-level retries. However, this result means that 50% of second attempts *succeed*—application-level failures are sometimes transient. We thus recommend retries even for end-to-end tests.

4 Comparing Network and Application-Level Probing

We next compare network- and application-level probes to judge cloud availability. We use our control sites to rule out problems near vantage points, and we use sufficient retries to avoid effects of random packet loss and transient network issue in the middle, leaving outages at or near the cloud as the primary problem. Cloud services are made up of Internet-facing front-ends with sophisticated back-end clusters. In some cases, ICMP may be handled by the front-ends, while HTTP's end-to-end tests reach into the back-end. Our goal is to compare this difference. While the protocols almost always agree, there are many small disagreements. We next show several causes of disagreement through representative examples.

4.1 Comparing ICMP and HTTP Probing

We first compare ICMP and HTTP probing, showing representative examples of several causes of disagreement. These results show the need for end-to-end measurement with application-level protocols.

Method Agreement: Figure 3 shows the percent of disagreement between ICMP and HTTP over 17 days. Both approaches give the same result in the vast majority of measurement rounds. They disagree in at most 3% of rounds

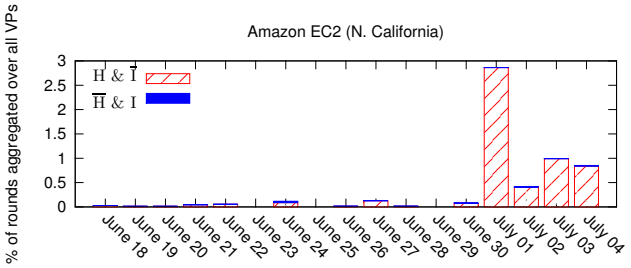

Fig. 3. Quantifying disagreements between HTTP and ICMP probes. This includes either HTTP success and ICMP failure (red striped bar) or HTTP failure and ICMP success (blue bar). Dataset: 2013-06-18+17.

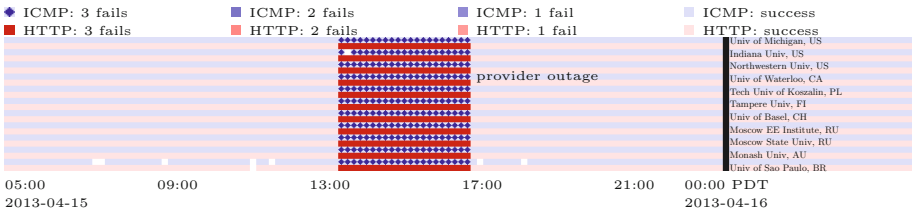

Fig. 4. Strip chart: Amazon VM (Singapore). Dataset: 2013-03-11+33.

in a given day, and, on most days, they disagree in 0.5% or less of the rounds. The high agreement is because the monitored service is almost always up, and both methods detect it as such. On three days (June 23, 25, 29), we see complete agreement (no outages). We also see that both methods report outages on some days (for example, June 18 and 24).

To illustrate the details of an outage, Figure 4 shows a *strip chart* for a provider-confirmed outage at one Amazon EC2 site [3]; ICMP and HTTP report the outage consistently. In this chart, each column of data shows one round of measurements (with 24-hour boundaries as vertical black lines), and each pair of rows shows ICMP and HTTP observations from one VP (the blue top is ICMP, and the lower red is HTTP). Light colors represent successful probes, medium colors represent failures of some tries (but eventual success). Dark blue diamonds show ICMP-determined outages (all ICMP tries fail); dark red squares show an HTTP outage (all HTTP tries fail). White areas show cases where one of the control nodes failed to respond to either ICMP or HTTP, or where we are unable to upload data to our collection site.

As a second example, Figure 5 shows a case where both ICMP and HTTP report intermittent failures from one VP. We see intermittent problems from Koszalin University of Technology in Poland to Amazon S3's Singapore site. In fact, we observe intermittent failures between that source and destination pair for the entire duration of our measurement. This case shows that sometimes network problems between the VP and cloud (such as routing problems) persist for some time. Both ICMP and HTTP report outages for this VP.

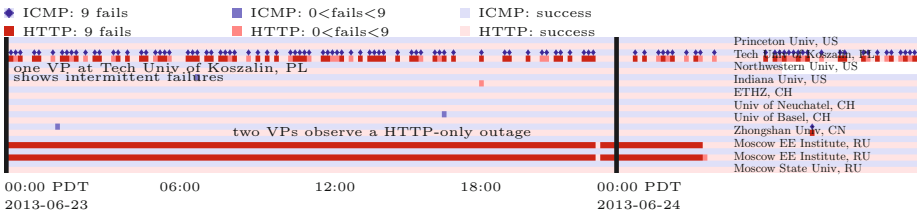

Fig. 5. Strip chart: Amazon S3 (Singapore). Dataset: 2013-06-18+75.

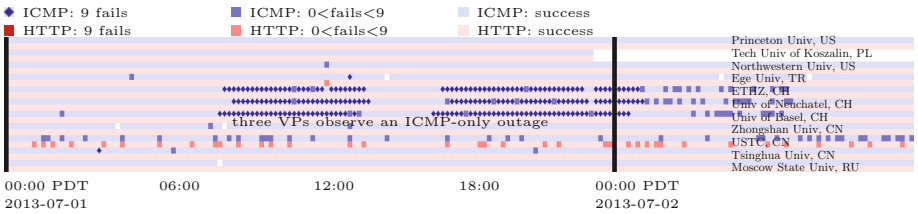

Fig. 6. Strip chart: Amazon VM (N. California). Dataset: 2013-06-18+17.

Method Disagreement: However, HTTP and ICMP probes can also show disagreement. We see disagreement in 0.01% to 3% of observations, as shown by the stacked bars in Figure 3. The source of the disagreement is usually ICMP failures with HTTP success (the bottom, red striped bars), but sometimes ICMP succeeds and HTTP fails (the much smaller blue bars on top).

As a first example where ICMP fails but HTTP succeeds, Figure 6 shows a case where three Swiss universities could not reach Amazon/VM in California. We see with `tcpdump` that filtering happens on the return path. Since the three VPs reporting this ICMP-only outage are at different sites in the same country, we hypothesize that reverse path changes–possibly to a path that filtered ICMP–caused the outage. In this case, despite ICMP reporting multiple outages, we can still fetch the data in the cloud, meaning that ICMP over-counts outages.

We also see the reverse case, where HTTP fails but ICMP succeeds, overestimating cloud availability. Figures 5 and 7 show two VPs in Russia observing an HTTP-only outage to both Amazon S3 and EC2 in Singapore. We observe route changes before and after the outage, and we confirm our probes (here TCP SYNs) reach the VM and replies are sent but do not reach the VP. We cannot confirm the root cause for this outage, although we guess there may be problems in a load-balancer at the cloud's edge.

4.2 Differences between Probing VMs and Storage

In addition to comparing network and application probing, we also probe different targets: virtual machines and storage. The *target* affects what the probing

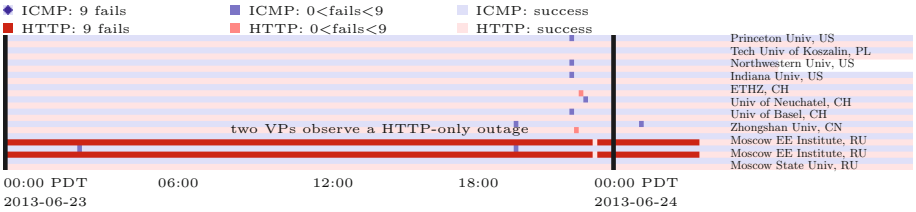

Fig. 7. Strip chart: Amazon VM (Singapore). Dataset: 2013-06-18+17.

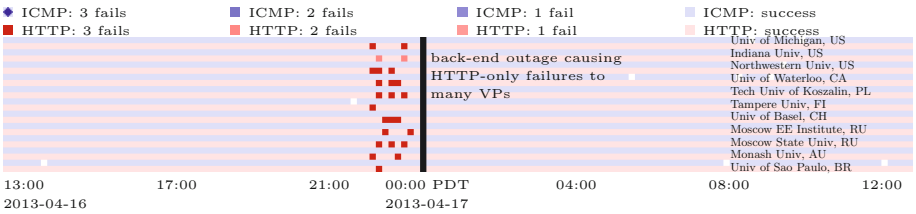

Fig. 8. Strip chart: Amazon S3 (Japan). Dataset: 2013-03-11+33.

mechanism sees. We next show that *end-to-end measurements are essential to observe outages in cloud storage* and other systems with complex back-ends.

Figure 8 shows an outage for Amazon S3 in Tokyo on April 16. Only HTTP measurements detect this outage; ICMP reports that all is well. This outage is confirmed by Amazon outage report [3].

To understand this discrepancy, we must consider what exactly ICMP and HTTP measure when observing a storage system. For storage systems, a user accesses a front-end system with a URL, but data retrieval exercises the back-end storage system. ICMP measures only to this front-end, while HTTP provides an end-to-end test, verifying that the storage system is functioning (at least for one stored object). We can therefore infer this outage was *inside* Amazon's storage system and not in the network from the VP to the datacenter. We conclude that *ICMP will overestimate the availability of cloud storage*, supporting our recommendation for *end-to-end outage testing for higher-level cloud services.*

To understand the root cause of these storage outages, we next use errors reported by our storage retrieval tool (curl). We look at the error returned from each failed attempt of storage retrieval from the 2013-06-18+75 storage dataset. We see that most of these (87%) are due to DNS lookup failure, with the second largest cause (10%) due to TCP connection setup failure. In contrast, for VMs (dataset: 2013-06-18+17), almost all failures (99%) are caused by TCP connection setup failures. All of the storage systems use DNS to map a request into the storage back-end systems. These DNS failures can represent either random loss of the request in the network, or failure of the storage system's DNS mechanism to identify a storage server. Since applications that use cloud storage will follow a similar process as curl which is used in our measurements, these types of outages reflect intermittent problems that should be reported.

Based on our measurement results, we show that ICMP probes can be inaccurate at estimating cloud availability. ICMP is not as robust as application-level measurements such as HTTP. ICMP's failure to solicit a response does not mean that the service is down, so ICMP can *underestimate* availability. At the same time, ICMP can also *overestimate* availability as it measures reachability of the cloud's edge, missing failures in the cloud's back-end. We therefore suggest using application-level probes such as HTTP rather than network-level probes to evaluate cloud reliability; the examples in this section present the motivation for a longer-term study.

5 Related Work

Our work builds upon previous efforts in two broad areas: characterizing the Internet's availability and measurement of cloud services.

Internet Availability: To date, a large number of measurement studies have probed the Internet from a distributed set of vantage points in order to characterize the Internet's availability. While some studies rely on passive measurements of Internet traffic to detect the onset of outages (for example, [27,4]), such monitoring is possible only by instrumenting a popular service. Therefore, most measurement studies of the Internet's availability have instead relied on continuous probing of a large number of end-hosts. These studies have focused on identifying outages [14,23], network failures [6,28], characterizing the typical duration of outages [14,10,15], and pinpointing their root causes [8,13]. Some studies have paid particular attention to measurement methodology of paths [7] and of the edge [23]. However, all of these studies have in common a reliance on ICMP-based probes. While ICMP may be necessary for Internet-wide studies, our results show that application-level measurements should be used when possible, and they are essential to understanding availability of cloud services, where ICMP-based probing can both over- and under-predict outages.

Measurements of Cloud Services: Some recent work has begun on measure and characterize the performance offered by cloud services. CloudCmp measures the compute, storage, and network performance offered by various cloud services with the goal of enabling application providers to choose from these services [16]. Others have performed measurements of cloud services in order to determine when it is beneficial for applications to be hosted in the cloud [11,21]. To the best of our knowledge, we are the first to investigate the methodology of active monitoring of the availability of cloud services. Motoyama et al. pursue a complementary approach of inferring outages from indirect information in Twitter posts [20]; further investigation is necessary to correlate outages in web services to outages of the underlying cloud services on which they are deployed.

6 Conclusion

This paper compared network and application level measurements sensitive of cloud service availability. We compare ICMP and HTTP over two types of services (VMs and storage) and three providers. We find that ICMP can both over- and under-report outages, suggesting that it is important to use end-to-end measures (such as HTTP) to best characterize cloud service availability. Our study raises concerns about the use of ICMP for monitoring availability and suggests that earlier results should be revisited. We are using these approaches as part of a long-term study of cloud availability. Part of our ongoing work is to understand cloud availability in order to deploy highly-available systems at low cost across various cloud providers, just as existing work uses multiple providers to provide low latency at low cost [26].

References

1. Outages mailing list. Mailing List, `http://www.outages.org`
2. Abu-Libdeh, H., Princehouse, L., Weatherspoon, H.: RACS: A case for cloud storage diversity. In: SoCC (2010)
3. Amazon. AWS Service Health Dashboard, `http://status.aws.amazon.com/`
4. Choffnes, D.R., Bustamante, F.E., Ge, Z.: Crowdsourcing service-level network event monitoring. In: SIGCOMM (2010)
5. Chun, B., Culler, D., Roscoe, T., Bavier, A., Peterson, L., Wawrzoniak, M., Bowman, M.: PlanetLab: An overlay testbed for broad-coverage services. In: SIGCOMM CCR (2003)
6. Cunha, I., Teixeira, R., Feamster, N., Diot, C.: Measurement methods for fast and accurate blackhole identification with binary tomography. In: IMC (2009)
7. Cunha, I., Teixeira, R., Veitch, D., Diot, C.: Predicting and tracking internet path changes. In: SIGCOMM (2011)
8. Dhamdhere, A., Teixeira, R., Dovrolis, C., Diot, C.: Netdiagnoser: troubleshooting network unreachabilities using end-to-end probes and routing data. In: CoNEXT (2007)
9. Flach, T., Dukkipati, N., Terzis, A., Raghavan, B., Cardwell, N., Cheng, Y., Jain, A., Hao, S., Katz-Bassett, E., Govindan, R.: Reducing web latency: the virtue of gentle aggression. In: SIGCOMM (2013)
10. Gummadi, K.P., Madhyastha, H.V., Gribble, S.D., Levy, H.M., Wetherall, D.: Improving the reliability of Internet paths with one-hop source routing. In: OSDI (2004)
11. Hajjat, M., Sun, X., Sung, Y.-W.E., Maltz, D., Rao, S., Sripanidkulchai, K., Tawarmalani, M.: Cloudward bound: planning for beneficial migration of enterprise applications to the cloud. In: SIGCOMM (2010)
12. Heidemann, J., Pradkin, Y., Govindan, R., Papadopoulos, C., Bartlett, G., Bannister, J.: Census and survey of the visible Internet. In: IMC (2008)
13. Javed, U., Cunha, I., Choffnes, D.R., Katz-Bassett, E., Krishnamurthy, A., Anderson, T.: PoiRoot: Investigating the root cause of interdomain path changes. In: SIGCOMM (2013)
14. Katz-Bassett, E., Madhyastha, H.V., John, J.P., Krishnamurthy, A., Wetherall, D., Anderson, T.: Studying black holes in the Internet with Hubble. In: NSDI (2008)

15. Katz-Bassett, E., Scott, C., Choffnes, D.R., Cunha, I., Valancius, V., Feamster, N., Madhyastha, H.V., Anderson, T., Krishnamurthy, A.: LIFEGUARD: Practical repair of persistent route failures. In: SIGCOMM (2012)
16. Li, A., Yang, X., Kandula, S., Zhang, M.: Cloudcmp: comparing public cloud providers. In: IMC (2010)
17. Lohr, S.: Amazon's trouble raises cloud computing doubts (April 2011), http://www.nytimes.com/2011/04/23/technology/23cloud.html
18. Luckie, M., Hyun, Y., Huffaker, B.: Traceroute probe method and forward IP path inference. In: IMC (2008)
19. Madhyastha, H.V., Isdal, T., Piatek, M., Dixon, C., Anderson, T., Krishnamurthy, A., Venkataramani, A.: iPlane: An information plane for distributed services. In: OSDI (2006)
20. Motoyama, M., Meeder, B., Levchenko, K., Voelker, G.M., Savage, S.: Measuring online service availability using Twitter. In: WOSN (2010)
21. Palankar, M.R., Iamnitchi, A., Ripeanu, M., Garfinkel, S.: Amazon S3 for science grids: a viable solution? In: DADC (2008)
22. Paxson, V.: End-to-end internet packet dynamics. In: SIGCOMM (1997)
23. Quan, L., Heidemann, J., Pradkin, Y.: Trinocular: understanding internet reliability through adaptive probing. In: SIGCOMM (2013)
24. Spring, N., Peterson, L., Bavier, A., Pai, V.: Using PlanetLab for network research: Myths, realities, and best practices. SIGOPS Oper. Syst. Rev. (2006)
25. Wood, T., Cecchet, E., Ramakrishnan, K.K., Shenoy, P., van der Merwe, J., Venkataramani, A.: Disaster recovery as a cloud service: economic benefits & deployment challenges. In: HotCloud (2010)
26. Wu, Z., Butkiewicz, M., Perkins, D., Katz-Bassett, E., Madhyastha, H.V.: Spanstore: Cost-effective geo-replicated storage spanning multiple cloud services. In: SOSP 2013 (2013)
27. Zhang, M., Zhang, C., Pai, V., Peterson, L., Wang, R.: PlanetSeer: Internet path failure monitoring and characterization in wide-area services. In: OSDI (2004)
28. Zhang, Z., Zhang, Y., Hu, Y.C., Mao, Z.M., Bush, R.: iSPY: Detecting IP prefix hijacking on my own. In: SIGCOMM (2008)