Observed Availability of Cloud Services

Santeri Paavolainen

Master's Thesis
UNIVERSITY OF HELSINKI
Department of Computer Science

Helsinki, June 5, 2016

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Santeri Paavolainen

Työn nimi — Arbetets titel — Title

Observed Availability of Cloud Services

Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's Thesis	June 5, 2016	53

Tiivistelmä — Referat — Abstract

Cloud computing is used widely and is going to be used even more in the future. Many internet-based services are now designed to be "cloud native" using architectures that allow them to take advantage of the scalability of the underlying cloud infrastructure allowing customer services to meet potentially rapid changes in customer demand. While there are many customers successfully leveraging cloud services for their benefit, the use cloud computing has also drawn critique on its other aspects such as its reliability and security. Although issues of security and operational cost benefits have been studied and actively marketed by the major cloud infrastructure service vendors, the question of service reliability and more specifically, availability of cloud services is less researched in academia and also less publicised by the cloud vendors themselves.

This study takes a look at the service availability of cloud infrastructure services. The study focuses on the largest public cloud infrastructure provider e.g. Amazon Web Services, and uses publicly available incident information to analyse outages from multiple viewpoints. The use of publicly available information at has allowed this work to analyse a wider selection of services than earlier studies, but also does limit the scope of outages that can be analysed to relatively large-scale outages.

The overall result is that Amazon Web Services' services during the analysis period of June 5th 2013 to June 4th 2014 are reliable services with an overall availability of 99.983% over all of the services included in this study. During the analysis period there was a total of 139 separate outage events where an average outage event lasted 130 ± 20 minutes. EC2 and RDS, two of the services with known Service Level Agreement availability target, meet their contractual availability targets by a comfortable margin with both having over 99.9999% availability when measured in comparable units to the SLA's target of 99.95% availability.

ACM Computing Classification System (CCS):

General and reference Empirical studies,
Computer systems organization Availability,
Computer systems organization Reliability

Avainsanat — Nyckelord — Keywords
cloud computing, reliability, availability, empirical study, Amazon Web Services, AWS

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1 Introduction 1

1.1 Background . . . 1

1.2 Problem Statement . . . 2

1.3 Related Work . . . 3

1.4 Outline . . . 6

2 Theory 6

2.1 What is Cloud Computing? . . . 6

2.2 Delivery Models of Cloud Services . . . 7

2.3 Reliability and Availability . . . 11

2.4 Cloud Failure Modes . . . 14

3 Data 16

3.1 Potential Data Sources . . . 16

3.1.1 Monitoring . . . 16

3.1.2 Media and Customer Reports . . . 17

3.1.3 Information Published by the Vendor . . . 19

3.2 Outage Reporting by AWS . . . 20

4 Methods 23

4.1 Overview . . . 23

4.2 Data Collection . . . 24

4.3 Message Processing . . . 24

4.4 Analysis . . . 25

4.5 Statistical Methods . . . 25

4.6 Infrastructure Data . . . 26

5 Analysis 26

5.1 Overview . . . 26

5.2 Messages . . . 27

5.3 Incidents . . . 28

5.4 Outages . . . 34

5.5 Availability . . . 35

6 Discussion 40

7 Further Work 42

A Service Level Agreement Availability vs. Zone Availability 51

B AWS Service Level Agreements 52

C Detailed Summaries of Messages, Incidents and Outages 53

1 Introduction

1.1 Background

Cloud computing has become an indispensable part of today's software ecosystem. Cloud computing enables companies of all size and industries as well as individual software developers and entrepreneurs an access to a large selection of global infrastructure, platform and application services without the need for skills or capital to operate or invest in the required infrastructure themselves. The scalability and resiliency of cloud computing services has many potential benefits for companies by for example allowing them to adapt to optimize operational costs by growing and shrinking their computing resource use to adapt to daily and monthly changes. Even small and young companies have now the possibility to meet rapid changes to their computing needs — Animoto for example was able to meet fast-paced viral growth of their user base by scaling up their cloud computing usage over fifty-fold within a few days [Bar08]. In another case NASA delivered a live video stream for over 120 000 simultaneous viewers without the need to put down long-term investment in the required computing and network capacity [Geo+13]. Companies such as Netflix are practical examples how cloud computing can be used as the platform to build networked consumer services with global reach.

The rapid growth of the cloud computing market [Lou14] has also highlighted many relevant concerns about its security, safety and reliability. While it relatively easy for an interested party to determine performance metrics of cloud resources such as CPUs, memory, disk and network, it is much more difficult to objectively evaluate the more qualitative metrics such as security and reliability. Some of the concerns can be be addressed by the vendor through gaining trusted security and process quality certifications (such as SOC 1, ISO 9001 and ISO 27017). While security incidents on major cloud services have been rare, on the side of service reliability there have been several high-profile outages [Coc12; Bil12; Mik12; Ama12c; Ama12a] that have raised awareness and questions on the reliability of cloud computing.

While anecdotes and media reports of cloud service outages make a good reading, we should be careful of putting too much trust on them. It is known that human perceptions of probabilities can be biased. For example, the availability heuristic causes humans to rely more on easily recalled events when estimating risks [SV02] — and in case of cloud computing it is likely to be easier to remember outages than situations where everything worked. Secondly even if media reports are examined methodically to avoid perception biases, one must realize that media itself is biased towards publishing large-scale outages — they make better headlines — which can lead to under-reporting of smaller problems. It is likely that any estimate of cloud computing reliability based solely on human perceptions or media reporting is going to at worst be gross over- or underestimates and at best to have a large variation in their estimates.

Well-known reliability engineering techniques allow the creation of highly reliable systems from unreliable components using techniques such as redundancy, these techniques come at a cost of time, money or both. It is possible that a highly reliable commercial service built using cloud computing service components — while meetings its reliability target — is actually over-engineered due to the use of conservative estimates on the reliability of the underlying components. Similarly it is possible that a system designed with redundancy is not as reliable in reality as has been assumed when too optimistic component reliability estimates are depended on. Thus while for most day-to-day problems the *exact* value of reliability metrics of cloud services are not relevant — as long as they are "good enough", in some situations a more accurate estimate of the reliability metrics would allow better resource planning and utilization of cloud computing resources.

While there are several industries which report service quality metrics (for example, power utilities in many countries are legally required to post information on reliability of power plants and electricity transmission grids), most cloud computing vendors do not publish reliability metrics. Thus there is very little concrete information a system designer, reliability analyst or a business decision-maker could use on cloud service reliability for analysis or evaluation. At the moment they would most likely rely on ad hoc estimates or use a proxy value such as availability goal specified in a service level agreement.

1.2 Problem Statement

This thesis sets out to produce statistically robust reliability estimates for the services of a major cloud computing vendor (Amazon Web Services). This work bases the analysis on public incident information published by Amazon Web Services, collected over a period of one year during years 2013 and 2014. The primary goal of this thesis is to provide useful metrics for reliability analysis and to evaluate whether AWS meets its own availability goals as set in its service level agreements, and to perform the analysis using methods and information in a way that it could be reproduced independently.

Since the work is based on external and public observations (e.g. published incident information and other reliable sources) it is subject to several limitations. For example, it is not possible to analyze root causes of the reported incidents as there is no visibility to the internal operational processes nor knowledge of the software or hardware that can contribute to the causes of observed outages. Neither it is not possible to determine prorated reliability metrics, as the proportion of affected customers is not known. Since there are only a few works looking into cloud service reliability from a practical (empirical) viewpoint, this work purposefully has set its target on a few simple metrics and purposefully avoids any deeper analysis such as looking into the causes and consequences of the underlying incidents.

1.3 Related Work

An infrastructure cloud service is a complex combination of hardware and software and thus research from many different fields is relevant when considering the reliability of such a composite system. There has been a lot of research into reliability of computers on different levels, starting from low-level analysis of failure behaviors of discrete computer hardware components such as DRAM memories and hard disks leading up to complex fault-tolerant server systems. There has also been research into the reliability of even larger computing systems comprising of hundreds of servers as well as reliability of whole data centers and its major non-computing components such as power, HVAC systems and external network cabling.

Schroeder et al [SPW09] have characterized reliability of ECC DRAM memory[1] in a large number of servers at Google and report that annually a third of the machines experienced at least one correctable memory error and 1.3% of experienced an uncorrectable memory error. Nightingale et al [NDO11] found that for consumer PCs the failure rate for CPU alone was 1 in 190 over a period of 8 months. Vishwanath et al [VN10] have analyzed server failures at a large cloud data center and came up with an annual failure rate of 8% for server machines with the largest portion caused by hard disk failures. Schroeder and Gibson [SG10] looked into failures in a high-performance computing environment. They found out that hardware could be attributed as the root cause for 53–64% of failures, with the software being the root cause for 18–22% of failures. While many of the previous authors note the lack of consensus on *absolute values* for hardware reliability — there are large variations even on relatively narrowly defined component failure rates — the overall result should be clear: computer hardware, while mostly very reliable, is subject to random failures.

Computer hardware is not the only cause of failures in cloud services. A cloud service is, by definition, accessed via a network making the network's reliability also a factor in overall cloud service reliability. Datacenter networks are designed to be reliable and fault-tolerant [BH09], yet despite this Bailis and Kingsbury [BK14] list in their overview paper several failures of data center networks — networks that have been designed to be redundant and fault-tolerant! Other layers in a cloud service such as operating system, virtualization software, management and monitoring systems, human operators etc. also have potential to cause failures — see [BA12] for a

[1] ECC stands for error-correcting memory, a type of memory that can detect and repair certain types of memory errors.

comprehensive review of potential failure points of cloud services. Thus it is clear that a cloud service is not immune to failures and may experience them on a wide scale, starting from minimal effect on a single cloud resource (virtual machine, for example) to large-scale correlated failures of a whole datacenter.

Existing research on reliability and availability of cloud services can be roughly divided taking either a *theoretical* or an *empirical* approach. Theoretically oriented research can be further subdivided into multiple categories, of which relevant for this paper are those that 1) analyze *contractual requirements* and mechanisms between a customer and a cloud vendor and those that 2) define *analytical models* of cloud services, either from the vendor's or customer's point of view, and aim to provide either estimates for service reliability or availability based on given assumptions, or try to determine what prerequisite assumptions need to hold for a given target availability to be met.

Research on contractual requirements such as the work done by Xiaoyong et al look at existing cloud vendors' service level agreements (SLA) and their penalty clauses [Xia+15]. Xiaoyong et al note that there is "variability of availability commitment and penalty in SLA offered by different cloud providers". This view is shared by Hogben and Pannetrat who note that availability as defined by different cloud SLAs could result simultaneously in both 0% and 100% availability with the same system state (failure) history [HP13]. This shows that definition of "availability" is ambiguous and its interpretation varies between different cloud vendor SLAs, making direct comparison from one vendor to another difficult. There also exists work on dynamic SLA negotiation and brokerage between vendor and customer and subsequent service quality monitoring such as work by Son and Jun and by Son et al [SJ13; SKK14]. It should be noted that no cloud vendor at this moment supports any kind of SLA negotiation nor provides their SLA in any other form other than a legal agreement (e.g. human-readable text). Thus it seems that both having a coherent and shared definition of "availability" between SLAs and capability of customers to compare different vendors' SLAs will not be likely in the near future. *Caveat emptor.*

Given that the availability targets of SLAs are going to be dictated by cloud vendors perhaps a more fruitful approach is to consider what will be availability of a cloud-based service with given component availabilities. Predicting availability of a system can be made by defining a *model* of the system which can then be analyzed either analytically or through simulations. This reliability analysis can be driven by the needs of cloud customers or by those of the cloud vendor — for example Faragardi et al [Far+13] and Beaumont et al [BEL13] look into provisioning and resource allocation in cloud services with the goal of meeting customer SLAs. Khazaei et al [Kha+12] look into how a vendor can utilize tools such as admission control to ensure that their service offering meets given reliability targets. Model-based approaches usually assume that component failure characteristics are known *a priori*, making these models most accurate for cloud vendors themselves who have access to the underlying hardware and software component reliability history. Currently consumers doing availability modeling for their systems have usually to rely on using availability targets directly from vendor SLAs (this applies also to most published papers).

In contrast, work by Fiondella et al [FGM13], Bermudez et al [Ber+13] and Naldi [Nal13] are empirical in their approach and focus on *observations* of cloud services and on the inferences that can be drawn from these observations. Fiondella et al analyzed Cloutage[2] dataset and estimated availabilities for multiple cloud vendors and different types of services. Naldi used multiple sources of outage information, including the same Cloutage data used by Fiondella et al in addition to IWGCR data, and looked into number of outages and inter-outage interval for major cloud vendors. They found out variations in service availability between different types of cloud services and different vendors. Instead of using public datasets, Bermudez et al used passive measurements by recording network traces at multiple ISPs [Ber+13]. The network traces were used to characterize traffic to and from Amazon's data centers. Although their paper does not discuss service availability or reliability, it could have potentially been used to detect at least some forms of network outages in AWS.

While there has been plenty of papers about *monitoring* cloud QoS parameters — including availability — there has not been any substantial efforts to actually collect availability measurements on public cloud infrastructures. This may be caused by the difficulty and cost of active monitoring efforts, or the fact that results from such monitoring effort would become useful and result in publishable papers only after a certain, probably quite long time. There are some companies performing cloud service monitoring and offering availability metrics for public consumption [Clo16], but these may be limited and detailed information or analyses are available for paying customers.

The importance of including *correlated failures* into availability analysis has been noted by Gonzales and Helvik [GH12] and Ford et al [For+10], and correlations between failures in separate systems have been found by Ford et al and Sahoo et al [Sah+04]. This leads to the conclusion that it is also important to study *correlated failures* in cloud services, especially given that most availability modeling approaches make the assumption of uncorrelated failures. While it is known that the underlying cloud infrastructure is just as susceptible to correlated failures as any service in a data center (see [Ama12c]), many of the potential correlations and cascade effects have not been studied.

[2] The `cloutage.org` site appears to be currently inaccessible.

1.4 Outline

The next section (section 2) provides more information on what cloud computing is, how cloud offerings are provided for customers, background on reliability and availability and an overview of how cloud services may fail and have failed in the past. Then section 3 covers potential sources of data that can be used to determine cloud service reliability and reviews differences in applicability and accuracy of these sources. section 4 describes methods used for data collection and analysis and results of the data analysis is presented in section 5. The analysis results are discussed in section 6. Finally some of the shortcomings in this work and potential targets for future work are discussed in section 7.

2 Theory

2.1 What is Cloud Computing?

It is not possible to give a single, clear definition for the the term "cloud computing". Cloud computing is not any specific set of technologies and it is not a new business model either — regardless something that is referred to as "cloud computing" has had a large effect on the IT industry within the last decade. The impact of cloud computing on existing institutions and ways of working has been described for example by Simon Wardley as *"[cloud computing is] a generic term used to describe the disruptive transformation in I.T. towards a service based economy driven by a set of economic, cultural and technological conditions"* [Sim09]. It is possible to view cloud computing as a business model (for providing networked services), as a technological change (of wide-spread adoption of networked services), as a change in business models (shift from computer products to computing products, e.g. utility computing) — and many others. The fact that people often describe cloud computing from their viewpoint may give an illusion as if cloud computing was something tangible. In this context Wardley's view of cloud computing as a transformative process captures the impact of cloud computing wide impact quite well.

Yet while the broad definition of cloud computing is useful in providing a broad framework for understanding cloud computing's impact for the purposes of this thesis its definition is too generic. A definition of cloud computing provided by Mell and Grance that focuses more on the use of cloud computing as a technology is more appropriate in this context:

> *"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released*

> *with minimal management effort or service provider interaction"* [MG09].

This thesis specifically views cloud computing as *the delivery of computing services over the Internet.* This is a stance focused on the technology where cloud computing is viewed as a convenient mechanism to acquire and use computing resources. This viewpoints ignores the "why" and "how" of cloud computing and just makes the assumption that 1) there is a need, or a decision to use cloud computing for business purposes, and 2) there are vendors providing cloud computing services. There are customers and vendors. Vendors provide cloud computing resources, and customers use them. Customers are interested in qualities such as cost, performance and *reliability* of the services they are purchasing.

Different cloud services are often categorized as "something-as-a-service" such as IaaS (infrastructure as a service), PaaS (platform …) or SaaS (software …) [ISO14]. While this categorization can be useful in understanding the conceptual placement of a particular cloud service in a broader context, the categorization is not meaningful when — as in this thesis — we look at the service as a whole and are not interested in how we would actually use the service. There are aspects of cloud services that are relevant on evaluating a service's reliability, but they depend on how the service operates instead of how it is categorized. These include for example how the geographic distribution is presented to customers and what are the types of failure modes that are visible to customers. These aspects are discussed in detail in following subsections.

2.2 Delivery Models of Cloud Services

While cloud services may be consumed anywhere as long as a working internet connection is available, the underlying computing capacity used to produce cloud services is bound to a physical location — a server is located in a rack which in turn is in a datacenter that is in a city, county, country and a continent. In the end the physical distance between a cloud resource and its consumer is the most important factor that determines the network latency and bandwidth between cloud vendors and their services and the consumer of those cloud services.

Different cloud services take different approaches to managing service's geographical locality. It is possible to provide services that attempt to hide geographic locality of the underlying cloud computing resources by replicating the service in multiple physical locations and routing the consumer to the nearest one. While this kind of *global distribution* offers location transparency, it may also impose limitations on the functionality of the service or increase costs to a level that customers may not find acceptable for their use cases.

The cloud vendor may as well have decided to offer a service only as a location-bound service, for which there may be multiple reasons. For example when providing virtual machines the actual definition of the service may directly lock it to a specific server machine. For other services the cost overhead of providing location transparency could push its cost higher than customers would accept. Operating a location-transparent service in multiple geographical areas could also put the vendor into legal risk due to conflicting legislation at different countries.

There are many reasons why a customer might prefer services that *do not* provide location transparency but are bound to a given geographical area. Statutory or contractual requirements may require customer to be physically limited to be operating from a certain region or country, end-user requirements may require a low network latency or the customer wants to ensure adequate physical separation of redundant service components to avoid correlated failures due to natural or man-made disasters.

Cloud vendors offer their services in different categories to meet potentially conflicting technical and business challenges and customer requirements. These categories are defined by the kind of location transparency they offer and service reliability guarantees they provide: 1) global services that offer location transparency globally, 2) regional services with location transparency for services within a geographical region, 3) zone-based services that are bound to a given geographic location. These categories are usually layered as shown in Figure 1 so that zones are placed in a region. This placement is visible to a customer for example by pricing, latency or bandwidth differing between inter-zone and inter-region connectivity[3].

Global services offer the highest level of location transparency. Customers of global services are usually not offered any choice on where the service is delivered from or where related data is stored. Most often consumer-oriented services offer a global service model to simplify service interface to end users. Typically the majority a vendor's global services are *supporting services* such as identity management, access control and cost tracking.

Regional services operate out of a broad geographic region chosen by the customer. Regional services differ from zone-based services in that they are provided as highly available, redundant services that will stay operational even in a situation where zone-based services may fail. The cloud vendor takes the responsibility of operating these services in a fault-tolerant and highly available configuration and is able to provide high guarantees of service availability.

[3] Different cloud vendors may use different terms to describe their geographic locality abstraction. The use of terms *region* and *zone* match those used by Amazon Web Services and Google.

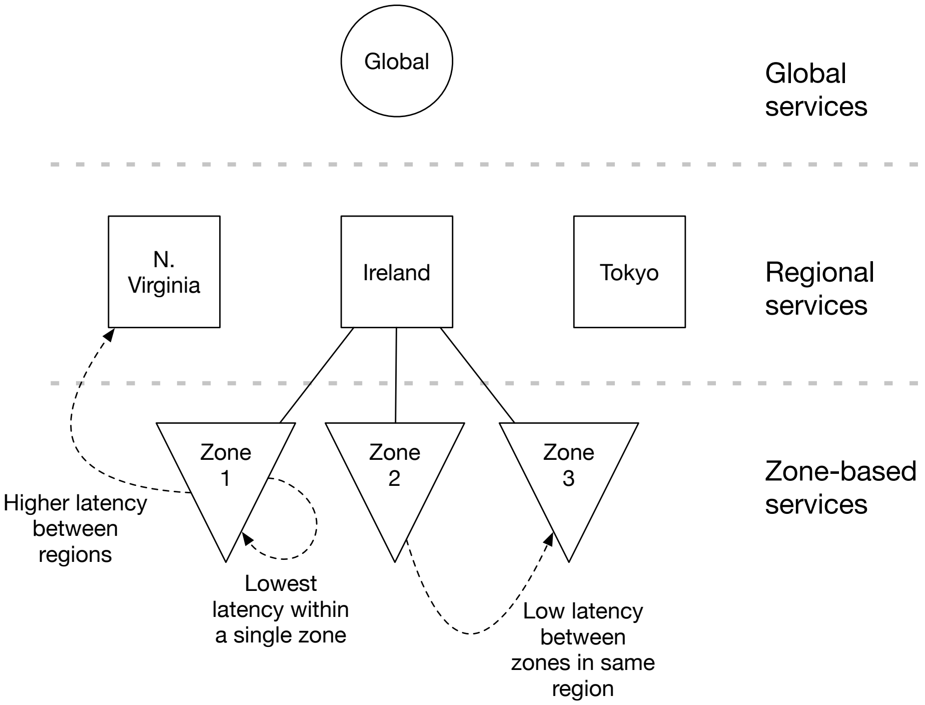

Global
Global services
N. Virginia
Ireland
Tokyo
Regional services
Zone 1
Zone 2
Zone 3
Zone-based services
Higher latency between regions
Lowest latency within a single zone
Low latency between zones in same region

Figure 1: Different common abstractions of geographical distribution of cloud services. The division between *global*, *regional* and *zone-based* services is common. In this model individual zones are part of regions. The most important distinguishing feature between regions and zones is the network latency between different configurations.

There are many different types of regional services. These may include for example messaging services, distributed load balancing, distributed databases or distributed content delivery.

The location of a regional service is either a broad region such as Germany (country) or Virginia (U.S. state) or a specific city such as Dublin (Ireland) or Mumbai (India).

Zone-bound services provide a fine-grained abstraction of their physical location for customers to choose from[4]. Cloud vendors do not usually guarantee availability of services in any specific zone.

Given the fine-grained location abstraction it is possible for a single natural or man-made disaster to reduce all cloud resources within a single zone unreachable or inoperable. If a customer wants better availability than a single zone alone can provide they have themselves use reliability engineering techniques to run the service in multiple zones in a fault-tolerant manner.

Several zones are typically grouped together within a region. Usually this zones-in-a-region grouping for geographic location is the same as used for regional services. The physical separation and network latency between grouped zones is low — in reality this is likely to map to multiple closely located but distinct data centers. Note that a single zone may be composed of multiple data centers — a "zone" is a geographic abstraction defined by cloud vendors and may map to multiple physical data centers [Ama12b].

Reliability of global and regional services is expected to be higher than for zone-based services. The reliability of zone-based services is limited by the reliability of the underlying hardware whereas global and regional services can use reliability engineering techniques (redundancy, fail-over etc.) to reduce or eliminate the effect of a single hardware failure or even a failure of a whole zone.

While failures in zone-based services may be more common, their observed reliability, oddly enough, is easier to analyze than for global and regional services. The reason is simple: there are more zones than regions — there is only a single "global" service region. Any failure in a global service thus has a potential to impact more customers than a failure in a single zone. For example Amazon Web Services at the end of the study period had 23 distinct zones in 8 regions[5]. For any percentage value chosen

[4] Sometimes it is possible to specify need for close affinity between zone-based service resources in which case the vendor either guarantees or tries to place them physically close — in the same physical server, the same rack in the datacenter, or within a single routing region within a datacenter. The specific physical location of resources is still controlled by the vendor and the customer may still only choose the zone used.

[5] Excluding GovCloud.

to quantify “some customers” the absolute number of impacted customers will be 1 in 23 for a failure of a zone-based services than for a global service. Finally as will be later discussed, the outage information that this thesis is based on does not adequately quantify the customer impact of individual outages. The source data often uses vague terms such as “some customers were affected.”

2.3 Reliability and Availability

Reliability and availability are closely related concepts and are often used interchangeably. Reliability can be defined as *the probability a system performs a required function, under specified conditions* [Smi11] whereas availability is defined as *a system’s ability to be in an operating state, ready for use* [Eus+08]. Avižienis et al highlight the difference between reliability and availability so that availability is *readiness for correct service* whereas reliability is *continuity of correct service* [Avi+04].

As a clarification on this difference consider a situation as shown in Figure 2 where a service is not responding to requests e.g. is unavailable periodically for one second after 99 seconds of being available. This is by definition a 99% available system. Next consider two different users, each using the system over a long period of time at random times with one user 5 seconds at a time and the other for 100 seconds at a time. Each of the users needs the system to be available without interruption for the time they are using the system to complete their work. While the system is available 99% of the time, its reliability is lower and is either 95% or 0% reliable. The first user is able to complete their work 95% of the time whereas the second user will never complete their work successfully.

This example should show that reliability is highly dependent on how a system is used and thus is often more difficult to compare between different use cases. A system operating according to its specifications and being highly available may still be considered unreliable by its users when requests are not served correctly *from their perspective* [BA12]. Thus it is often easier to refer only to availability and its related metrics such as mean time between failures and mean time to repair and other metrics that are useful in reliability modeling and analysis. A further consideration is that cloud services have several different independent failure modes that make the system unavailable in different ways (more on this later).

From now on this thesis will focus solely on availability. Availability can be further divided into instantaneous, interval and steady state availabilities [MM11]:

Instantaneous availability Instantaneous availability is probability that the service or business process is in the correct state and ready to perform its function at a specified time instant.

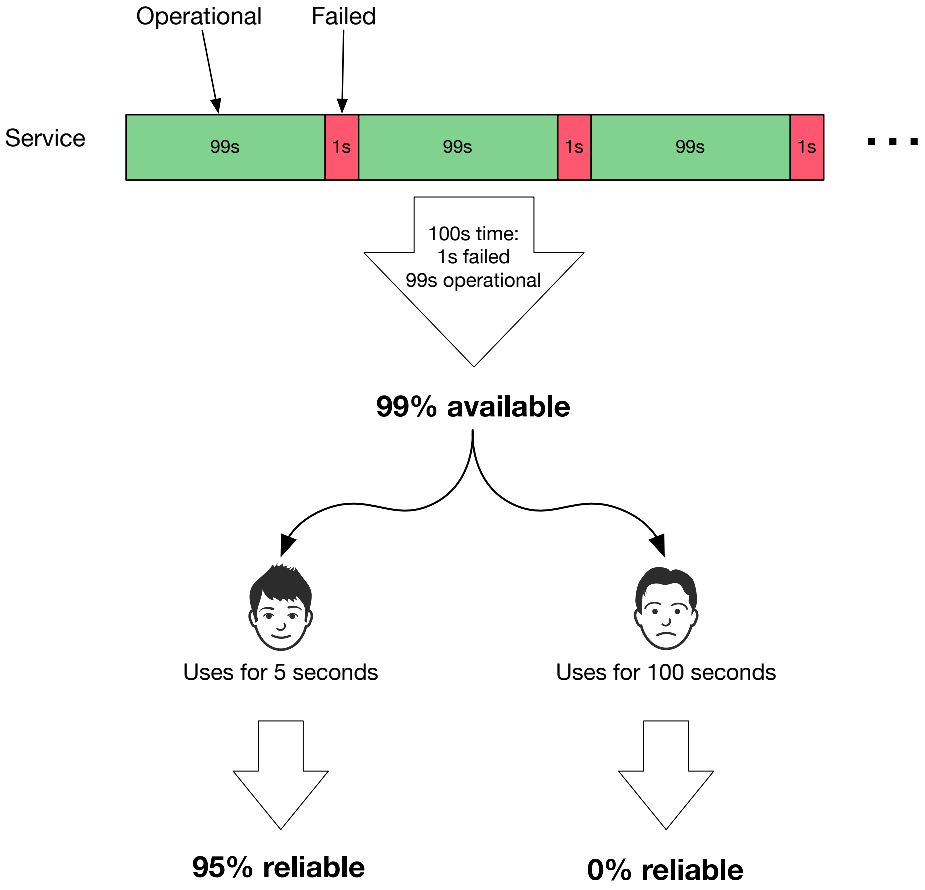


Figure 2: Availability and reliability example where a system goes through periodic cycles of being operational and in failed state. Availability, mean time to failure and mean time to repair in this system are absolute values, but a percentage reliability value is dependent on how the service is used.

Interval availability Interval availability is probability that the service or business process is operating correctly during a period of time.

Steady state availability Steady state availability is the fraction of lifetime that the system is operational.

While availability is often given as a simple metric, it must be noted that in real-world environments a single metric is a simplification. For example the instantaneous availability for many systems is a function that depends on the past history of the system — if the system is currently down then the likelihood of it being available in the near future is low! Interval availability can vary for similar time intervals from day to day for example for environmental reasons (such as dry vs. flood season). Finally a system may not even have a defined steady state availability if it is in development as many software-based services often are.

Looking at the past it is simple to produce *observed* availability metric. It is important to realize that an observed availability metric is a summary statistic that *is not* a probability value but a *proportion* value. For example, given availability observations of a system in a given time period we can unambiguously determine whether the system was operational at a given time or not — this is a fact. It is not however meaningful to discuss about the *probability* of the observed system being available at some time in the past — probability there is not, facts alone. The observed availability can of course be assumed to be representative of the system's long-term or steady state availability. If this assumption is valid then the observed availability can be extrapolated as an probability for the system being available in the future.

Giving a definition for *observed availability* is straightforward:

$$A_O = \frac{O}{T} = \frac{T - I}{T} = 1 - \frac{I}{T} \tag{1}$$

Here A_O is the proportion of time the system was in operational state O compared to total service time T ($I = T - O$ is the time the system was in inoperable state). In simpler terms the observed availability A_O is the proportion of time the system was operational over the observation period and is between values 0% and 100%.

While the equation is simple, in practice values of I and T are not always immediately obvious and can lead to subtle interpretation problems (consider the difference between reliability and availability as noted earlier). Consider a situation shown in Figure 3 with one global service, one regional service operating in two regions and one zone-based service operating in two different zones in the same region. Calculating A for any service at its smallest boundary (global, per region for regional service, per zone per zone-based service) is simple. The global service alone was available for $A_g = 13/15$ and the individual zones for $A_{z_1} = 12/15$ and $A_{z_2} = 15/15$. How one should then answer questions "what was the overall service availability?" For overall service availability we could use (for simplicity this example uses discrete observation and time units):

$$A_1 = \frac{\sum_{x \in all} O_x}{\sum_{x \in all} T_x} \tag{2}$$

which is 65/75 (here $x \in all$ iterates over the set of all observed atomic services). Yet it could be possible to interpret the question equally well as

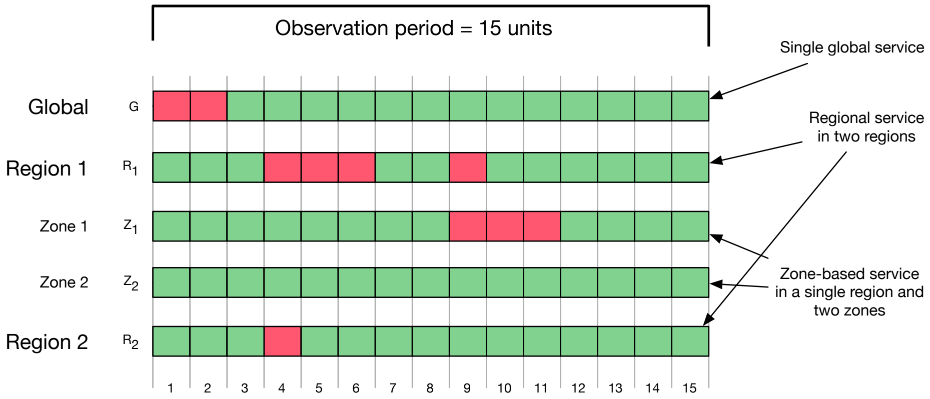

Observation period = 15 units
Single global service
Regional service in two regions
Zone-based service in a single region and two zones
Global
G
Region 1
R1
Zone 1
Z1
Zone 2
Z2
Region 2
R2
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Figure 3: Example of three services, one global, one regional and one zone-based available in two zones. The total observation period is 15 time units. Red signifies an outage and green that the service (in the particular region or zone) was normally available.

$$A_2 = \frac{\sum_t |all| \prod_{x \in all} O_x^i(t)}{\sum_{x \in all} T_x} \tag{3}$$

where $O_x^i(t) = 1$ if the service x was operational at time t and 0 otherwise and $|all|$ is the number of elements (cardinality) of set *all*. This specifically is the proportion of time that *all* services under consideration were simultaneously available. In this case $A = 35/75$, significantly worse availability than with Equation 2 which is an unweighted average of the availability of all services, regions and zones.

This example tries to illustrate how easy it is to start considering a system's *reliability* with an implicitly assumed use case. In the previous example only A_1 was a "true" availability value while A_2 assumed an use case where all services under observation were required to be operational at the same time and in reality is a reliability metric.

2.4 Cloud Failure Modes

At some level cloud computing services are running on unreliable hardware. While individual cloud resources may stochastically fail in ways that are externally visible (virtual servers crash or freeze, disks produce read and write errors, networks drop packets etc.), these are normally transient, recoverable and affect only very limited number of customers. While these need to be taken into account there is no reason to assume these failure modes are any different for cloud resources than for similar resources in a managed (non-cloud) data center. Also as later discussed in subsubsection 3.1.3 cloud service vendors do not make small-scale failure information available (e.g. it does not cross reporting threshold). Externally detecting small-scale failures below the reporting threshold would thus require active use of the resource by the monitoring entity.

Cloud vendors also may use selective resource placement to decrease the likelihood of a single physical failure causing multiple simultaneous failures for the same customer. As an example at least in 2009 AWS avoided placing multiple virtual machines from a single customer into the same physical server [Ris+09], thus failure of a physical server would not by default cause correlated failures for a single customer. While this or similar techniques does not increase actual service reliability, it can spread a single failure over multiple customers in a way that turns these correlated failures into apparently (at least from a single customer's point of view) to an uncorrelated event.

Failure that occur on larger scale e.g. that can cause failures on many resources or for multiple customers can be roughly categorized into *core resource* failures, *network connectivity* failures and *control plane* failures. For example a failure of the core resource would mean that it is not operational — a virtual server is down or an attached disk drive is returning read errors. Another failure model is where the core resource is operational, but it is unreachable due to a failure of (internal or external) network. In a network connectivity failure a virtual server could be fully operational but not reachable by end-users. Finally both the actual core cloud resource may be both operational and reachable over the network but the mechanism used to configure, provision and de-provision the cloud resource may not be operating correctly. In such situation for example a cluster of virtual servers working as a web service front-end would be successfully servicing end-user requests, but the customer would not be able to increase the number of front-end servers if the number of customers increased thus potentially leading to a degraded end-user service overall. Note that different usage patterns may lead to different perception of the system reliability when considering different failure modes — a system running fluid mechanics simulation for several days might not even notice temporary losses of network connectivity or control plane failures during its long computation phase!

These correlated failures affecting either core resources, network connectivity or control plane can have a wide variety of root causes. Large-scale cloud failures are known to have been caused for example by power system failures [Ama12a], network configuration errors [Mik12], control system software bugs [Bil12] and operational failures [Jas14].

3 Data

3.1 Potential Data Sources

To determine availability of a cloud service, the first step is to collect data for analysis. Naturally the cloud vendors themselves are best positioned to collect this data as it is reasonable to assume they have comprehensive monitoring data of their own systems. They do not, however, make this data available for the general public or external researchers. While some services publish their own summary statistics these are of limited use. The choices for collecting service status information are more limited to customers, for example. The generally available sources of availability information for a cloud service customer are:

- Monitoring of a cloud service using active or passive techniques,
- Media and customer reports such as newspapers, trade journals, blogs, twitter messages etc., and
- Vendor-published information such as post-mortem analyses, service status updates on public web sites such as service status dashboards and on other media such as social media feeds.

Each of these are discussed in detail below discussing the benefits and downsides of different methods.

3.1.1 Monitoring

Monitoring a cloud service for its whether a service is available or not allows the party doing the monitoring to set their own definition for "availability" and recording granularity. This allows the monitoring party to tailor the measuring system to their own needs and business goals. While this makes interpretation of the data more straightforward in that context, it may have a downside of making it more difficult to compare results with other sources.

Most monitoring systems consist of active probes or agents deployed in a target system. Typically their primary use is not availability monitoring. A monitoring system that is already deployed to monitor the state of a service deployed using cloud computing can be used also to collect information about the availability of the underlying cloud resources. While this type of availability monitoring is active in nature, it is not exclusively targeted for availability monitoring — in a way availability information collection can be cost-effectively piggy-backed on a monitoring system deployed primarily for other reasons.

While it is possible to deploy a system to explicitly monitor cloud service availability, this could become prohibitively expensive. The reason for high costs of monitoring is simple: you have to use cloud computing services to monitor them. When a monitoring system is deployed to monitor a system using cloud resources for a business purpose this cost is implicitly included in the operational costs of the system. If no such piggy-backing is possible then all costs are directly attributed to the monitoring effort. These costs are a function of the monitoring coverage wanted but could even for relatively small monitoring effort raise to thousands of dollars[6]. The underlying problem is a sampling problem — there is a finite number of failures of which monitoring needs to have a good enough number of samples to get statistically meaningful results.

Note that the approach taken by Bermudez et al to measure performance metrics of AWS's data centers [Ber+13] can be considered a monitoring approach that piggy-backed as implicit data on top of network traces that were collected by a system that was *not* specifically designed for cloud service quality monitoring. While possible for researchers this type of network trace data is not readily available to other parties.

Using monitoring to evaluate a cloud service availability also has an inherent chicken-and-egg problem. If there is need for cloud availability data *before* deploying a system using cloud resources, does it actually make any sense to deploy a monitoring system for the sole purpose of getting availability information for a system, which, when deployed, is going to have a monitoring system that will record information that is readily usable for determining the *actual* availability of the whole system and the cloud resources it uses?

Companies that make use of cloud computing do undoubtedly already have a good record of monitoring data and detailed availability statistics — or at least the possibility to have them calculated from existing monitoring data. This information — again — is generally not available. Thus while monitoring is useful for gathering information suitable for availability calculations it is not an approach that can be used to gather availability information beforehand or with a limited budget.

3.1.2 Media and Customer Reports

Many cloud service outages are reported in media by professional publishers as well as directly by customers in other venues such as blogs or on social media services such as Twitter. There are also some sites that collect databases on cloud outages, primarily relying on either direct customer reports or reports from interested third parties. While reports in media may be triggered either indirectly by highly visible customers reports or from a

[6] While monitoring a cloud service within a narrowly defined scope can be done on a low budget, accomplishing monitoring coverage for all cloud services of a vendor is more expensive. For example covering only virtual machine failures and network failures for only the cheapest instance type in AWS while covering all regions and availability zones would at November 2015 prices cost about $3000 per year.

Variable	**Effect**
Outage size	Outages affecting more customers are more likely to be reported than (even serious) outages affecting only a few customers
Outage severity	Outages with high severity e.g. having effects that are visible to general public are more likely to be reported
Vendor size	Any news on a large vendor is more interesting to media outlets
Number of customers	Large number of customers increases the likelihood that an outage will affect a vocal customer
Beliefs	People with beliefs that cloud services are unreliable are more likely to notice and thus redistribute content on cloud service failures
Fault-tolerance	Customers with fault-tolerant and redundant systems are less likely to make notice of a failure
Novelty	New and novel technology does receive more media coverage (for both good and bad news) than older, less exciting solutions

Table 1: Potential variables that affect media and customer reporting.

journalist following vendor's own publication channels, customer reports are primarily triggered by customers experiencing outages directly.

Media and customer primarily report only outages and normal service operation is not normally noted. This reporting itself is subject to many biases that affect whether the outage is reported at all, in what detail and how widespread the reporting is. Some of these biases are described in Table 1. While theoretically all of the data published on the Internet "is out there", in practice finding this information itself can be very difficult. Most media publications can be searched using media-specific search engines and this is likely to produce a good coverage of relevant articles, but searching the general internet is much more subject to many biases and selection effects caused by the search engine and exact search terms used. Even if the used search engine does not introduce biases the sheer volume of search results quickly becomes an obstacle.

The detail in individual media and customer reports may not have the necessary information for reliably characterizing the outage, or other relevant feature such as when the outage occurred, duration of the outage and what services were affected by the outage. This means that multiple reports about the same outage may be needed to get collect all the necessary information for analysis. These biases in combination have potentially the effect of increasing the relative number of reports on highly visible outages compared to smaller outages.

While media and reports by customers and individuals are potential source of information for availability analysis, they are affected by human, business and personal biases. While specifically customer reports offer a way to evaluate actual impact of outages, at least the author of this work considers their practical value for understanding the broad scope to be less useful than for analyzing specific incidents.

3.1.3 Information Published by the Vendor

Cloud service vendors often publish information about the operational state of their services. The published information varies widely and includes examples such as current system status, outage reports and outage post-mortem analyses. At the moment major cloud vendors do not publish current or historical availability statistics for their services.

The most detailed reporting on incidents is found in post-mortem analyses. These are usually published only for major incidents so the number of post-mortem analyses published by a vendor is low compared to the number of overall outages and other types of status updates. In contrast to normal outage reporting the post-mortem reports include more details on the incident such as accurate outage start and end times, list of affected services and sometimes an indication of the number or portion of customers affected by the incident. A major part of a typical post-mortem analysis is the root cause analysis of the outage followed by a generic boilerplate text where the vendor assures that adequate steps are taken to prevent the same kind of major incident from occurring again [Bil12; Ama12c; Ama12a]. Post-mortem reports provide good insight into how major failures occur and offer a chance to cross-check the accuracy of information collected from other channels. The frequency of post-mortem incidents is low with a typical rate of a few per year by a single vendor. While post-mortem reports usually offer quite detailed information about an outage, given the small relative number of them (compared to smaller but not inconsequential incidents) this does limit their use as they might not be representative of typical outages.

Most vendors offer a web-based view of their cloud service's operational status. These status dashboards allow an at-a-glance overview of vendor's services showing whether any service is non-operational, degraded or operating normally. Their primary purpose is to let customers to check on the service status. The main interface for the status dashboard is the web browser and while dashboards often show historical data the visible time window is often limited to a few weeks with earlier data not being accessible. For examples of status dashboards see AWS's Service Health Dashboard [Ama13c], Azure's Status [Mic15] and Google's Cloud Status [Goo15].

In their for-human consumption form these status dashboards offer a time-constrained view to current and past system status. Thus while it is possible to perform availability analysis based on the information from status dashboards it would be limited by the lack of historical data older than a few weeks.

All status dashboards mentioned above also provide RSS feeds[7] — a format more suitable as input to alerting systems, for example. RSS feeds provide an easy way to automatically retrieve outage information. They are also limited in size and thus also limit the number of outage report messages that can be published. These limits are set by the vendor with for example Google apparently including outages for a full month in their feed and AWS limiting the number of entries in a single feed to a fixed number of outages. Since no other public, well-defined and stable access methods for cloud outage data is available, this work uses RSS feeds to collect specifically AWS outage information. (The next section covers the behavior of AWS Service Health Dashboard and its RSS feed contents in detail.)

The outage information published by vendors themselves is open to several potential biases. Some of these biases are discussed in Table 2. Most of these biases stem from vendor's internal processes and reporting policies of the vendor. Even a cursory look at any vendor's status dashboard confirms that only failures crossing some (unknown) threshold in severity or size are reported, especially if correlated with customer reports in social media about random failures (e.g. showing that there are incidents which are not reported).

3.2 Outage Reporting by AWS

The AWS Service Health Dashboard and the corresponding RSS feeds contain outage reports in the format of messages. Each message contains information relating to a single service or to a single service and region[8]. Each message also contains a unique URL, publication time, subject line and message text body.

There are limits on how many messages are shown on both the dashboard page and RSS feeds. The dashboard web page is limited to 30 days of history

[7] RSS stands for Rich Site Summary. It is a structured data format commonly used to publish frequently updated information. While mostly used to publish blog and other social media feeds it has also found use in providing a way to distribute status incident information.

[8] The way AWS structures outage reporting is not entirely orthogonal or consistent. Each outage report is associated with a particular service, but not all services have their own reporting RSS feed. For example EBS outages are reported as part of other services. Auto Scaling outages were originally reported as part of other outages but from late 2014 onwards Auto Scaling outages were published in their own RSS feed.

Variable	Effect
Length, size and severity	It is likely that there are thresholds in what outages get reported such as outage length, size, severity etc.
Process adherence	There may be regional or other differences in how well reporting policies are followed
Process changes	Changes in reporting policies can introduce unknown changes in reporting rates
Software changes	Changes in monitoring and alerting systems can change reporting rates
Vendor	Reporting policies and processes differ from one vendor to another

Table 2: Potential variables that can affect decisions on whether an outage is reported by a cloud service vendor.

and each separate RSS feed[9] is limited to a maximum of 20 most recent messages. While both show the same information, messages related to the same outage are grouped together in the web page version whereas in the RSS feed they are disconnected as there is no correlation identifier to link different messages in the RSS feed together.

Messages can be grouped into a few generic categories (sample messages can be found in Table 3):

- **Initial report messages** give an indication that there is or may be a problem that is being investigated.
- **Resolution messages** report on a solved incident and usually mark the end of an outage message chain. It usually includes the start and end times of the outage and may include some additional information.
- **Ongoing investigation messages** are often published for longer outages. They do not usually contain new information on the outage.
- Sometimes only a single message is published that reports the end of an incident and potentially other information on the outage.
- Other of types messages such as information about a scheduled outage, or reports of investigations that showed no problem are much more rare.

[9] There is one RSS feed for each global service and one per service per region for regional and zone-based services, so a RSS feed may contain messages older than the 30 days they would be visible at most on the web page.

Type	**Subject** and **Body text**
Initial report	**Informational message: Increased API error rates** We are investigating increased API error rates and latencies for the EC2 APIs in the EU-WEST-1 Region.
Resolved	**[RESOLVED] Network connectivity** Between 12:07 PM PDT and 1:15 PM PDT we experienced impaired Internet connectivity affecting a small number of instances in a single Availability Zone in the US-EAST-1 region. Additionally, between 12:07 PM PDT and 1:20 PM PDT we experienced increased error rates for the DescribeReservedInstances and DescribeReservedInstancesOfferings APIs in the US-EAST-1 region. Both issues have been resolved and the service is operating normally.
Ongoing	**Informational message: Connectivity issues** We are currently investigating connectivity issues to a small number of RDS database instances in a single Availability Zone in the AP-NORTHEAST-1 Region.
Single message	**Informational message: Increased API error rates** Between 3:05 PM PDT and 3:45 PM PDT Elastic MapReduce customers experienced increased API error rates in the EU-WEST-1 Region. Some customers experienced delays when starting or terminating their job flows. The service is now operating normally.
Other	**Service is operating normally: [Resolved] Increased API Error Rates** The RDS service was, and is operating normally. Our investigation has shown that the errors detected were false positives that did not affect the operation of the service.

Table 3: Samples of the common types of messages in AWS outage reports.

A feature shared by all AWS outage reports that complicates automated data processing is that the message text itself is free form text, clearly written by humans and meant primarily for humans to read and not meant to be automatically processed by computers. While most outage reports follow a few common patterns these patterns are not rigidly followed. Messages also contain errors such as typographic errors (“EU-WEST-2”) or logical errors (text reports times in PST when daylight saving time is in effect and PDT would have been correct and vice versa). Times provide other complications as they are written in many different formats such as “14:40 PST”, “2:37 PM PST”, “12/17 10:32PM” and “2:10 A.M. PST”, often omitting information such as date or AM/PM distinction that needs to be inferred from context.

While not affecting actual outage analysis, it is also clear that sometimes messages that have already been published are retroactively edited. This is confirmed by noticing that when messages are periodically collected sometimes an original and a modified version of it are collected and can be compared directly. Another way to confirm the retroactive edits is that some messages refer to events that occur after the apparent message publication time. The third potential retroactive edit is the addition of “[RESOLVED]” text to subject lines to all messages of an outage, although this occurs so regularly that it is likely to be automatically added by the publishing system instead of being manually changed.

For analyzing the impact of outages there are two important metrics that are missing from outage reports: number of affected customers and the severity of the outage. In cases where the report refers to the impact of the outage it is most often described qualitatively using vague terms such as “some instances”, “a small number of customers” or “increased error rates.” Generally there is very little concrete information on the absolute or relative number of affected customers on any outage. Sometimes more quantitative numbers are found in outage post-mortem reports or in reports written by affected customers (for examples, see [Ama12c] and [Coc12]).

In summary, while AWS outage information can be retrieved automatically via RSS feeds, the data itself is in unstructured free text format and requires substantial manual and automated processing before being usable for analysis.

4 Methods

4.1 Overview

The processing path from raw data to final analysis consists of three primary steps: 1) data collection, 2) message processing (collation, categorization and outage clustering), and 3) analysis. These data processing steps are described more in detail below followed by description of the statistical methods used and how other needed information was collected and processed.

4.2 Data Collection

The data collection process is very simple: a program runs periodically and connects to the AWS Service Health Dashboard page, retrieves the list of available RSS feeds, downloads and stores all of the feeds on disk. The data collection process is run redundantly on multiple computers several times a day. The collection process was designed to be robust and ensure safe storage of the collected data as it had to remain working correctly for the data analysis period (from mid-2013 to mid-2014) with limited supervision.

4.3 Message Processing

Message processing is broken into multiple discrete steps. These steps are 1) combining the separate RSS feeds, 2) extracting and parsing outer message structure, 3) parsing the textual message and extracting time intervals from them, 4) clustering parsed messages into events and separating events with outages from non-outage events, and 5) writing out results in a format usable for analysis.

While first two steps and the last step are straightforward data processing, in contrast third and fourth steps are more complex. As noted earlier in subsection 3.2 the status dashboard information is written in human-readable english and contains many features which makes it difficult to parse mechanically. The third step — the parsing step does two important tasks: it identifies distinguishing features from the message (such as mentions about regions and services, or language that implies a network failure, for example), and parses any valid timestamps and time intervals mentioned in the text.

Identifying relevant textual features and extracting time interval uses a custom regular language parser working on a tokenized message text augmented by a context-aware time value parser. For example the time interval regular language expression used in the parser can handle over a hundred differently worded expressions of the form "from <time> to <time>." Time values are parsed by context-aware regular expression parser — context awareness is required since individual time values may be lacking AM/PM indicators, time zone information etc. which must be inferred from other surrounding time values and message publication time.

Extracted features and time intervals are used to cluster messages into events. An event is thus a group of messages that the clustering algorithms considers to be related to each other. The clustering algorithm is an ad hoc connectivity-based algorithm operating in eight metric dimensions. These metric dimensions are calculated from message features and time intervals[10].

[10] These map roughly to region similarity, service similarity, point time equality, closeness

The parameters for the different metric functions have been determined using genetic algorithm using a training data with the fitness based on comparing resulting clusters to a manually determined clustering target.

While the above steps may appear straightforward, it must be noted that there are also semi-manual "fixes" on the data stream. These include non-semantic changes in original message text to work around limitations in message parsing as well as fixing problematic semantic and typographic errors, manipulating the feature list (for example, adding data to scheduled maintenance messages that helps identifying non-outage events during analysis) and adding or removing time intervals for messages with multiple distinct time intervals which the parser does not handle correctly.

4.4 Analysis

Analysis is performed using the statistical computing program R. The analysis includes reading output files from the message processing phase, filtering out data outside the selected analysis period, and annotating different projections of the data with AWS infrastrutucture information[11].

4.5 Statistical Methods

The main statistical methods used for descriptive statistics and error analysis are:

- Sample mean and sample error are usable for straightforward sample statistics such as the average outage length as shown in Table 10. It is important to note that these statistic are *sample statistics* and thus describe statistics of the *collected sample.*

- The bootstrap method is used for generating statistics for different metrics with unknown distributions [Hal88; DH99]. The bootstrap method is a Monte Carlo method based on re-sampling of known (sample) distribution. This method makes it possible to generate meaningful statistics without the need to make assumptions of the actual model distribution. The bootstrap method also allows to generate error estimates for the statistics as well as a range where 95% of the distribution based on resampling lies.

of publication time, exact time matches, generic feature similarity, Levenshtein distance and temporal overlap. Although anecdotal, it is interesting to note the genetic algorithm placed a lot of weight on temporal closeness of messages but a *negative* weight on textual similarity! This may be due to messages often using similar phrases, making textual similarity a bad metric for clustering — although why it did not receive a *zero* weight instead of a negative one is puzzling.

[11] The service time over a period of time for example depends on the number of regions, availability zones and the availability of services for general public and commercial use.

The bootstrap method thus generates *distribution statistics* that are based on re-sampling of the *collected sample*. They are thus valid only with the assumption that the collected sample is representative.

- Some incidents do not have explicitly written start and end times. During bootstrap analysis the length of these incidents is adjusted by sampling with replacement from the sample of all *known* start and end time difference values (Δt_s and Δt_e values in Table 5).

This work makes a conscious effort to present all values with error estimates to allow anyone using these values for example in availability modeling to have an understanding of the "goodness" of the values.

4.6 Infrastructure Data

Calculating availability as a fraction of the total availability to total service time requires knowledge of the service time. Cloud services are not static and neither is AWS's infrastructure either. To accurately calculate the total services time information about changes in the service offering and number of availability zones in regions is required. This data was collected by reading AWS's news announcement archives, AWS support forums and reviewing old versions of AWS's service web pages using The Wayback Machine[12]. The data allows analysis to accurately calculate how many services, in what regions, and how many availability zones were available for any time interval within the analysis period.

5 Analysis

5.1 Overview

The raw data is analyzed in multiple stages. First, individual messages are analysed without considering their textual content. This is followed by analysis of events and incidents, and finally actual outages and availability.

The data for analysis was collected for a year from June 5th 2013 to June 4th 2014, e.g. 365 days — a full year. Within this period data was actually collected at least once on 360 distinct days[13]. The AWS dashboard contents were successfully retrieved 1313 times resulting in a total data set of 700 megabytes of RSS feeds in XML format.

[12] `https://archive.org/web/`

[13] Total of 5 days data over the study period was not collected due to both collecting computers being unavailable at the time to collect data. These days are non-sequential and analysis of the collected data shows that there are no gaps in the collected messages during these days.

5.2 Messages

There are 662 unique messages incidents for 25 different services. The first message was published on 9 July 2013 and the last message on 26 June 2014, thus the time period covered by incident messages was slightly shorter than the full study interval. On average 1.8 messages were published per day, although *at least one message* was published only on 121 days. The number of messages has a high variance so on days *with at least one message* the average is significantly higher at 5.5 messages per day.

There are differences in the number of messages attributed to different services and regions as can be seen in Figure 4. Over half of the messages are for EC2, ELB and RDS services or for the `us-east-1` region. A likely explanation for the large portion of messages for these services and the US East region is simply that they simply account for a significant portion of the services used during the study period. This is however a hypothesis only since no reliable information on the relative weight of these services or the US East region compared to other services and regions is not available. Thus it is not possible to rule out the possibility these services or the US East region might not have other reasons to have more published messages (such as lower reliability or different reporting practices.)[14]

Most messages are short with all messages containing between 37 and 2670 characters with the median length of 176 characters and 95% percent of messages being 423 characters or shorter. While the majority of messages are short, there is a significant tail of longer messages as shown in Figure 5. Longer messages are associated primarily with either complex or long incidents and are either retroactively edited messages (with multiple concatenated updates) or otherwise longer explanations of an incident. In contrast the very shortest messages usually follow the pattern of "we are investigating *«a visible problem»* in the *«region affected»* region."

Many services had no incident messages published during the study period. Out of the 37 services listed in Table 9 almost a third of the service (12) had no published incident messages. None of the services with no incident messages were zone-based services, and some of them were relatively new services possibly with also (comparatively) low usage. While it is possible that there are differences between the set of services with no incident messages and other services such as differences in software component quality, differences in operating procedures between services, differences in reporting thresholds or more resilient service architectures it is also possible that through pure chance alone these services did not encounter any publicly visible incidents during the study period.

The messages themselves contain several anomalies. Some of the messages are retroactively edited which can be confirmed by noticing that some

[14] The `us-east-1` region is the very first AWS region, and it is plausible that newer regions have different designs.

messages have more than one version of their contents, showing up when a message has been collected multiple times with edits occurring at some time between the collection times.[15]. These retroactive edits occur primarily for longer incidents, but even then only irregularly. It seems there are two different approaches at AWS on how to report long-lasting incidents: either publish new messages with updates, or keep updating a single message with new information. It is possible this might show differences in preferences between regions or operations teams.

Another anomaly is a skew in message publication times. The seconds value of each message's publication time is zero seconds (e.g. timestamp value is `HH:MM:00`) in over 30% of all messages when by pure chance zero second values should be only a few percent of the total. All of other second values (1…59) have more uniform distribution as do hour and minute values. A possible explanation is that when a human enters the time value manually they do not usually enter a second value, and without an explicit second value the system might default it to zero. As such this anomaly does not affect incident or outage analysis but offers an interesting view into the normally hidden mechanics of incident response within Amazon Web Services.

5.3 Incidents

Running the clustering algorithm on the messages yields 142 clusters of messages e.g. events. The number of messages per event ranges from one message per event up to 50 messages, although most events have only a few events with the the median being 3 messages per event (see Figure 7.) This number of events includes 3 events that are not outages with them being either reports of issues that turned out to not be incidents or that were reports of a scheduled maintenance. Omitting these non-outage events leaves 139 outage events e.g. incidents.

Note that a single incident may affect multiple services and regions. Most incidents are restricted to affecting only a single global service, or a single regional service within a single region (see Table 6). Less than one quarter of all incidents affect more than one service or more than one region. Very few incidents affected more than one region or more than a single global service, with only less than 5% of incidents affecting more than one regional service. This means that most AWS incidents for region-based services are geographically constrained and do not cross regional boundaries.

A single incident may affect different services and regions and each of these may be affected for a different period. Thus a single incident consists of intervals of non-operational time e.g. outages. It is possible possible

[15] The incident subject line is changed for resolved messages to include the text `[RESOLVED]` but this change is consistent and regular and is likely to be caused by the incident report publishing system itself.

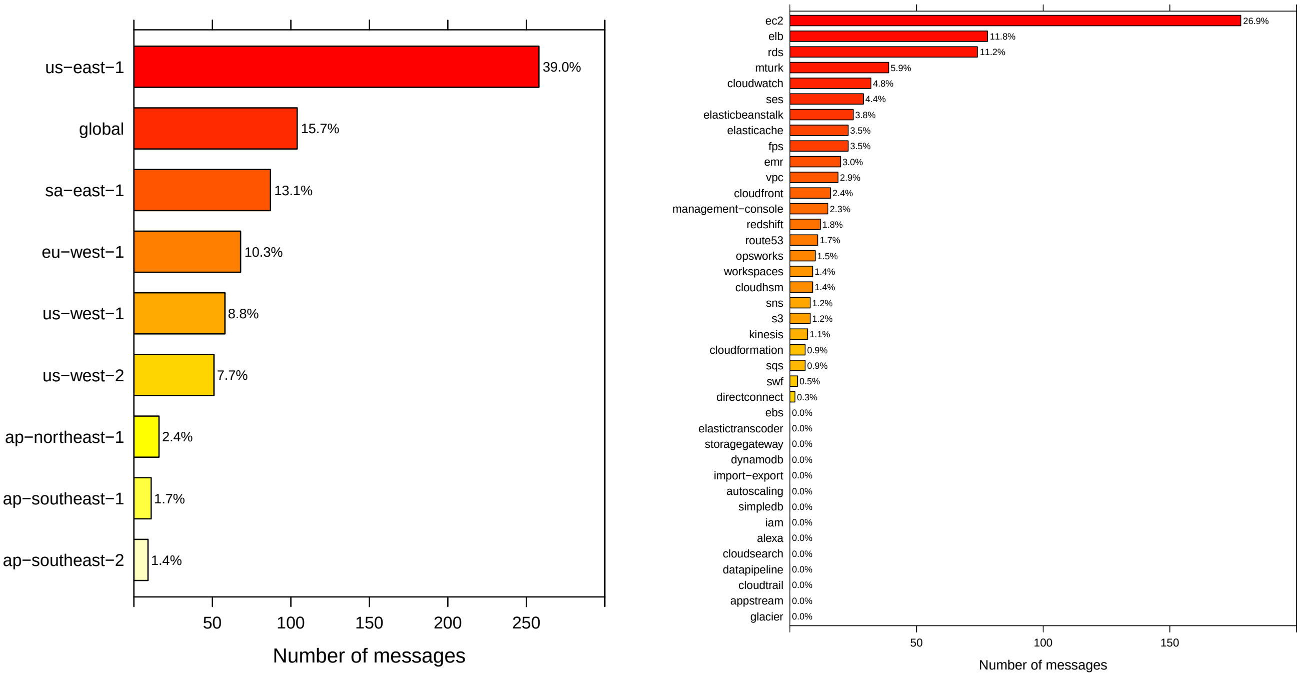


Figure 4: Number of messages and their portion of all messages published in each region (left) and by service (right). The bottom part of the bar shows the number of filtered messages compared to the total number of messages per region. (Detailed breakdown of messages by region and service can be found in Appendix C.)

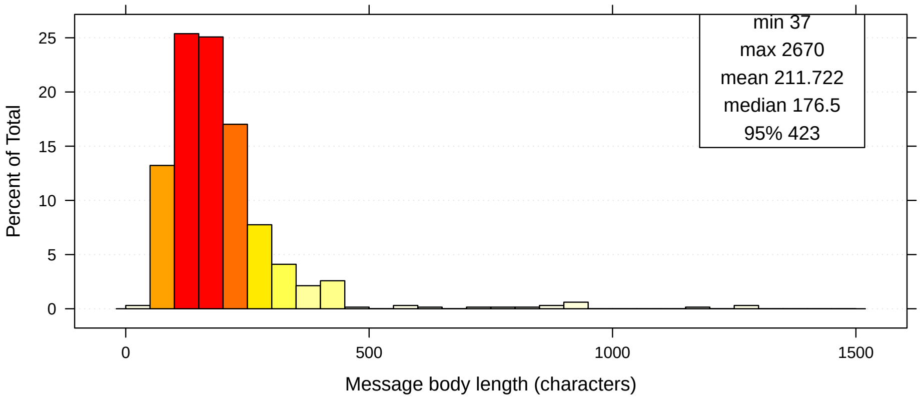


Figure 5: Message length distribution. This diagram is truncated at 1500 characters.

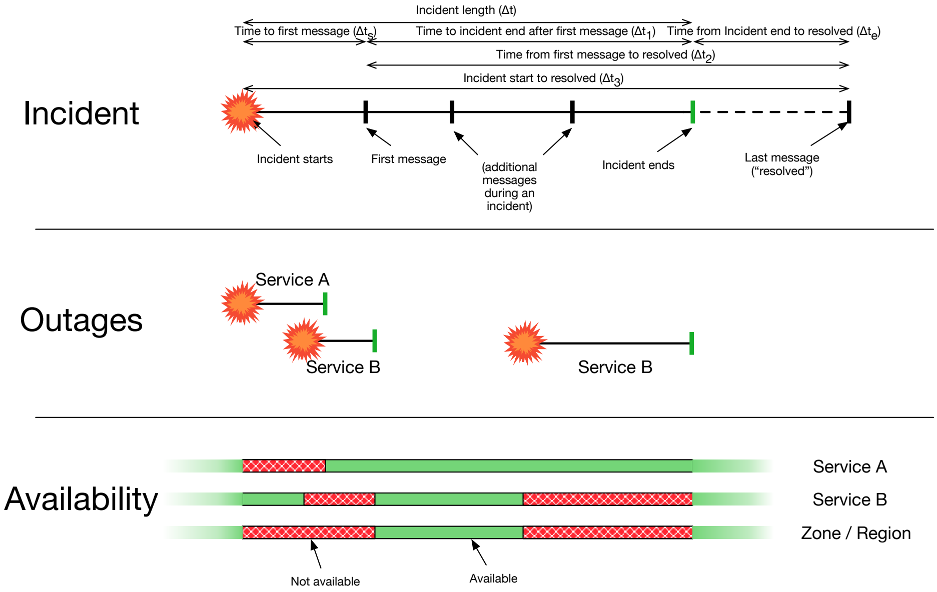

Incident length (Δt)
Time to first message (Δts)
Time to incident end after first message (Δt1)
Time from Incident end to resolved (Δte)
Time from first message to resolved (Δt2)
Incident start to resolved (Δt3)
Incident
Incident starts
First message
(additional messages during an incident)
Incident ends
Last message ("resolved")
Outages
Service A
Service B
Service B
Availability
Service A
Service B
Zone / Region
Not available
Available

Figure 6: Different time intervals associated with an incident. The incident is usually reported in multiple messages. A single incident can consist of multiple outages affecting different services over multiple time spans and potentially in multiple regions. Finally availability depends on how it is defined.

for an incident to last several hours but to have only a few minutes of actual outages during that time. It is important to keep this difference in minds since **incident length and service outage time are not the same.** Incident length, among other incident time metrics, may be useful for incident response planning. It can not be used for evaluating service availability, though.

There are several relevant time metrics for an incident as shown in Figure 6. These correspond to message publication times and the outage intervals during the incident:

Incident length Δt can be determined accurately from incidents that contain messages explicitly specifying incident start and end times. For incidents without such messages the incident length needs to be estimated based on the interval between first and last message instead. The majority of incidents (91%) have explicitly specified incident start and end times.

Time to first message Δt_s is the delay between an actual incident start, and the time when the first message about the incident is published by AWS.

		Interval value		
		$> \mathbf{0}$	$= \mathbf{0}$	$< \mathbf{0}$
Incident start to first message	Δt_s	126	①	0
Incident end to last message	Δt_e	126	0	①
First message to incident end	Δt_1	93	①	㉝
First message to last message	Δt_2	111	⑯	—

Table 4: Number of incidents with accurately known start and end times by their different delta values with respect to incident start, end, first published message and last published message. Circled values are discussed in detail in the main text.

Time from incident end to last message Δt_e is the time taken from actual incident end to when a message reporting it as resolved is published.

Time to incident end after first message Δt_1 is the time an incident can be expected to go on from the first time it is publicly acknowledged by AWS.

Time from first message to resolved Δt_2 is conversely the time from first report to the incident being publicly stated as resolved.

Note that the last four values can be determined only for incidents that contain explicitly reported incident start and end times. For incidents without accurate start and end times only the time between first and last incident message can be determined, and it needs to be statistically adjusted to take account the distribution of Δt_s and Δt_e.

A relatively large number of incidents of 27% do not conform to a “canonical” incident with all of Δt_s, Δt_e, Δt_1 and Δt_2 being positive. These anomalies are circled in Table 4. Some incidents are reported when the actual problem has already gone away ($\Delta t_1 < 0$) and some incidents have only a single message confirming a resolved incident ($\Delta t_2 = 0$). A closer inspection of “non-canonical” incidents shows that

- The single incident with $\Delta t_s = 0$ has multiple messages, with the first one following common “we are investigating” template and a later message confirming the incident start time as the time of the first message. It seems unlikely that the true incident start time was the time of the first message suggesting that the time was selected because true incident start time was not known or otherwise due to a human error.

- The single incident with $\Delta t_e < 0$ has had its first message retroactively edited to include information about the ongoing event. Since the publication timestamp of the edited message is not updated this causes an anomalous negative Δt_e value. (Taken at face value this would mean the end of the incident has been predicted and reported in advance.)
- The single incident with $\Delta t_1 = 0$ has the incident end time either erroneously marked as the time of the first published message or it just happens that the problem really ended just as it was being reported.

The previous three incidents are excluded from time interval analysis. The other two categories of interval anomalies that are included in analysis have natural explanations:

- All of 33 incidents with $\Delta t_1 < 0$ have their first message published after the problem has actually ended. These all follow the template of first message being "we are investigating" with a later message confirming that the problem was already solved by the time the first incident message was published.
- Finally 16 incidents with $\Delta t_2 = 0$ have only a single message per incident informing of the incident start and end times. These messages are reporting of an incident that has already ended.

Without going deeply into analyzing distributions of the interval values it is possible to note that they may match several logarithmic distributions, with log-normal distribution being a possible candidate. Instead of modeling the distribution, the mean value and the confidence interval (e.g. estimation error) are calculated using the studentized boostrap method with the results in Table 5.

There is significant uncertainty about both the incident mean length ($\Delta t = 130 \pm 20$) and its range, especially since the upper bound for incident length covering 95% of all incidents is high at 600 minutes. If these values are used for incident response planning the following needs to be considered:

- The mean time for AWS to acknowledge an outage (Δt_s) is 73 minutes with the upper likelihood bound being over five times that (400 minutes). This means that if a problem in AWS is suspected it may take an hour or more for the incident to be confirmed by AWS.
- The time from incident start to last published message for the incident (which usually is a message reporting the incident as resolved)

[16] Incidents with only one message and negative Δt_1 values are excluded.

[17] Incidents with only one message are excluded.

		Mean (minutes)	**95%**
Incident length	Δt	130 ± 20	11–600
Incident start to first message	Δt_s	73 ± 14	7–400
Incident end to last message	Δt_e	43 ± 10	5–300
First message to incident end[16]	Δt_1	90 ± 20	3–400
First message to last message[17]	Δt_2	110 ± 20	9–460
Incident start to last message	Δt_3	170 ± 20	38–800

Table 5: For all regions and services, incident length and its standard error, and the range of values containing 95% of incident times determined by the studentized bootstrap method.

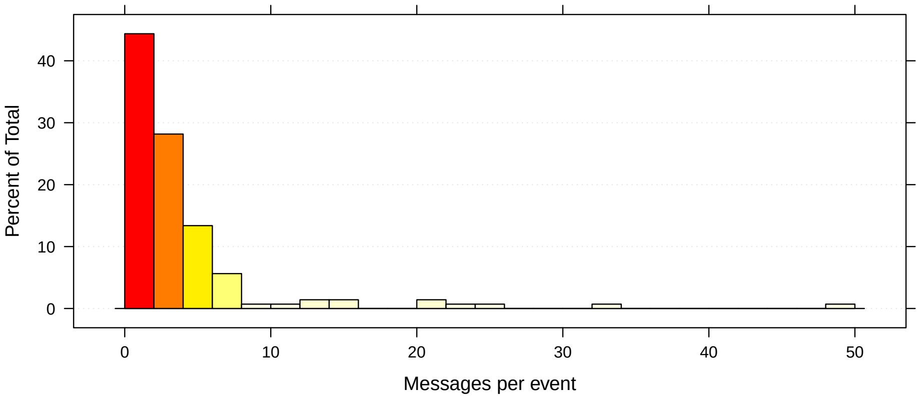


Figure 7: Distribution of messages per event.

is almost three hours at 170 minutes. The upper likelihood bound is again much higher (800 minutes). This means that it may take quite some time that even after the incident appears to be over to receive confirmation on the incident's end.

A somewhat tongue-in-cheek suggestion for incident response is that if AWS is suspected of being the root cause then one should be prepared for a long wait. Order in.

5.4 Outages

The previous section looked at incidents and their statistics. An incident, as shown in Figure 6, can consist of multiple distinct outages affecting one or more services in the same or in other regions. This difference is significant. For example incidents with explicitly given start and end times the total

Regions	Services								
	1	2	3	4	5	6	8	Total	
0	15 (15)	6 (6)	0	0	0	0	0	21 (21)	15.1%
1	88	13 (3)	3	3 (1)	2	2	2 (1)	113 (5)	81.3%
2	1	3	0	0	1 (1)	0	0	5 (1)	3.6%
Total	104 (15)	22 (9)	3	3 (1)	3 (1)	2	2 (1)	139 (27)	
	74.8%	15.8%	2.2%	2.2%	2.2%	1.4%	1.4%		

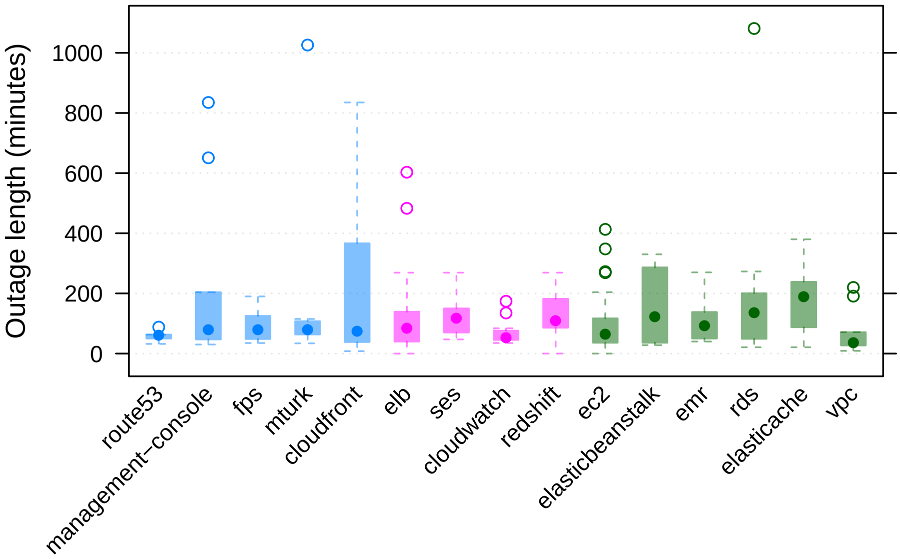
Table 6: Number of incidents by the number of regions and services affected. Number of parenthesis shows how many of the affected incidents affected also global services, for example at the table cell at one region and four services has “3 (1)” meaning there were three outages affecting one region and four services, and one of these outages also included at least one global service that was affected.

length of incidents was 262 hours during the study period while the same total for outages (not incidents) is much larger at 377 hours. This implies that many incidents affect multiple services. As shown in Table 6 over a quarter of incidents affect more than one service, with an average of 1.5 affected services per incident. The number of incidents that affect multiple regions or a global service and a region is much lower at less than 4% of all incidents. This means that 1) majority of all incidents affect only a single service in a single region or only a global service, and 2) an incident with wider impact is more likely to affect multiple services than multiple regions.

The average length of a single outage affecting a single service in a single region is 126 ± 11 minutes. This is practically the same as the average incident length of 130 ± 20 minutes. There are differences between regions and services in the average length of outages (see Figure 8). The most pronounced difference occurs between the `ap-northeast-1` (average outage length and 37 ± 9 minutes) `sa-east-1` regions (230 ± 60 minutes) and the Route 53 service (59±9 minutes) and CloudFront service (250±130 minutes). Some regions and services also have a larger number of outliers than others as shown in Figure 8.

5.5 Availability

Counting incidents and outages is straightforward and the average length of incidents and outages can be estimated with confidence intervals. While the historical availability is simply the proportion of operational time to the total service time the resulting number is not always useful. This is because there are large uncertainties in both outages and the total service time. First, if a service had few or zero outages was this really because the service was reliable? Also consider a service with one large outage, or a service with many outages but both with the same total outage time. While both would



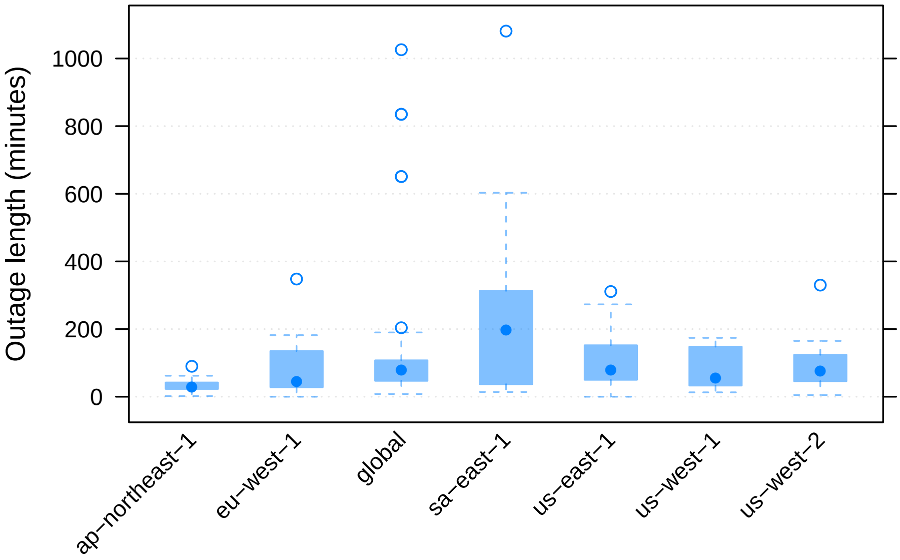


Figure 8: Box-and-whiskers of outage lengths by service and region. Regions and services with less than five outages and total outages less than one hour are omitted. Services are grouped so that first are global services, then regional services and last are zone-based services.

have the same availability metric it would be more accurate (e.g. with less uncertainty) for the latter service. For this reason availability analysis omits services with five or less outages.

Secondly we do almost never know the scope of an outage — how many customers or cloud resources were actually affected by the outage? Do different types of root causes have different impacts? Does service degradation count as an outage? While AWS does provide some qualitative descriptions of outage impact, this thesis makes the simplifying assumption that a service outage or degradation of any kind affected 100% of all customers or resources of that service (e.g. all customers for global services, the whole region for regional services and an availability zone for zone-based services). This means that from customer's point of view all availability estimates are under-estimates — it is likely that the service in question is more reliable.

Thirdly the total service time can be estimated only as a wall-clock time meaning that differences in capacity between different regions cannot be taken into account. For example it is reasonable to assume that Singapore region would have had more server and storage capacity than the Sao Paolo region simply because it had been operation for a longer time. In this analysis both regions have the same size of two availability zones.

When neither the relative customer impact nor capacity differences between services and regions is known this leads a disadvantage of global and regional services over zone-based services. During the study period there was a total of 23 availability zones. Thus an outage in any zone-based service will have an impact of just $^1/_{23}$ of a global service when the total outage time is calculated. This does have an effect of underestimating the availability of global compared to regional and zone-based services and of regional services compared to zone-based services.

With these limitations in mind this thesis looks at outages at the smallest externally measurable unit level (whole service, a region or an availability zone). This unfortunately means that what here is considered "unavailable" is the time *some unknown portion of customers is affected* by an outage and not the pro-rata (per customer or per resource) availability. Without knowing the portion of customers or resources affected in an outage it is possible to define availability only at the unit level.

Regardless it is now possible to define specific questions as measured by grouping the operational and service time on different axis:

1. What is the *general* availability of AWS's services?[18]

$$A_g = \sum_a O / \sum_a T = 99.983\%$$

A_g is thus simply the proportion of sum of operational time over all

[18] For simplicity O and T are taken to apply to whatever is relevant in context. In strict notation the sum over O would actually need to be $\sum_{u \in a} O(u)$.

measurable units a (services, or services and regions, or services, regions and zones). It is a general metric and given the biases listed above is an underestimate.

2. What is availability *by region*, where global services are counted to being in a pseudo-region “global”?

$$A_r = \sum_{a(r)} O / \sum_{a(r)} T$$

Here A_r is the availability for region r and $a(r)$ is the set of measurable units in that region. The results are shown in Table 11. Given the biases listed above, this is likely to be a significant underestimate for global services, for which $A_{\text{global}} = 99.830\%$ whereas all real (physical) regions have availability of $A_{\text{physical}} = 99.964\%$ or better (e.g. “three nines”.) There is also a difference between all physical regions attaining 99.990% or better *except* for $A_{\text{us-east-1}} = 99.967\%$ and $A_{\text{sa-east-1}} = 99.964\%$. The lower availability for US East could be explained by its age — being the very first region publicly available it might have more resources and capacity (more servers, more disks) meaning that the relative impact of its outages might be actually lower than in other regions. Since this work does not try to adjust for the relative impact of an outage this would give more weight to outages in the US East region. An alternative explanation is that with the US East being the “oldest” region it might actually have structural problems (legacy) that have been addressed in newer regions. The Sao Paolo region’s low availability might be partly because it was relatively recent (second youngest), although it being a reflection of true lower reliability cannot be dismissed either.[19]

3. What is availability *by service*, so that regional and zone-based services are aggregated over all regions and zones?

$$A_s = \sum_{a(s)} O / \sum_{a(s)} T$$

Here A_s is the availability for service s and $a(s)$ is the set of measurable units for that service globally. The results are shown in Table 11 and again as noted before, the global and regional services are relatively underestimated more than zone-based services. For this reason all comparisons should be made only within the same class of services and especially the availability values for global services need to be taken with a pinch of salt. For example neither $A_{\text{cloudfront}} = 99.671\%$ or $A_{\text{route53}} = 99.944\%$ apparently meet their

[19] The author has anecdotally heard that operating in Brazil was at least initially more difficult than in other regions due to difficulties and delays in sourcing hardware, for example.

specified SLA availability targets (99.9% and 99.988%[20] correspondingly). CloudFront suffers from one particular bias more, as its service is actually delivered from *edge locations* which there are more than fifty in total. Thus an outage in a single edge location will in this statistics cause it to appear as a failure of all edge locations. Unfortunately CloudFront's edge locations cannot be counted in the same manner as availability zones, for example, because a customer has only very limited control over the use of edge locations. This leaves a conundrum — count failures by edge locations, or as a single global service? For simplicity this work makes the latter choice.

Route 53 has another problem in that its SLA actually talks about not the general service availability, but that it responds correctly to DNS queries. Inspecting actual Route 53 outage messages shows that there were no problems reported of actual DNS query problems — the problems either affected Route 53 API (which is not covered by SLA) or there were delays in propagation of changes (but old data was still served correctly). This makes the simplifying assumption that any report of any kind of real incident affects the measured availability of the service. This means that availability as defined in this paper does no necessarily match that of SLA's, although for a user of Route 53 its API problems or slow propagation times may be relevant and for that use the availability result from this work may be useful.

There may be other reasons for low availability, as for example with SES at $A_{\mathrm{ses}} = 99.871\%$. The SES service was made available for customers during the study period, initially in one region only and later expanded to three regions. This means that SES had only $T_{\mathrm{ses}} = 690$ days. This means that a single incident has a larger relative weight than for a service that was operational for the whole year. It is also possible that AWS's reporting threshold is lower (e.g. reporting practices still in state of change) for new services. Of course it is possible that a new service suffers from "teething problems" and was actually performing worse during the study period than later.

Apart from CloudFront and Route 53 the only services that had a defined SLA during the study period were EC2 and RDS services. While both of these have published SLA targets of 99.95% ***these are not comparable*** to availabilities reported in Table 11. EC2 and RDS define "non-availability" of the service as when it is not available in a region in two zones simultaneously. Thus the *single-zone availability* needs to be adjusted (see Appendix A) to a *two-zone availability* value that actually is comparable to the SLA availablity target. The original observed single-zone availability, the calculated two-zone equivalent availability and the SLA target are shown in Table 7. Making this apples-to-apples comparison shows that EC2 and RDS services

[20] Route 53's SLA sets out an availability target of 100%, but if the total monthly outage time is less than 5 minutes no credits are paid back — 5 minutes a month corresponds to a target of 99.988%.

	Observed		SLA
Service	**Single-zone Availability**	**Two-zone Availability**	**Availability Target**[21]
ec2	99.9510000%	99.9999760%	99.9500000%
rds	99.9750000%	99.9999938%	99.9500000%

Table 7: Comparison of observed EC2 and RDS services single-zone availability and the calculated two-zone availability to service level agreement availability targets.

meet the SLA target with a substantial margin — the SLA-equivalent availability for both is more than 99.9999%.

6 Discussion

This work uses a viewpoint of external passive observer to infer availability of several AWS services. This offers both benefits such as low cost of data acquisition but has disadvantages such as coarse spatial granularity (a single availability zone level) and has numerous potential biases such as unknown reporting thresholds. It is also important to realize that the "availability questions" that can be answered based on this are limited to those working on higher system abstractions such as availability zones, regions or services and not on individual servers — the incident information used for this work apparently omits most "everyday" failures such as individual server failures (reboots), transient and small-scale errors within network, regional and global services and so on.

The value of 99.95% is often cited (incorrectly) as an availability target of many cloud services [Goo; Ama13a; Mic14]. Even against this incorrectly interpreted metric of global service availability AWS exceeds it at 99.983% availability over all services and regions as measured in this work. When the observed single-zone availability from this work is converted to SLA-equivalent simultaneous failures of two or more zones both EC2 and RDS meet the 99.95% target with a significant margin with EC2's SLA-equivalent availability f 99.9999760% and RDS's of 99.9999938%. The fact that AWS's services achieve a high level of externally measured availability should not be a surprise — after all AWS has been commercially successful and it would be implausible to assume that such success could be achieved with a poor level of service availability.

Naldi used a similar method to analyze public outage data, although using a different data sources than this work [Nal13]. Naldi provides results

[21] The SLA availability target considers a service unavailable if it is not available in at least two availability zones simultaneously. Thus the SLA target is value should be compared to the *two-zone availability* value.

for the number of outages and differentiates between outages that have known length and provides average outage length and inter-outage interval for those. In Naldi's analysis for AWS a total 21 outages were included of which 16 had duration information. This work includes a total of 139 incidents consisting of a total of 225 separate outages. In Naldi's study the average outage length is 474 minutes which is significantly larger than this study's 130 ± 20 minutes per incident[22].

While the intervals from which data was collected in Naldi's and this study are hardly equivalent[23] the significant difference in both the number of observed events and the higher average length in Naldi's study do suggest that incidents reported by media, customers and other interested parties are biased towards highly visible events. This means that Naldi's assertion of "those that have not been recorded are probably incidents of quite minor relevance" [Nal13, pp. 284] is not supported in light of this work.

Fiondella et al. use also Cloutage as their data source, but break down the analysis by category into CloudFront, CloudWatch, EBS, EC2 and S3 [FGM13]. Fiondella et al. counted the number of outages for AWS, identifying EC2 as the service with the largest number of outages. In their study 53% of all AWS outages were attributed to EC2. This is in line with the results from this study which finds EC2 having more outages than all other services combined.

This work demonstrated the difficulty of getting factual availability estimates for public infrastructure cloud services. First and foremost cloud vendors themselves do not publish any kind of availability or reliability metrics for their services. There are many public services in other B2B and B2C areas that *do provide* both real-time and historical service quality information publicly, often including measurable metrics such as system availability or response time. Currently major cloud vendors do not publish measurable quality metrics of their services.

In the same note Naldi writes that "in the absence of institutional monitoring and assessment activities, we must rely on both the providers' reports and the news gathered by third party entities, which in turn rely on customers' indications." [Nal13] Fiondella et al note that most incidents do not provide information on the number of affected customers, thus "some outages may have a significant impact, while others may go unnoticed" [FGM13]. Although this study tries to minimize the customers' bias of reporting only large outages by using the incident information from AWS dashboard, it is still hampered by the fact that practically all of the reporting processes are opaque — we do not know what are the thresholds at cloud

[22] Naldi refers to "outages" which I have taken to refer to what are referred to "incidents" in this paper due to Naldi's sources focusing on large-scale events and not providing per-service or per-region separation breakdowns.

[23] The time period in length is different and the periods do not overlap. Cloud services and their use has increased over time which would need to be taken into account too.

service providers for reporting incidents in the dashboard nor whether they are the same between different regions, services or over time.

Different actors in the cloud computing market have different needs for availability metrics. A customer is interested in historical and present situation of the services they are using or planning to use. If these metrics are used for reliability modeling then more descriptive statistical values are needed. A customer may also want to compare the service quality between cloud vendors. Depending on the use case a simple metric may suffice, for other cases descriptive statistics are required. Doing meaningful comparisons between different vendors is only possible if definitions of the published metrics are standardized or the incident data itself can be normalized to conform to comparable metrics.

At the moment few if any of these requirements are met. Even while the “big three” cloud vendors (AWS, Azure, Google) do publish incident information on the web the information and its availability has limitations. Compared to industries such as telecommunications or power utilities there are no *de facto* or *de jure* requirements for recording or publishing incident information, and no requirements for cloud vendors to provide accurate or standardized periodic summaries of their service quality. As an example of how other markets are regulated, consider legislation of the European Union that requires energy market participants to conform to a set of reporting rules [EC11]. The reporting rules regulate how and what to report when, for example an outage occurs in a power station [Nor14]. Similarly the nuclear industry, oil and gas industry and several others compile reliability data either for industry’s internal use, or by the customers of the industry [Uni15; Wik15].

7 Further Work

This work has taken a frequentist (classical) approach to statistics. This means all of the availability metrics in this work as they stand are only applicable to looking at what happened from mid-2013 to mid-2014. These values can be used in reliability models to analyze future scenarios only if they are assumed to be valid also now and in the future. In the rapidly developing cloud services market this may be an invalid assumption. An alternative Bayesian approach would allow the observer to integrate new information into the current availability knowledge as it becomes available and generate more relevant posterior probabilities.

The frequentist approach is also problematic when estimating the likelihood of low-probability events from only a few (or none at all!) outages. For example see Williams and Thorne [WT97] for a comprehensive discussion on problems and methods related to estimation of failure rates for low-probability events. This work chose to analyze in detail only services and regions with at least five outages which means that for example `ap-southeast-1` and `ap-southeast-2` regions are excluded from detailed analysis and for services more than half of all commercially available services do not meet this threshold. Note that while these services are included in summary statistics and thus contribute towards aggregate region and overall service availability the author did considers publishing apparently "perfect" availability values as something that does not provide useful information towards potential users of the availability metrics.

This work also purposefully omits all analyses on probability distributions (e.g. does the distribution of incident lengths follow gamma distribution or not and so on). Primarily this is due to the need to keep the scope of this work manageable, but secondarily because fitting distributions to data is not meaningful without having a prior hypothesis. It is always possible to find spurious correlations when a large number of different models and fitting parameters are tried. Considering a value such as Δt_s it would be more beneficial to first have a model of the underlying processes (in this case the model would include service monitoring automation, human response times etc.) and then seeing how well the models fits the data. The author looked into the Δt_s distribution and thinks it might be a mixture distribution, suggesting that there are multi-stage processes with distinctly different parameters at play when an incident is detected and subsequently reported.

Although both in this work and in many other analyses of cloud failures it is assumed that cloud infrastructure failures are not correlated between availability zones and regions, there are known failures that have affected services in multiple regions[24]. This data also contains incidents that have affected multiple different services. Since most reliability modeling mechanisms assume an uncorrelated failure model even a slight correlation between service failures could potentially lead to a large error in the analysis. For this reason understanding of service failure correlations could be important.

While any kind of reporting of service reliability by cloud vendors would be useful for customers, for researchers a more serious problem is the lack of open data sets. While most cloud vendors have their outage information available over the network, their access methods and formats differ and there is a retention limit on the data after which it is no longer available. These both mean that any analysis (such as done in this work) needs to actively collect the data and keep monitoring for changes in access methods. Any improvement in this area would potentially have a huge benefit for researchers in the future.

[24] Elastic Beanstalk had a performance degradation across all regions on June 25th 2013.

Acknowledgments

I would like to thank all the people who have supported me in writing this thesis. This has been a long process. A big thanks goes to professor Sasu Tarkoma for accepting the thesis topic that I presented out of the blue and my graduate thesis instructors Toni Ruottu and Eemil Lagerspetz. Specifically I would like to thank Elina for practical support at home and Pia from work in whom I found a fellow sufferer of thesis angst. While I have probably bored uncounted number of people on the intricacies of cloud availability measurement all of the people who have suffered me in silence have been very important helping me progress bit by bit. I hardly can remember all by name but thanks to at least Roope, Juri, Olli, Sami, Tommi, Marko. Also there are several people from AWS who I have squeezed unofficially at various events for bits of insider information that I'd like to anonymously thank.

References

[Ama12a] Amazon Web Services. *Summary of the Amazon EC2, Amazon EBS, and Amazon RDS Service Event in the EU West Region.* Amazon Web Services, Inc. Aug. 7, 2012. URL: `http://aws.amazon.com/message/2329B7/` (visited on 12/14/2014).

[Ama12b] Amazon Web Services. *Summary of the AWS Service Event in the US East Region.* Amazon Web Services, Inc. July 2, 2012. URL: `https://aws.amazon.com/message/67457/` (visited on 02/07/2016).

[Ama12c] Amazon Web Services. *Summary of the December 24, 2012 Amazon ELB Service Event in the US-East Region.* Dec. 2012. URL: `http://aws.amazon.com/message/680587/` (visited on 03/26/2013).

[Ama13a] Amazon Web Services. *Amazon EC2 SLA.* June 1, 2013. URL: `http://aws.amazon.com/ec2-sla/` (visited on 04/10/2011).

[Ama13b] Amazon Web Services. *Amazon RDS SLA.* Amazon Web Services, Inc. June 1, 2013. URL: `http://aws.amazon.com/rds/sla/` (visited on 03/06/2015).

[Ama13c] Amazon Web Services. *AWS Service Health Dashboard*. 2013. URL: http://status.aws.amazon.com/ (visited on 04/10/2013).

[Ama13d] Amazon Web Services. *CloudFront SLA*. Amazon Web Services, Inc. June 1, 2013. URL: http://aws.amazon.com/cloudfront/sla/ (visited on 11/20/2015).

[Ama15] Amazon Web Services. *Amazon Route 53 SLA*. Amazon Web Services, Inc. May 15, 2015. URL: http://aws.amazon.com/route53/sla/ (visited on 11/20/2015).

[Avi+04] A. Avizienis et al. "Basic concepts and taxonomy of dependable and secure computing". In: *IEEE Transactions on Dependable and Secure Computing* 1.1 (Jan. 2004), pp. 11–33. ISSN: 1545-5971. DOI: 10.1109/TDSC.2004.2.

[BA12] Eric Bauer and Randee Adams. *Reliability and availability of cloud computing*. Hoboken, N.J: Wiley-IEEE Press, 2012. 323 pp. ISBN: 978-1-118-17701-3.

[Bar08] Jeff Barr. *Animoto - Scaling Through Viral Growth*. Apr. 20, 2008. URL: http://aws.typepad.com/aws/2008/04/animoto---scali.html (visited on 03/06/2011).

[BEL13] O. Beaumont, L. Eyraud-Dubois, and H. Larchevêque. "Reliable Service Allocation in Clouds". In: *2013 IEEE 27th International Symposium on Parallel Distributed Processing (IPDPS)*. 2013 IEEE 27th International Symposium on Parallel Distributed Processing (IPDPS). 2013, pp. 55–66. DOI: 10.1109/IPDPS.2013.64.

[Ber+13] I. Bermudez et al. "Exploring the cloud from passive measurements: The Amazon AWS case". In: *2013 Proceedings IEEE INFOCOM*. 2013 Proceedings IEEE INFOCOM. 2013, pp. 230–234. DOI: 10.1109/INFCOM.2013.6566769.

[BH09] Luiz André Barroso and Urs Hölzle. "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines". In: *Synthesis Lectures on Computer Architecture* 4.1 (Jan. 2009), pp. 1–108. ISSN: 1935-3235, 1935-3243. DOI: 10.2200/S00193ED1V01Y200905CAC006.

[Bil12] Bill Laing. *Summary of Windows Azure Service Disruption on Feb 29th, 2012 | Microsoft Azure Blog*. Mar. 9, 2012. URL: http://azure.microsoft.com/blog/2012/03/09/summary-of-windows-azure-service-disruption-on-feb-29th-2012/ (visited on 11/14/2014).

[BK14] Peter Bailis and Kyle Kingsbury. "The Network is Reliable". In: *Queue* 12.7 (July 2014), 20:20–20:32. ISSN: 1542-7730. DOI: 10.1145/2639988.2639988.

[Clo16] CloudHarmony. *CloudSquare*. 2016. URL: `https://cloudharmony.com/cloudsquare` (visited on 04/09/2016).

[Coc12] Adrian Cockcroft. *The Netflix Tech Blog: A Closer Look At The Christmas Eve Outage*. Dec. 31, 2012. URL: `http://techblog.netflix.com/2012/12/a-closer-look-at-christmas-eve-outage.html` (visited on 12/14/2014).

[DH99] A. C. Davison and D. V. Hinkley. *Bootstrap Methods and Their Application*. Cambridge Series on Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press, 1999. ISBN: 978-0-521-57471-6.

[EC11] European Parliament and Council of the European Union. *Council regulation EU no 1227/2011*. Oct. 25, 2011.

[Eus+08] Irene Eusgeld et al. "Hardware Reliability". In: *Dependability Metrics*. Ed. by Irene Eusgeld, Felix C. Freiling, and Ralf Reussner. Lecture Notes in Computer Science 4909. Springer Berlin Heidelberg, Jan. 1, 2008, pp. 59–103. ISBN: 978-3-540-68946-1 978-3-540-68947-8.

[Far+13] Hamid Reza Faragardi et al. "An analytical model to evaluate reliability of cloud computing systems in the presence of QoS requirements". In: *2013 IEEE/ACIS 12th International Conference on Computer and Information Science (ICIS)*. 2013 IEEE/ACIS 12th International Conference on Computer and Information Science (ICIS). 2013, pp. 315–321. DOI: `10.1109/ICIS.2013.6607860`.

[FGM13] L. Fiondella, S.S. Gokhale, and V.B. Mendiratta. "Cloud Incident Data: An Empirical Analysis". In: *2013 IEEE International Conference on Cloud Engineering (IC2E)*. 2013 IEEE International Conference on Cloud Engineering (IC2E). 2013, pp. 241–249. DOI: `10.1109/IC2E.2013.28`.

[For+10] Daniel Ford et al. "Availability in Globally Distributed Storage Systems." In: OSDI. 2010, pp. 61–74.

[Geo+13] B. George et al. "Mission critical cloud computing in a week". In: *2013 IEEE Aerospace Conference*. 2013 IEEE Aerospace Conference. 2013, pp. 1–7. DOI: `10.1109/AERO.2013.6497326`.

[GH12] A.J. Gonzalez and B.E. Helvik. "System management to comply with SLA availability guarantees in cloud computing". In: *2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom)*. 2012 IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom). 2012, pp. 325–332. DOI: `10.1109/CloudCom.2012.6427508`.

[Goo] Google. *Google Compute Engine Service Level Agreement (SLA)*. Google Developers. URL: `https://cloud.google.com/compute/sla` (visited on 11/16/2014).

[Goo15] Google. *Google Cloud Status*. 2015. URL: `https://status.cloud.google.com/` (visited on 12/01/2015).

[Hal88] Peter Hall. "Theoretical Comparison of Bootstrap Confidence Intervals". In: *The Annals of Statistics* 16.3 (Sept. 1988), pp. 927–953. ISSN: 0090-5364, 2168-8966. DOI: `10.1214/aos/1176350933`.

[HP13] G. Hogben and A. Pannetrat. "Mutant Apples: A Critical Examination of Cloud SLA Availability Definitions". In: *2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom)*. 2013 IEEE 5th International Conference on Cloud Computing Technology and Science (CloudCom). Vol. 1. Dec. 2013, pp. 379–386. DOI: `10.1109/CloudCom.2013.56`.

[ISO14] ISO/IEC. *ISO/IEC STANDARD 17788: Information technology — Cloud computing — Overview and vocabulary.* ISO/IEC 17788:2014(E). 2014.

[Jas14] Jason Zander. *Final Root Cause Analysis and Improvement Areas: Nov 18 Azure Storage Service Interruption*. Microsoft Azure Blog. Dec. 17, 2014. URL: `http://azure.microsoft.com/blog/2014/12/17/final-root-cause-analysis-and-improvement-areas-nov-18-azure-storage-service-interruption/` (visited on 12/20/2014).

[Kha+12] H. Khazaei et al. "Availability analysis of cloud computing centers". In: *2012 IEEE Global Communications Conference (GLOBECOM)*. 2012 IEEE Global Communications Conference (GLOBECOM). 2012, pp. 1957–1962. DOI: `10.1109/GLOCOM.2012.6503402`.

[Lou14] Louis Columbus. *Roundup Of Cloud Computing Forecasts And Market Estimates, 2014*. Forbes. Mar. 14, 2014. URL: `http://www.forbes.com/sites/louiscolumbus/2014/03/14/roundup-of-cloud-computing-forecasts-and-market-estimates-2014/` (visited on 12/14/2014).

[MG09] P. Mell and T. Grance. "The NIST definition of cloud computing". In: *National Institute of Standards and Technology (NIST)* (Sept. 2009).

[Mic14] Microsoft. *Microsoft Azure Support: Service Level Agreement*. Nov. 2014. URL: `http://azure.microsoft.com/en-us/support/legal/sla/` (visited on 11/16/2014).

[Mic15] Microsoft. *Azure Status.* 2015. URL: https://azure.microsoft.com/en-us/status/#current (visited on 12/01/2015).

[Mik12] Mike Neil. *Root Cause Analysis for recent Windows Azure Service Interruption in Western Europe.* Microsoft Azure Blog. Aug. 2, 2012. URL: http://azure.microsoft.com/blog/2012/08/02/root-cause-analysis-for-recent-windows-azure-service-interruption-in-western-europe/ (visited on 12/14/2014).

[MM11] N. Milanovic and B. Milic. "Automatic Generation of Service Availability Models". In: *Services Computing, IEEE Transactions on* 4.1 (2011). MiM11, pp. 56–69. ISSN: 1939-1374. DOI: 10.1109/TSC.2010.11.

[Nal13] M. Naldi. "The availability of cloud-based services: Is it living up to its promise?" In: *Design of Reliable Communication Networks (DRCN), 2013 9th International Conference on the.* Design of Reliable Communication Networks (DRCN), 2013 9th International Conference on the. Mar. 2013, pp. 282–289.

[NDO11] Edmund B. Nightingale, John R. Douceur, and Vince Orgovan. "Cycles, Cells and Platters: An Empirical Analysis of Hardware Failures on a Million Consumer PCs". In: *Proceedings of the Sixth Conference on Computer Systems.* EuroSys '11. New York, NY, USA: ACM, 2011, pp. 343–356. ISBN: 978-1-4503-0634-8. DOI: 10.1145/1966445.1966477.

[Nor14] Nord Pool Spot AS. *Guidelines for publishing Urgent market messages.* Dec. 15, 2014. URL: http://www.nordpoolspot.com/globalassets/download-center/market-surveillance/guidelines-for-publishing-umm.pdf.

[Ris+09] Thomas Ristenpart et al. "Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds". In: *Proceedings of the 16th ACM conference on Computer and communications security.* CCS '09. New York, NY, USA: ACM, 2009, pp. 199–212. ISBN: 978-1-60558-894-0. DOI: 10.1145/1653662.1653687.

[Sah+04] R.K. Sahoo et al. "Failure data analysis of a large-scale heterogeneous server environment". In: *2004 International Conference on Dependable Systems and Networks.* 2004 International Conference on Dependable Systems and Networks. June 2004, pp. 772–781. DOI: 10.1109/DSN.2004.1311948.

[SG10] B. Schroeder and G.A. Gibson. “A Large-Scale Study of Failures in High-Performance Computing Systems”. In: *IEEE Transactions on Dependable and Secure Computing* 7.4 (Oct. 2010), pp. 337–350. ISSN: 1545-5971. DOI: 10.1109/TDSC.2009.4.

[Sim09] Simon Wardley. “Cloud Computing - Why IT Matters”. OSCON 09, July 25, 2009. URL: https://www.youtube.com/watch?v=okqLxzWS5R4 (visited on 12/12/2014).

[SJ13] Seokho Son and Sung Chan Jun. “Negotiation-Based Flexible SLA Establishment with SLA-driven Resource Allocation in Cloud Computing”. In: *2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid).* 2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid). May 2013, pp. 168–171. DOI: 10.1109/CCGrid.2013.81.

[SKK14] Seokho Son, Dong-Jae Kang, and Jin-Mee Kim. “Design considerations to realize automated SLA negotiations in a multi-Cloud brokerage system”. In: *2014 International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom).* 2014 International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom). Oct. 2014, pp. 466–468.

[Smi11] David J. Smith. *Reliability, Maintainability and Risk - Practical Methods for Engineers (8th Edition).* Elsevier, June 29, 2011. 464 pp. ISBN: 978-0-08-096902-2.

[SPW09] Bianca Schroeder, Eduardo Pinheiro, and Wolf-Dietrich Weber. “DRAM Errors in the Wild: A Large-scale Field Study”. In: *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems.* SIGMETRICS ’09. New York, NY, USA: ACM, 2009, pp. 193–204. ISBN: 978-1-60558-511-6. DOI: 10.1145/1555349.1555372.

[SV02] Norbert Schwarz and Leigh Ann Vaughn. “The Availability Heuristic Revisited: Ease of Recall and Content of Recall as Distinct Sources of Information”. In: *Heuristics and Biases.* Cambridge University Press, 2002. ISBN: 978-0-511-80809-8.

[Uni15] United States Nuclear Regulatory Commission. *NRC Reactor Operating Experience Data.* 2015. URL: https://nrod.inl.gov/ (visited on 11/20/2015).

[VN10] Kashi Venkatesh Vishwanath and Nachiappan Nagappan. “Characterizing Cloud Computing Hardware Reliability”. In: *Proceedings of the 1st ACM Symposium on Cloud Computing.* SoCC

’10. New York, NY, USA: ACM, 2010, pp. 193–204. ISBN: 978-1-4503-0036-0. DOI: 10.1145/1807128.1807161.

[Wik15] Wikipedia. *OREDA*. In: *Wikipedia, the free encyclopedia.* Page Version ID: 687641451. Oct. 26, 2015. URL: https://en.wikipedia.org/w/index.php?title=OREDA&oldid=687641451 (visited on 11/20/2015).

[WT97] M. M. R. Williams and M. C. Thorne. “The estimation of failure rates for low probability events”. In: *Progress in Nuclear Energy* 31.4 (1997), pp. 373–476. ISSN: 0149-1970. DOI: 10.1016/S0149-1970(96)00022-4.

[Xia+15] Yuan Xiaoyong et al. “An Analysis on Availability Commitment and Penalty in Cloud SLA”. In: *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual.* Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual. Vol. 2. July 2015, pp. 914–919. DOI: 10.1109/COMPSAC.2015.39.

A Service Level Agreement Availability vs. Zone Availability

The oft-quoted 99.95% availability target from AWS, Azure and Google SLAs must be taken with a pinch of salt. These SLAs differ in details, but share the fact that *they refer availability as the complement of the simultaneous unavailability of two or more zones.* Let's we define A_i to be *true* if zone i is available and *false* if it is unavailable at some indeterminate time. Given this notation the SLA statement becomes an assertion that Equation 4 holds[25]:

$$\begin{aligned} A = (\forall i : A_i) \vee (\exists\, i : \neg A_i \wedge \forall j, j \neq i : A_j) \\ P(A \text{ is true}) \geq 99.95\% \end{aligned} \tag{4}$$

A would be false only when there exist two (or more) simultaneous zone failures. Assuming all A_i failures are independent and that $P(A_i) = p_A$, we can calculate probability for *at least two zones* in a region with n zones failing simultaneously:

$$\begin{aligned} P(\neg A) &= (1 - p_A)^2 + \cdots + (1 - p_A)^n \\ P(\neg A) &\approx (1 - p_A)^2 \end{aligned} \tag{5}$$

For an approximation in Equation 5 it is possible to omit higher-order terms as the probability for three or more zones failing simultaneously is small for any $p_A \approx 1$. Now we can put $P(\neg A)$ back into Equation 4 and calculate the minimum availability for one zone p_A:

$$\begin{aligned} P(A) &\geq 99.95\% \\ 1 - P(\neg A) &\geq 99.95\% \\ 1 - 1 + 2p_A - p_A^2 &\geq 99.95\% \\ -p_A^2 + 2p_A - 99.95\% &\geq 0 \\ p_A &\geq 97.7639\% \end{aligned}$$

This means that independent zones with availability *as low as* 97.77% will meet SLA's target of 99.95% availability. This looks pretty bad **if used out of context**, which is the reason why this paper compares SLA's values only with "SLA-equivalent" versions of p_A and discusses per-zone availability separately from SLA values.

[25] This is a simplification. The *real* SLA conditions are more convoluted.

B AWS Service Level Agreements

The service level agreements (SLAs) of services examined are summarized below in Appendix B. In all cases availability is calculated over a monthly billing cycle. All of the SLAs have exclusions limiting AWS's liability for outages out of their own control and *force majeure* events. Values in the table are valid as of November 1st 2015 when Amazon Web Services provided service level agreements for four of the services included in Table 11: CloudFront, EC2, RDS and Route 53.

Service	**Availability target**	**Definition of availability**
CloudFront	99.9%	This is a service *reliability* target — not an availability target. The metric is counted as a ratio of erroneous responses to the number of requests made during each 5 minute period in a month, with all 5 minute periods averaged to achieve a *monthly uptime percentage* [Ama13d].
EC2	99.95%	Availability is based on the proportion of time a region is unavailable. A region is considered unavailable when all running instances in two or more availability zones of that region are externally unreachable simultaneously. [Ama13a]
RDS	99.95%	Applies only to Multi-AZ RDS instances[26]. RDS is considered unavailable only when all connection attempts to it fail during a 1 minute period. [Ama13b]
Route 53	100%	Requires a minimum of 5 minutes of outages during a month for a customer to be eligible for credits [Ama15]. The service availability target refers only to DNS queries performed for Route 53 hosted DNS zones.

[26] Multi-AZ RDS has at least two instances configured to mirror each other with AWS providing transparent fail-over from primary to secondary database instance in case of primary RDS instance failure.

C Detailed Summaries of Messages, Incidents and Outages

	June 5th 2013 to June 4th 2014					
Region	**Messages**		**Incidents**		**Outages**	
ap-northeast-1	16	2%	7	5%	9	4%
ap-southeast-1	11	2%	4	3%	4	2%
ap-southeast-2	9	1%	2	1%	3	1%
eu-west-1	68	10%	18	12%	25	11%
global	104	16%	27	18%	36	16%
sa-east-1	87	13%	6	4%	21	9%
us-east-1	258	39%	61	41%	91	40%
us-west-1	58	9%	9	6%	17	8%
us-west-2	51	8%	16	11%	19	8%
Total	662	100%	150	100%	225	100%

Table 8: Number of messages, incidents and outages and their proportion of all by region and service.

	June 5th 2013 to June 4th 2014					
Service	**Messages**		**Incidents**		**Outages**	
Global						
alexa	0	0.0%	0	0.0%	0	0.0%
cloudfront	16	2.4%	7	3.3%	7	3.1%
fps	23	3.5%	8	3.8%	8	3.6%
iam	0	0.0%	0	0.0%	0	0.0%
management-console	15	2.3%	8	3.8%	9	4.0%
mturk	39	5.9%	7	3.3%	7	3.1%
route53	11	1.7%	5	2.4%	5	2.2%
Regional						
appstream	0	0.0%	0	0.0%	0	0.0%
cloudformation	6	0.9%	4	1.9%	4	1.8%
cloudhsm	9	1.4%	3	1.4%	3	1.3%
cloudsearch	0	0.0%	0	0.0%	0	0.0%
cloudtrail	0	0.0%	0	0.0%	0	0.0%
cloudwatch	32	4.8%	11	5.2%	12	5.3%
datapipeline	0	0.0%	0	0.0%	0	0.0%
directconnect	2	0.3%	1	0.5%	1	0.4%
dynamodb	0	0.0%	0	0.0%	0	0.0%
elastictranscoder	0	0.0%	0	0.0%	0	0.0%
elb	78	11.8%	26	12.3%	26	11.6%
glacier	0	0.0%	0	0.0%	0	0.0%
import-export	0	0.0%	0	0.0%	0	0.0%
kinesis	7	1.1%	2	0.9%	2	0.9%
opsworks	10	1.5%	1	0.5%	1	0.4%
redshift	12	1.8%	5	2.4%	5	2.2%
s3	8	1.2%	3	1.4%	3	1.3%
ses	29	4.4%	9	4.2%	10	4.4%
simpledb	0	0.0%	0	0.0%	0	0.0%
sns	8	1.2%	1	0.5%	1	0.4%
sqs	6	0.9%	2	0.9%	2	0.9%
storagegateway	0	0.0%	0	0.0%	0	0.0%
swf	3	0.5%	2	0.9%	2	0.9%
workspaces	9	1.4%	2	0.9%	3	1.3%
Zone-based						
ec2	178	26.9%	60	28.3%	66	29.3%
elasticache	23	3.5%	8	3.8%	8	3.6%
elasticbeanstalk	25	3.8%	8	3.8%	8	3.6%
emr	20	3.0%	6	2.8%	6	2.7%
rds	74	11.2%	16	7.5%	16	7.1%
vpc	19	2.9%	7	3.3%	10	4.4%
Total	662	100%	212	100%	225	100%

Table 9: Number of messages, incidents and outages and their proportion of all by service.

Service	**Count**	**Outage length**	
		Average	**Total**
		(minutes)	(days)
Global			
cloudfront	7	250 ± 130	1.2
fps	8	90 ± 20	0.51
management-console	9	220 ± 100	1.4
mturk	7	210 ± 140	1
route53	5	59 ± 9	0.2
Regional			
cloudwatch	12	69 ± 12	0.58
elb	26	130 ± 30	2.3
redshift	5	130 ± 50	0.45
ses	10	130 ± 20	0.89
Zone-based			
ec2	66	90 ± 10	4.1
elasticache	8	180 ± 40	0.99
elasticbeanstalk	8	160 ± 50	0.87
emr	6	110 ± 40	0.48
rds	16	190 ± 60	2.1
vpc	10	70 ± 20	0.48

Region	**Count**	**Outage length**	
		Average	**Total**
		(minutes)	(days)
ap-northeast-1	9	37 ± 9	0.23
eu-west-1	25	90 ± 20	1.5
global	36	170 ± 40	4.3
sa-east-1	21	230 ± 60	3.4
us-east-1	91	111 ± 9	7
us-west-1	17	85 ± 15	1
us-west-2	19	90 ± 20	1.2

Table 10: Number of outages, the sample mean outage length and sample error and total length of outages in minutes summarized by service and region. Regions and services with fewer than five or less than one hour of outages are omitted.

Service	**Outages / Service Time** (days)		**Availability**
	Global		
cloudfront	1.2	365	99.671%
fps	0.51	365	99.861%
management-console	1.4	365	99.616%
mturk	1	365	99.718%
route53	0.2	365	99.944%
	Regional		
cloudwatch	0.58	2920	99.980%
elb	2.3	2920	99.923%
redshift	0.45	2000	99.978%
ses	0.89	693	99.871%
	Zone-based		
ec2	4.1	8395	99.951%
elasticache	0.99	8395	99.988%
elasticbeanstalk	0.87	8395	99.990%
emr	0.48	8395	99.994%
rds	2.1	8395	99.975%
vpc	0.48	8395	99.994%

Region	**Outages / Service Time** (days)		**Availability**
ap-northeast-1	0.23	13383	99.998%
eu-west-1	1.5	14660	99.990%
global	4.3	2555	99.830%
sa-east-1	3.4	9598	99.964%
us-east-1	7	21376	99.967%
us-west-1	1	13558	99.993%
us-west-2	1.2	14744	99.992%

Table 11: Total length of outages compared to total service time for each atomic service unit. Note that for zone-based services this value is the observed availability for a single zone (see Appendix A on how to interpret this value in relation to availability targets given in service level agreements). Regions and services with fewer than five or less than one hour of outages are omitted.