Vrije Universiteit Amsterdam

Honours Programme, Project Report

How Do Cloud Services Fail? A Study of Provider-Reported Data

---

Author: Alexander Balgavy (2619644)

1st supervisor: Alexandru Iosup
daily supervisor: Sacheendra Talluri

A report submitted in fulfillment of the requirements for the Honours Programme, which is an excellence annotation to the VU Bachelor of Science degree in Computer Science/Artificial Intelligence/Information Sciences.

June 30, 2020

Abstract

Cloud computing is the paradigm that powers many of the services we use daily. We expect these services to be continuously available, but in reality, many of them fail. In this work, we analyze data collected from the public status pages of three leading cloud service providers. We carry out a classification and labelling of failure events during a one-year time span. Based on this classification, we observe meaningful patterns to answer the question of how cloud services fail. For example, we find that the majority of outages are caused by configuration errors, and that outages caused by increased load more often affect only some users in the range of the outage.

1 Introduction

Cloud computing as a paradigm has become a de-facto standard for services and applications. We build our modern distributed services in an increasingly 'cloud-native' manner, based on cloud-computing abstractions (IaaS, SaaS, etc.) and technologies (VMs, containers, functions, etc.). These 'cloud services' can be client-facing (such as the Netflix web application), or on the back-end (e.g. Amazon Simple Storage Service). In fact, many organisations that are essential for society (such as banking or healthcare) rely on cloud services which are invisible to end users [1], [2]. But how reliable are these services?

Though cloud services are expected to always be available, they can fail in a variety of ways. These failures can range from mild, such as a single-region outage affecting some users, to severe, such as a multi-region outage of multiple services. Furthermore, client-facing services are prone to upstream errors, and can fail when any one of their dependencies fails [3]. Such a failure results in service downtime, which can lead to disruption of critical services [4], [5], data loss [6], and other issues.

Moreover, many popular cloud services are built on top of other services. For example, Netflix uses Amazon Web Services to host their applications, with around 100 000 instances per day [7]. Due to this, a failure of one service can easily lead to the failure of many other services [8]–[10].

If we can understand the characteristics of these failures, we can potentially discern reasons for their occurrence. Therefore, we need to answer the question: *how do cloud services fail?* Characterising the failures can aid cloud providers in the prevention of outages, and allow them to further increase the availability and fault-tolerance of their services.

The main contribution of this study is a systematic analysis of provider-reported failures during one calendar year. We develop a framework for conducting such an analysis, which can be used in future studies. For me, the main contribution of this research was learning how to process and clean a large dataset, and how to iteratively develop a methodology to analyze the dataset.

2 Background information

Service architecture There are three major cloud service providers: Amazon (Amazon Web Services, AWS), Microsoft (Azure), and Google (Google Cloud Platform, GCP) [11], [12]. A provider makes available a number of services to its customers, and the customers are free to choose which services they want to use for their application. A service is not tied to one physical location, but can run in any of the regions offered by the provider, and can potentially span multiple regions. The geographical location of the service can often also be configured by the client. A particular service may depend on other services offered by the provider. For example, if a client decides to use AWS Elastic Beanstalk (EBS), they will also be relying on a number of other AWS resources, such as the Amazon Elastic Compute Cloud (EC2) and Amazon Simple Storage Service (S3) [13]. Each of these service dependencies may itself have other dependencies, which results in a complex dependency graph, where the failure of a single service may lead to

the outage of other services. Therefore, service availability is of utmost importance for cloud providers.

Terms & definitions We define a service *failure (event)* or *outage* as a period of time where the service is not functioning as expected. The *expected functionality* of a service is defined both explicitly in Service Level Agreements, and implicitly by the assumptions of clients and end users. During a failure, a service may become *unavailable*, which occurs when one or more users cannot access the service. However, many services utilize fault-tolerance techniques, and thus do not fail entirely, but instead experience *performance degradation*. This is when there is an increase in the latency of the service, or in the error rates of requests made to the service. A service may fail in a single region, or in multiple regions. In the case of AWS, a failure may be restricted to a single availability zone inside of a region. Due to the interdependence of services, the failure of one service can cause failures of upstream services, resulting in a *multi-service* failure. A failure can have a varying *area of effect*, which we define as the affected users and the geographical range of the failure (for example, some users in a single region). We define the *impact level* of an outage as the number of affected services, and the geographical range of the outage (for example, one service in a single region). The *duration* of a failure is the period of time between the start of a reported failure and the end of the failure.

3 Methodology

3.1 Data extraction & processing

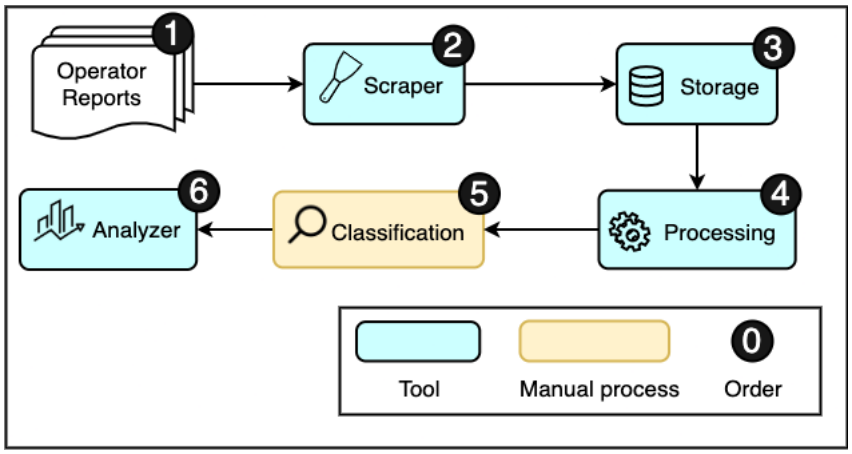

Figure 1: Data collection, extraction, and analysis process.

Cloud providers self-report information about service availability and failures through public status pages. We selected three of the largest worldwide cloud service providers: Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP). The whole workflow is shown in Figure 1. We use data scraped from the providers' public status pages for self-reported failures [14]–[16] in the span of one year, Jan-Dec 2018. The pages were scraped every six hours, to avoid burdening the page host with more frequent scrapes and potentially incurring penalties (step 2 in Figure 1). The raw dataset is approximately 522 KiB in size; the cleaned and processed dataset is around 172 KiB.

We use Python 3.7.7, Pandas 1.0.3, NumPy 1.18.4, SciPy 1.4.1, and Pytz 2019.3 to process and analyze the extracted structured failure data. We first deduplicate the data based on the service name, failure location, and event start time, selecting the events with the longest duration in cases where the other properties are equal (step 4 in Figure 1). This removes 400 events in total: 137 from Azure, 127 from AWS, and 136 from GCP. We convert all reported event start

times to the timezones appropriate to the region specified for the outage. We calculate the event duration in minutes based on the event start and end times. We also compute the year the event took place, as well as the hour of the week (with hour 0 denoting midnight on Monday). We then filter the data, selecting events that occurred in the 2018 calendar year. This yields a grand total of 411 events: 139 from GCP, 144 from AWS, and 128 from Azure.

3.2 Failure classification

Service providers include in their reports a textual description of the outage. The description does not follow a specific format or standard, and must thus be manually analyzed for each event. We classify outages across several categories based on the description of the outage (step 5 in Figure 1). The classification is partially based on that done by Gunawi et al. [17], [18], but as many of the failure descriptions are terse and brief, we cannot capture as much detail in our classification.

The description provides information about the qualitative aspects of the outage. We identify six such aspects: how many services were affected, the severity, the range, the users affected in the outage, the root cause of the outage, the duration of the outage. An outage can affect one or multiple services; we assume that only one service is affected (namely, the reported service), unless the provider explicitly states otherwise in the description. The severity can be strictly a visual error (i.e. performance is not affected, only visual feedback is incorrect), service performance degradation, or complete service unavailability. The range of an outage can be limited to (in ascending order of geographical size) a single availability zone, a single region, or multiple regions. As the range of a single availability zone only applies to AWS services, outages that occurred in a single availability zone (13 events) were merged with those occurring in a single region, to simplify analysis. An outage can affect some users, or all users; this is understood in combination with the range of the outage (e.g. an outage may affect all users in a single region). An event is assumed to affect all users of the service indiscriminately, unless the provider explicitly reports that only some users are affected. The cause of an outage can be a code error, a side effect of maintenance, a configuration error, a network error, an external factor, increased load, or an unhealthy unit. A "unit" is not necessarily a physical (i.e. hardware) unit, but can be a virtual node, a cluster, etc.

Some of these causes can be further separated into narrower categories. A configuration error can stem from a direct change to a configuration file, or can happen as part of a deployment task. A network error can indicate an internal API issue, or an internal network issue. An external factor can be the environmental conditions in the datacenter (e.g. higher temperatures or humidity), a shock event (e.g. a thunderstorm), or an issue caused by a third party.

If one of the features is not stated in the description of a particular outage, we note that information about it was not provided for the outage.

4 Results & analysis

Hourly and daily failure trends We first observe the distribution of outages across the week, this is shown in Figure 2. GCP and AWS both display significant peaks at two points during the week: around the middle (Tuesday and Wednesday), and at the start of the weekend (Friday and Saturday). AWS also shows an increase in outages in the afternoon/evening hours on Sunday. The data provided by Azure does not indicate any clear trends.

Root causes of outages by impact level and vendor We next analyse the distribution of outages across root causes and impact levels. We identify seven classes of root causes: UNIT (individual nodes, instances, or clusters, not necessarily hardware), NETW (related to the internal or external network), MAINT (side effects caused by maintenance), LOAD (increased load on the

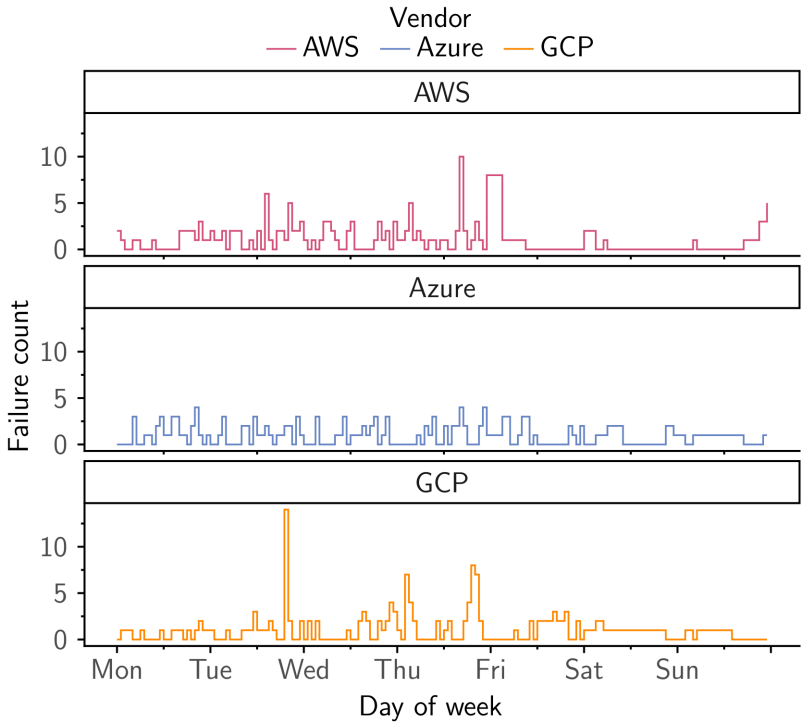

Figure 2: Distribution of outages across the week, separated by vendor.

service), EXTERN (external causes, i.e. environmental or third-party), CODE (code errors/bugs), and CFG (configuration errors). We further separate the outages by vendor, indicated by the color of the bar in Figure 3. We do not include AWS, as we do not have sufficient data regarding root causes from AWS (a cause was only specified for 2.1% of AWS events). Excluded from the plot are outages that did not provide a root cause (a total of 122 events, 45.69%), a range (5 events, 1.87%), or the number of affected services (4 events, 1.5%).

The majority of outages across all levels of impact are caused by configuration errors. For multi-regional outages, the configuration category accounts for the majority by a wide margin. On the other hand, for single-region outages, there are multiple leading causes: apart from configuration errors, outages affecting one service are also caused by increased load and failing instances, and those affecting more than one service are also caused by network errors and maintenance side effects.

Failure distribution across the week, separated by root cause We analyze the frequency of root causes at various points in the week, separated by vendor. We exclude AWS outages, due to a lack of specified root causes. Outages from other providers that did not state a root cause are also excluded (262 events, 63.75%). From Figure 4, it seems that load-related outages tend to happen starting between midday on Wednesday and Thursday evening, with smaller peaks during the night on other days. Most outages caused by misconfiguration happen in the first half of the week, from mid-Monday until midnight on Thursday. This could perhaps be because large code changes are usually introduced during the first few days of the week. The major peaks for these outages generally occur around midnight. The reason for this could be that deployments happen during the night, when it is likely that fewer customers will be using the service; as Langford et al. found, there is a significant difference in traffic during the day and during the night [19]. It could also be that changes are deployed during the day, but there is a delay before bugs appear. Outages related to other root causes generally occur during the weekdays, and there are only a few peaks during the weekend. It is important to note here that this is a relatively small dataset, and the trends observed in Figure 4 could change if more data becomes available.

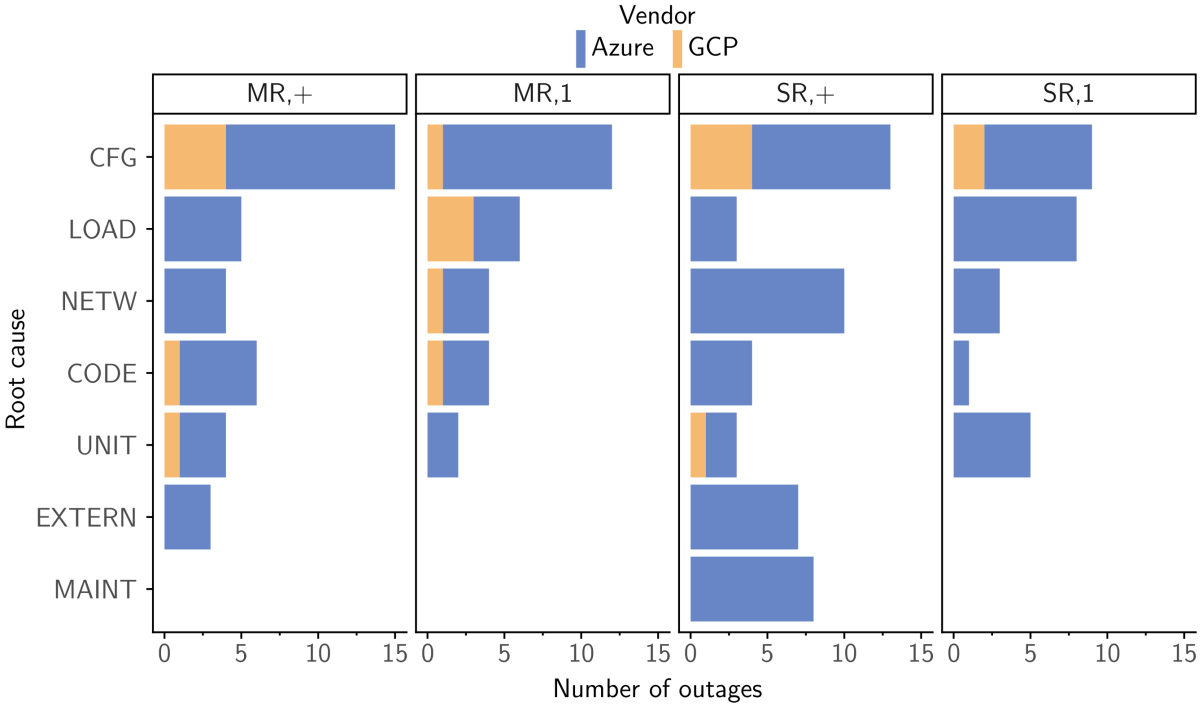

Figure 3: Root causes of outages across varying impact levels, separated by vendor. (MR = multiple regions, SR = single region, 1 = one service, + = multiple services)

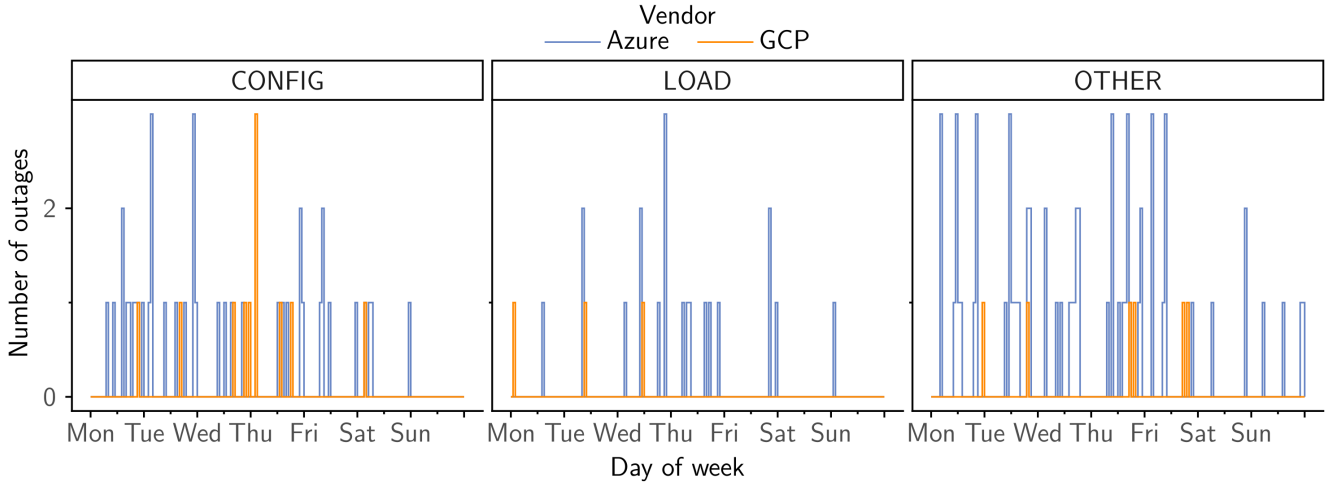

Figure 4: Outages across the week, separated by root cause and vendor. (CONFIG = configuration error, LOAD = increased load, OTHER = other)

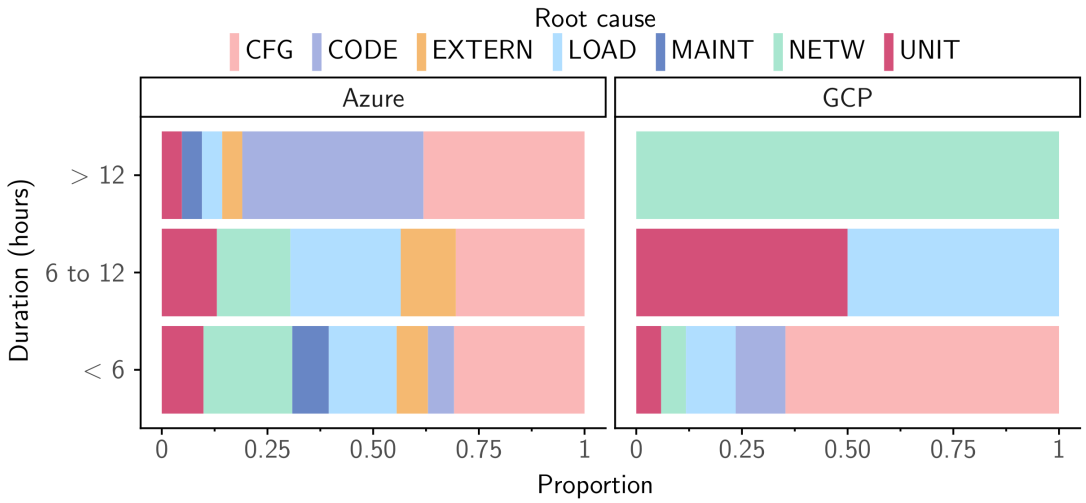

Figure 5: Root causes of outages, separated by duration and vendor. (CFG = configuration error, CODE = code error, EXTERN = external, LOAD = increased load, MAINT = maintenance, NETW = network, UNIT = failing unit (physical/virtual))

Root causes of outages, separated by duration and vendor We separate the outages by duration, in three categories: short (outages lasting less than six hours, '< 6'), medium (six to twelve hours, '6 to 12'), and long (more than twelve hours, '> 12'). The outages are grouped by duration and vendor, such that the horizontal axis shows the proportion of outages for a specific duration category and vendor. We do not plot outages that did not specify a root cause (262 events, 63.75%). All AWS outages that specified a cause were shorter than six hours, and were caused by external events (e.g. third party or environment). However, as discussed above, only three AWS outages specified a cause, so AWS is not included in Figure 5. Azure services have more diverse outage causes than GCP services. All outages of GCP services that lasted longer than twelve hours were caused by network errors; in contrast, network errors did not play a role in these types of outages for Azure services. The longest outages of Azure services were caused in approximately equal parts by code and configuration errors. Configuration errors were a common cause of Azure outages for all three duration categories, while only the shortest GCP outages (shorter than six hours) had configuration errors as a cause. For both GCP and Azure, increased load was a main factor in the short- and medium-duration outages. For GCP, there were more medium-duration outages caused by increased load than short-duration outages; for Azure services, the proportion is approximately the same. GCP also had failing computational units as a major cause for medium-duration outages; this was not as prominent for Azure services, where failing units accounted for approximately the same proportion of short-duration outages as for medium-duration outages.

Root causes of outages, separated by area of effect We also analyze the root causes of outages, separated by the area of effect. Here we exclude those events that did not provide a cause (262 events, 63.75%), a range (5 events, 1.22%), or the affected users (1 events, 0.24%). From Figure 6, we conclude that configuration errors account for the majority of outages with the widest area of effect (all users in multiple regions). The second main cause of such outages are code errors, which interestingly do not play a major role in single-region outages, and do not appear as a cause of single-region outages affecting all users. Failing instances are mainly a cause of single-region outages, more so for outages that affect all users – this is probably due to the fact that individual instances, nodes, or clusters are localised in a single region [20]. From the available data, it seems that outages caused as a maintenance side effect only affect some users of

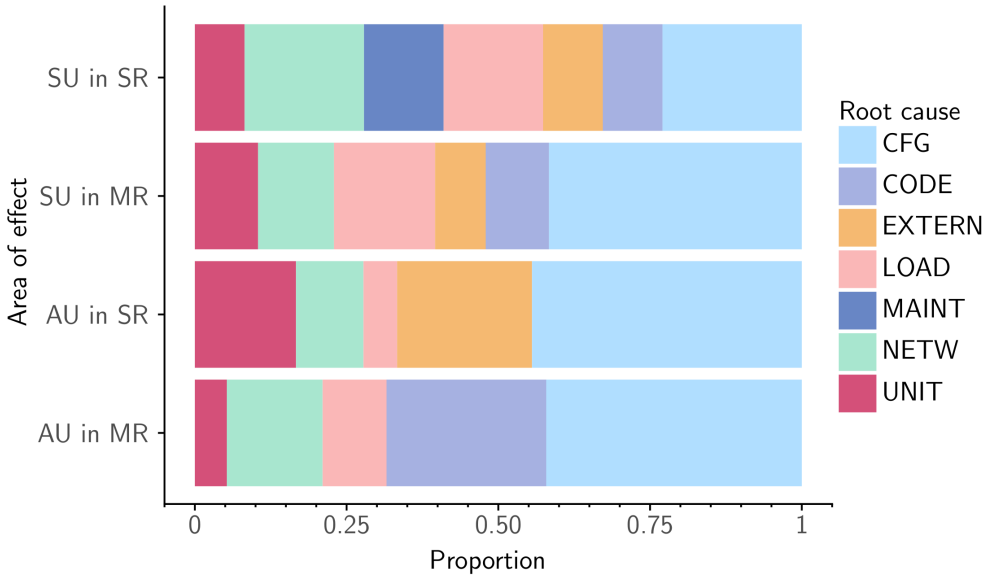

Figure 6: Root causes of outages at various areas of effect. (SU = some users, AU = all users, SR = single region, MR = multiple regions)

the service, mostly in a single region. It also appears that outages caused by increased load more commonly affect only some users in a given range. The majority of outages caused by network issues affect some users in a single region, though they also play a somewhat significant role in the other area of effect categories (accounting for around 10% of the outages in each category).

Severity and duration of failures by the area of effect Next, we consider the severity and duration of an outage, depending on its area of effect. We exclude events that did not state the affected users (1 events, 0.24%), range (5 events, 1.22%), severity (41 events, 9.98%), or duration (41 events, 9.98%). The first immediate finding from Figure 7 is that the majority of outages result in one or more services becoming intermittently unavailable. For outages affecting all users, in a single region or in multiple regions, there is a higher proportion of outages that cause a service to be intermittently degraded or unavailable than for outages affecting some users. Furthermore, the majority of outages that resulted in a service being continuously unavailable (for some period of time) affected all users in multiple regions. When the performance of one or more services is degraded, it generally happens for a continuous period of time.

Outages separated by area of effect and vendor We analyse the distribution of outages per vendor, in figure Figure 8. We do not include events that are missing information about the users (1 events, 0.24%) or the area of effect (5 events, 1.22%). We observe that the range of the majority of outages depends on the vendor. The majority of GCP outages affect all users in multiple regions, while the majority of AWS outages affect all users in a single region. For Azure services, the outages mostly affect some users, with an approximately equal distribution between a single region and multiple regions.

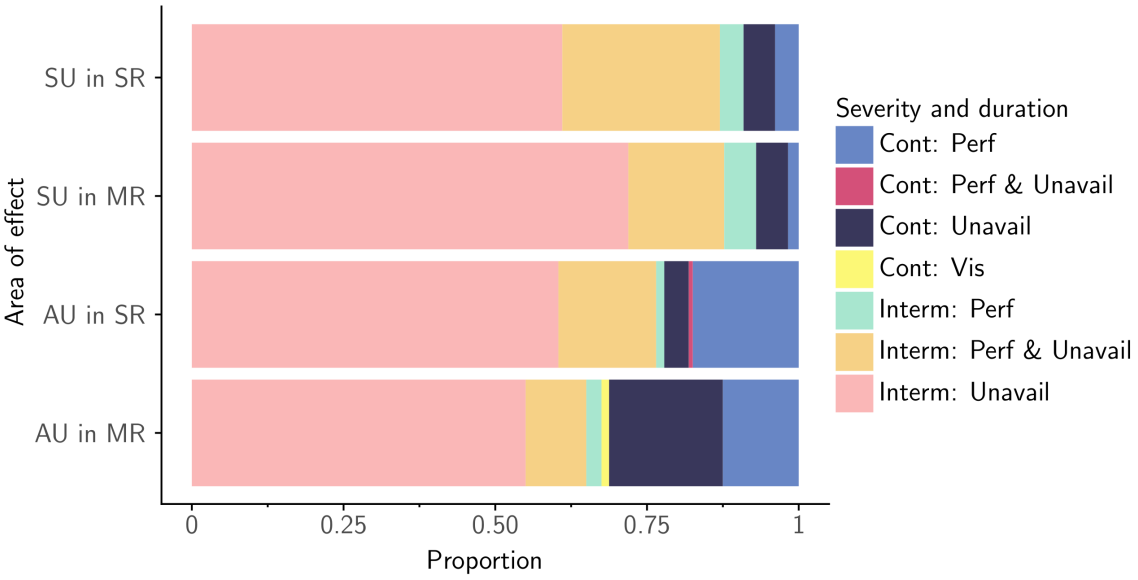

Figure 7: Severity of an outage and the users affected by the outage. (SU = some users, AU = all users, Cont = continuous, Interm = intermittent, Perf = performance degradation, Unavail = unavailable, Vis = visual)

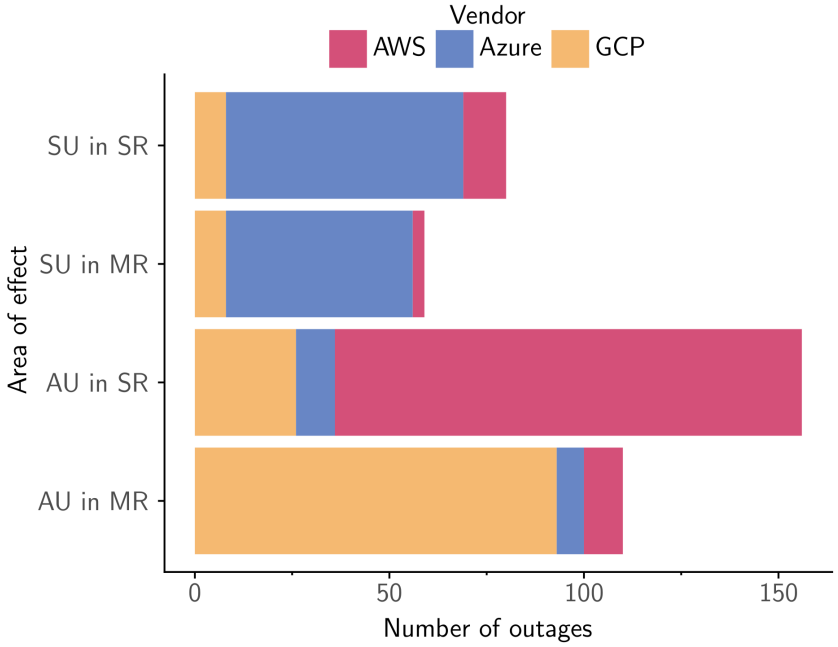

Figure 8: Area of effect of outages by vendor. (AU = all users, SU = some users, SR = single region, MR = multiple regions)

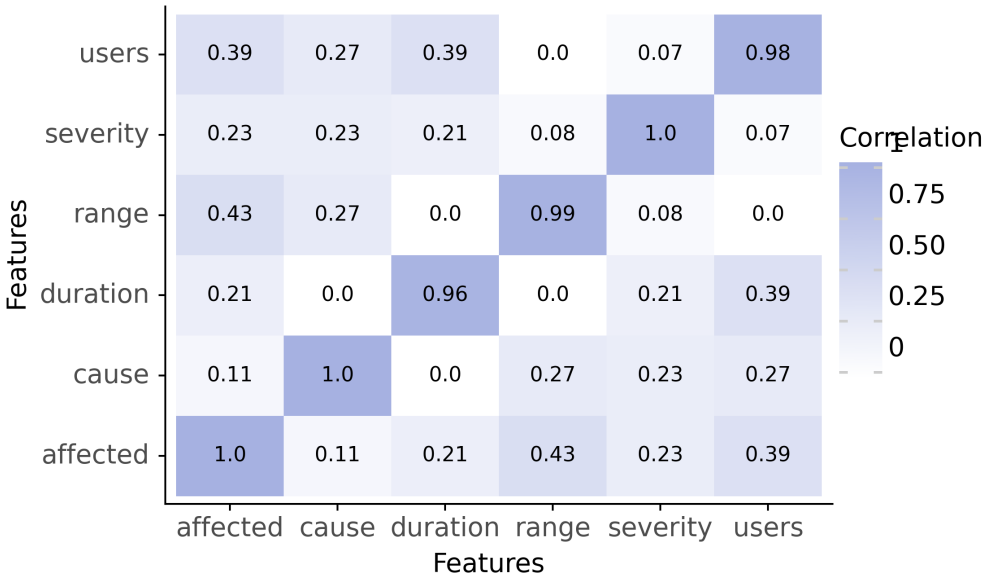

Figure 9: Correlation matrix (Cramér's V)

Correlation analysis (Cramér's V) Finally, we conduct correlation analysis on the various features of the data. The first statistic we consider is Cramér's V (also known as Cramér's phi coefficient, written as ϕ_c), which measures association between two variables. It extends the phi coefficient to contingency tables larger than 2×2, and results in a value between 0 and 1, with 0 indicating no correlation. Cramér's V is computed as:

$$V = \sqrt{\frac{\chi^2}{n \times \min(k-1, r-1)}}$$

where r is the number of rows and k the number of columns in the contingency table, n is the number of observations, and χ^2 is derived from Pearson's chi-squared test [21].

Applying the measure to the dataset, we obtain a correlation matrix shown in Figure 9. We observe that there is no strong correlation between any of the features. Relative to the other values in the matrix, the component affected in the outage seems to be moderately correlated with the range of the outage (0.5), and with the users affected in the outage (0.49). There may also be a slight correlation between the cause of the outage and the range of the outage (0.38), as well as with the users affected in the outage (0.31). However, since the values are low, these results are inconclusive.

Association analysis with the uncertainty coefficient (Theil's U) We calculate uncertainty coefficients, also known as Theil's U, for the features of the data; this is shown in a matrix in Figure 10. The uncertainty coefficient of y with respect to x, written as $U(y|x)$, shows how much information x provides about y, and is a value between 0 and 1. A value of 0 indicates that x gives no information about y, and a value of 1 that knowledge of x completely predicts y. Computing Theil's U clarifies the associations seen from Cramér's V: the symmetry of Cramér's V does not necessarily hold for the actual correlations. The uncertainty coefficient provides more information about the true relations between the different features [22].

$U(y|x)$ is computed as:

$$U(y|x) = \frac{H(y) - H(y|x)}{H(y)}$$

where H is entropy of a single distribution, and $H(y|x)$ is the entropy of y conditional on x [23].

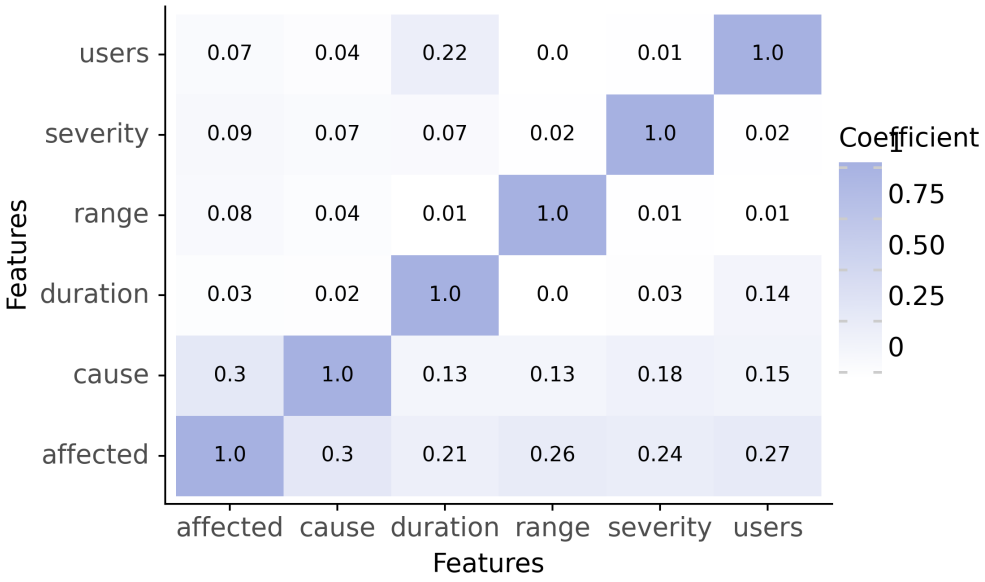

Figure 10: Uncertainty coefficient matrix (Theil's U)

The results in Figure 10 show no clear association, as the coefficients are low. However, the users affected in the outage could determine the affected component (0.34) and range of the outage (0.3). This seems to support the findings from Figure 9. Similarly, the affected component could determine the cause of the outage (0.28).

5 Threats to Validity

Having presented the results and analysis, this section discusses the main limitations of the study.

1. Manual classification of data points. Due to the lack of structure or standard format among the failure descriptions, all events had to be classified manually across different categories (step 5 in Figure 1). Though the classification was conducted diligently and checked multiple times, such a process is vulnerable to human error. Furthermore, by relying on our interpretation of the descriptions for classification, we introduce an element of subjectivity in the dataset. It is possible that some events are misclassified, which would negatively affect the validity of our results. Unfortunately, because of the aforementioned lack of structure in the reports and insufficient tools for analysis of such data, this is currently unavoidable.

2. Lack of available data. In our analysis, we use a dataset with a relatively small amount of events. To draw more significant and valid conclusions, a much larger dataset would be needed. As no such public repository of data is currently available, we do not have a way to obtain these data. Moreover, for some analyses, we exclude a few data points due to insufficient information. Such selective cleaning may lead to unintended consequences or biases in our results. Finally, as information about failures could be damaging to cloud providers, the self-reporting process could result in bias in the data itself.

3. Heuristic processing errors. For AWS and Azure failures, we extract the event start and end time from textual descriptions. We use heuristic methods based on sentence structures. Despite taking the utmost care, there is a non-zero possibility that we might have missed or wrongly attributed certain failures.

6 Related work

Having presented and discussed our results, we examine previous work done in the field.

One similar study is that of Gunawi et al. [17], who analyze the availability of popular cloud services during a time period of seven years. They examine services for online chat (e.g. WhatsApp), e-commerce (e.g. Amazon.com), email (e.g. GMail), SaaS (e.g. Salesforce), video streaming (e.g. Netflix), and others. They collect data from news headlines, and from the providers' public post-mortem reports (published after the failure event). Similarly to our study, they manually tag the outages with metadata. This metadata describes the root causes, impacts, fixes, downtime, type (planned or unplanned), and service scope of each outage. Due to the nature of their dataset, they are able to extract more detail in some metadata categories, such as root causes. They find that almost 50% of the services they analyze experience an average of three or more outages per year, and 25 services do not reach 99.9% uptime. These outages do not decrease as the service matures; rather, as a service matures, it becomes more popular, and thus need to handle more users and increases in complexity. Perhaps complementary to our findings regarding root causes of outages, fixing software is a major fix procedure category in their results, accounting for 22% of all outages. Their analysis of root causes shows that to ensure that cloud services do not have a single point of failure, perfection is required along the whole failure recovery chain. They also conduct a detailed analysis of root causes of outages.

Another related study focuses on popular cloud systems, particularly on those generally not seen by end users [18]. These include Hadoop MapReduce, Hadoop File System, HBase, Cassandra, ZooKeeper, and Flume. The aim of the study is to characterise the bugs present in cloud systems. Gunawi et al. collect data from issue repositories maintained by the development projects of the aforementioned systems. The issues are generally submitted by the developer or user community of the system. The authors select issues submitted over a period of three years, and analyze patches and developer responses for the issues. They conduct a manual classification of the issues for each service, applying metadata labels related to the aspect of the issue (reliability, performance, availability, etc.), hardware type, failure mode (stop, corrupt, or "limp"), and bug scope (single machine, multiple machines, or a whole cluster). They find that 8% of bugs are unique to cloud systems, and that even though there are software measures in place, 13% of issues are still caused by hardware failures. Their results also suggest that cascades of failures can occur in subtle ways from a "killer bug", which is perhaps similar to how failures can cascade across services in our analysis. Finally, the largest category of software bugs in their results consists of those that are logic-specific, which complements our findings.

There have also been other studies of failures in the cloud, with various aims. Some have focused on failure analysis in the area of high-performance computing (HPC). The majority of high-performance computing studies use data from providers, and conduct an analysis of large-scale production systems such as the Google Cloud cluster or the Los Alamos National Lab, or supercomputers such as the Titan or Eos [24]–[36]. The data used in these studies most commonly comes from traces and reliability-availability-stability (RAS) logs, with some studies using job logs and system logs; one study conducts a combined analysis of RAS logs and job logs [27]. There is also one analysis of HPC system failures with user-reported data [37]. Next, perhaps closer in subject to our work, there have been a number of studies analyzing provider data for failures in large-scale services, such as storage systems and cloud platforms [35], [36], [38]–[43]. Those that study more customer-facing services such as Apache Cassandra, Amazon S3, or MySQL generally utilize user-reported data [44]–[50]. Compared to these studies, our work contributes an analysis of customer-facing services using data from providers; in particular, the data they report publicly, as opposed to data from internal error tracking databases or reports. Some studies synthesize data from both providers and customers [51], [52]. Finally, other analyses investigated network failures using provider data [53]–[55], and virtual/physical machine failures using either provider data [56]–[58] or a combination of provider and user data [59].

7 Conclusion

Cloud services run many of the modern applications we use today, and are expected to be constantly available. This is not always the case, as services outages happen relatively frequently. When a service goes down, other services that depend on it may also fail. Therefore, prevention is key, and understanding how these outages happen is the first step towards preventing them. In this paper, we have conducted a study of public cloud provider reports of outages during the 2018 calendar year. We developed a framework for classifying and analyzing these reports, and used it to examine the collected data. We identified the challenges associated with the gathering and analysis of failure data, and extracted patterns relating to the failure of cloud services. Some of the data we generated and analyzed in this study has been used for a section of a paper submitted to a top conference.

Future work in this area could address some of the limitations and concerns discussed in section 5. In particular, the classification process could be improved, perhaps via language processing algorithms. This would eliminate the need to classify events manually, reducing the probability of erroneous classification. Furthermore, data limitations could be addressed by conducting a synthesis of all data from various sources, such as news reports, provider reports, and user reports. Data from multiple sources for a particular outage could be correlated using timestamps, or other identifying features. This would provide a more objective view of the data, and would result in a much larger dataset.

References

[1] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, pp. 50–58, Apr. 2010. DOI: 10.1145/1721654.1721672.

[2] J. Dean, "The rise of cloud computing systems," in *SOSP History Day 2015*, ser. SOSP '15, Monterey, California: Association for Computing Machinery, 2015, ISBN: 9781450340175. DOI: 10.1145/2830903.2830913. [Online]. Available: https://doi.org/10.1145/2830903.2830913.

[3] M. Steen and A. S. Tanenbaum, "A brief introduction to distributed systems," *Computing*, vol. 98, no. 10, pp. 967–1009, Oct. 2016, ISSN: 0010-485X. DOI: 10.1007/s00607-016-0508-7. [Online]. Available: https://doi.org/10.1007/s00607-016-0508-7.

[4] KATC News, *Lafayette 911 moving to cloud-based dispatch system*, https://katc.com/news/around-acadiana/st-martin-parish/2019/03/19/lafayette-911-moving-to-cloud-based-dispatch-system/, Accessed: 2020-06-01, Mar. 2019.

[5] A. Mazmanian and F. Konkel, *Cloud failure temporarily crashes healthcare.gov*, https://fcw.com/articles/2013/10/28/cloud-failure-crashes-healthcare-gov.aspx, Accessed: 2020-06-01, Oct. 2013.

[6] Genaro Network, *Tencent was claimed ten million for data loss due to cloud hard drive glitch*, https://medium.com/genaro-network/tencent-was-claimed-ten-million-for-data-loss-due-to-cloud-hard-drive-glitch-344a26449fe2, Accessed: 2020-06-01, Aug. 2018.

[7] N. Hunt, *Neil hunt of netflix discusses how aws supports deployment of new features and tools*, https://www.youtube.com/watch?v=SorHbAiZ918, Accessed: 2020-06-11, Dec. 2016.

[8] Z. Whittaker, *Amazon web services suffers outage, takes down vine, instagram, others with it*, https://www.zdnet.com/article/amazon-web-services-suffers-outage-takes-down-vine-instagram-others-with-it/, Accessed: 2020-06-11, Aug. 2013.

[9] *Microsoft to refund windows azure customers hit by 12 hour outage that disrupted xbox live*, https://techcrunch.com/2013/02/24/microsoft-to-refund-windows-azure-customers-hit-by-12-hour-outage-that-disrupted-xbox-live/, Accessed: 2020-06-11, Feb. 2013.

[10] *Microsoft azure and xbox live services experiencing outages*, https://gadgets.ndtv.com/internet/news/microsoft-azure-and-xbox-live-services-experiencing-outages-622865, Accessed: 2020-06-11, Nov. 2014.

[11] L. Dignan, *Top cloud providers in 2020: AWS, Microsoft Azure, and Google Cloud, hybrid, SaaS players*, https://www.zdnet.com/article/the-top-cloud-providers-of-2020-aws-microsoft-azure-google-cloud-hybrid-saas/, Accessed: 2020-06-09, May 2020.

[12] E. Jones, *Cloud market share – a look at the cloud ecosystem in 2020*, https://kinsta.com/blog/cloud-market-share/, Accessed: 2020-06-09, May 2020.

[13] *AWS Elastic Beanstalk FAQs*, https://aws.amazon.com/elasticbeanstalk/faqs/, Accessed: 2020-06-09.

[14] AWS, *AWS status JSON feed*, Accessed: 2020-05-18. [Online]. Available: http://status.aws.amazon.com/data.json.

[15] Google Cloud Platform, *Google Cloud incidents JSON feed*, Accessed: 2020-05-18. [Online]. Available: https://status.cloud.google.com/incidents.json.

[16] Microsoft Azure, *Azure status history*, Accessed: 2020-05-18. [Online]. Available: https://status.azure.com/en-us/status/history/.

[17] H. S. Gunawi, M. Hao, R. O. Suminto, A. Laksono, A. D. Satria, J. Adityatama, and K. J. Eliazar, "Why does the cloud stop computing?: Lessons from hundreds of service outages," in *Proceedings of the Seventh ACM Symposium on Cloud Computing - SoCC '16*, Santa Clara, CA, USA: ACM Press, 2016, pp. 1–16, ISBN: 978-1-4503-4525-5. DOI: 10.1145/2987550.2987583. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2987550.2987583 (visited on 03/02/2020).

[18] H. S. Gunawi, V. Martin, A. D. Satria, M. Hao, T. Leesatapornwongsa, T. Patana-anake, T. Do, J. Adityatama, K. J. Eliazar, A. Laksono, and J. F. Lukman, "What bugs live in the cloud?: A study of 3000+ issues in cloud systems," in *Proceedings of the ACM Symposium on Cloud Computing - SOCC '14*, Seattle, WA, USA: ACM Press, 2014, pp. 1–14, ISBN: 978-1-4503-3252-1. DOI: 10.1145/2670979.2670986. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2670979.2670986 (visited on 03/02/2020).

[19] J. Langford, L. Li, R. P. McAfee, and K. Papineni, "Cloud control: Voluntary admission control for intranet traffic management," *Information Systems and e-Business Management*, vol. 10, pp. 295–308, 2012.

[20] *Choose an AWS region*, https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-plan-region.html, 2020.

[21] E. W. Holmes, "Handbook of Parametric and Nonparametric Statistical Procedures," *Clinical Chemistry*, vol. 44, no. 11, pp. 2384–2384, Nov. 1998, ISSN: 0009-9147. DOI: 10.1093/clinchem/44.11.2384. eprint: https://academic.oup.com/clinchem/article-pdf/44/11/2384/32727301/clinchem2384.pdf.

[22] S. Zychlinski, *The search for categorical correlation*, https://towardsdatascience.com/the-search-for-categorical-correlation-a1cf7f1888c9, Feb. 2018.

[23] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed. Cambridge University Press, 2007, ISBN: 9780521880688.

[24] X. Chen, C.-D. Lu, and K. Pattabiraman, "Failure analysis of jobs in compute clouds: A google cluster case study," *2014 IEEE 25th International Symposium on Software Reliability Engineering*, pp. 167–177, 2014.

[25] N. El-Sayed, H. Zhu, and B. Schroeder, "Learning from failure across multiple clusters: A trace-driven approach to understanding, predicting, and mitigating job terminations," *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1333–1344, 2017.

[26] Y. Liang, Y. Zhang, A. Sivasubramaniam, M. Jette, and R. K. Sahoo, "Bluegene/l failure analysis and prediction models," *International Conference on Dependable Systems and Networks (DSN'06)*, pp. 425–434, 2006.

[27] Z. Zheng, L. Yu, W. Tang, Z. Lan, R. Gupta, N. Desai, S. Coghlan, and D. Buettner, "Co-analysis of ras log and job log on blue gene/p," *2011 IEEE International Parallel & Distributed Processing Symposium*, pp. 840–851, 2011.

[28] S. Kavulya, J. Tan, R. Gandhi, and P. Narasimhan, "An analysis of traces from a production mapreduce cluster," *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 94–103, 2010.

[29] S. Gupta, D. Tiwari, C. Jantzi, J. H. Rogers, and D. Maxwell, "Understanding and exploiting spatial properties of system failures on extreme-scale hpc systems," *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 37–44, 2015.

[30] S. Gupta, T. Patel, C. Engelmann, and D. Tiwari, "Failures in large scale systems: Long-term measurement, analysis, and implications," *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2017.

[31] S. Di, H. Guo, E. Pershey, M. Snir, and F. Cappello, "Characterizing and understanding hpc job failures over the 2k-day life of ibm bluegene/q system," *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 473–484, 2019.

[32] N. El-Sayed and B. Schroeder, "Reading between the lines of failure logs: Understanding how hpc systems fail," *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 1–12, 2013.

[33] C. D. Martino, Z. T. Kalbarczyk, R. K. Iyer, F. Baccanico, J. Fullop, and W. Kramer, "Lessons learned from the analysis of system failures at petascale: The case of blue waters," *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 610–621, 2014.

[34] B. Schroeder and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," *IEEE Trans. Dependable Secur. Comput.*, vol. 7, pp. 337–351, 2010.

[35] B. Javadi, D. Kondo, A. Iosup, and D. H. J. Epema, "The failure trace archive: Enabling the comparison of failure measurements and models of distributed systems," *J. Parallel Distributed Comput.*, vol. 73, pp. 1208–1223, 2013.

[36] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an mttf of 1, 000, 000 hours mean to you?" In *FAST*, 2007.

[37] J. Gray, "Why do computers stop and what can be done about it?" In *Symposium on Reliability in Distributed Software and Database Systems*, 1986.

[38] D. L. Oppenheimer, A. Ganapathi, and D. A. Patterson, "Why do internet services fail, and what can be done about it?" In *USENIX Symposium on Internet Technologies and Systems*, 2003.

[39] D. Ford, F. Labelle, F. I. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in globally distributed storage systems," in *OSDI*, 2010.

[40] P. Garraghan, P. Townend, and J. Xu, "An empirical failure-analysis of a large-scale cloud computing environment," *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering*, pp. 113–120, 2014.

[41] P. Yalagandula and S. Nath, "Beyond availability: Towards a deeper understanding of machine failure characteristics in large distributed systems," in *WORLDS*, 2004.

[42] S. Li, H. Zhou, H. Lin, T. Xiao, H. Lin, W. Lin, and T. Xie, "A characteristic study on failures of production distributed data-parallel programs," *2013 35th International Conference on Software Engineering (ICSE)*, pp. 963–972, 2013.

[43] H. Zhou, J.-G. Lou, H. Zhang, H. Lin, H. Lin, and T. Qin, "An empirical study on quality issues of production big data platform," *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, vol. 2, pp. 17–26, 2015.

[44] F. Frattini, R. Ghosh, M. Cinque, A. Rindos, and K. S. Trivedi, "Analysis of bugs in apache virtual computing lab," *2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 1–6, 2013.

[45] D. Yuan, Y. Luo, X. Zhuang, G. R. Rodrigues, X. Zhao, Y. Zhang, P. Jain, and M. Stumm, "Simple testing can prevent most critical failures: An analysis of production failures in distributed data-intensive systems," *login Usenix Mag.*, vol. 40, 2014.

[46] M. Palankar, A. Iamnitchi, M. Ripeanu, and S. L. Garfinkel, "Amazon s3 for science grids: A viable solution?" In *DADC '08*, 2008.

[47] P. Fonseca, C. Li, V. Singhal, and R. Rodrigues, "A study of the internal and external effects of concurrency bugs," *2010 IEEE/IFIP International Conference on Dependable Systems & Networks (DSN)*, pp. 221–230, 2010.

[48] T. Benson, S. Sahu, A. Akella, and A. Shaikh, "A first look at problems in the cloud," in *HotCloud*, 2010.

[49] W. Jiang, C. Hu, Y. Zhou, and A. Kanevsky, "Are disks the dominant contributor for storage failures?: A comprehensive study of storage subsystem failure characteristics," *TOS*, vol. 4, 7:1–7:25, 2008.

[50] Z. Yin, X. Ma, J. Zheng, Y. Zhou, L. N. Bairavasundaram, and S. Pasupathy, "An empirical study on configuration errors in commercial and open source systems," in *SOSP '11*, 2011.

[51] S. Ostermann, A. Iosup, N. Yigitbasi, R. Prodan, T. Fahringer, and D. H. J. Epema, "An early performance analysis of cloud computing services for scientific computing," 2008.

[52] S. K. Sahoo, J. Criswell, and V. S. Adve, "An empirical study of reported bugs in server software with implications for automated bug diagnosis," *2010 ACM/IEEE 32nd International Conference on Software Engineering*, vol. 1, pp. 485–494, 2010.

[53] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: Measurement, analysis, and implications," in *SIGCOMM*, 2011.

[54] R. Banerjee, A. Razaghpanah, L. Chiang, A. Mishra, V. Sekar, Y. Choi, and P. Gill, "Internet outages, the eyewitness accounts: Analysis of the outages mailing list," in *PAM*, 2015.

[55] D. Turner, K. Levchenko, A. C. Snoeren, and S. Savage, "California fault lines: Understanding the causes and impact of network failures," in *SIGCOMM 2010*, 2010.

[56] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in *SoCC '10*, 2010.

[57] E. B. Nightingale, J. R. Douceur, and V. Orgovan, "Cycles, cells and platters: An empirical analysisof hardware failures on a million consumer pcs," in *EuroSys '11*, 2011.

[58] A. Rosà, L. Y. Chen, and W. Binder, "Understanding the dark side of big data clusters: An analysis beyond failures," *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 207–218, 2015.

[59] R. Birke, I. Giurgiu, L. Y. Chen, D. Wiesmann, and A. P. J. Engbersen, "Failure analysis of virtual and physical machines: Patterns, causes and characteristics," *2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 1–12, 2014.

A Problem statement

The main question we try to answer is: how do cloud services fail? To this end, we use provider-reported data from public status pages. Finding an answer to this question is important, because knowing how services fail can pave the way towards partial or total prevention of such failures. This would allow cloud providers to come closer to their goal of constant availability, and would make cloud services more dependable.

Our results show only a part of the picture, and further research is necessary to clarify our observations, as well as those of other research in the area. One related question is: how can we create a more complete view of cloud service failures? For example, how could a synthesis of the various data sources used in the work mentioned in section 6 and the data source used in this study help with failure analysis? Or, would it perhaps be useful to create a repository similar to the Failure Trace Archive [35] for service failure data, and how could such a repository be implemented? More questions could be asked about the analysis process: how can we improve the accuracy of failure classifications based on textual descriptions? What other classification frameworks would allow us to extract the most information from failure reports? These and other questions can be explored in future work.

B Self-reflection

This was the first opportunity I had to conduct a 'proper' research project, and as such benefited me greatly. I learned how to clean a large dataset, and how to conduct a scientific analysis of that dataset, including data visualisation. I learned how to use software tools such as Pandas, which is an industry standard for data analysis, and Plotnine for data plotting – these skills will be useful in any future projects that use large sets of data. I also learned how to compile the process and results into a research paper, and how to look for and discuss related articles. The possibility of contributing to a paper submitted to a top conference was a great motivator. Overall, the project helped me develop into a more independent researcher.

The largest period of time was spent on the preparation of the dataset; that is, on cleaning and labelling the dataset for analysis. The manual classification of events was especially time-consuming. The second longest amount of time was spent on analysis of data, particularly on selecting the best aspects to analyze and on creating good visualisations of the results. The rest of the time was spent on researching, and on writing the report.