Vrije Universiteit Amsterdam

Universiteit van Amsterdam

Master Thesis

A Statistical Analysis of Cloud Service Failures by Source, Popularity, and Severity

Author: Shane Minnema (2572028)

First Reader:	Prof. Alexandru Iosup	a.iosup@vu.nl
Daily Supervisor:	Sacheendra Talluri	s.talluri@vu.nl
Second Reader:	Dr. Animesh Trivedi	a.trivedi@vu.nl

A thesis submitted in fulfillment of the requirements for the joint UvA-VU Master of Science degree in Computer Science

August 16, 2022

"It is a capital mistake to theorize before one has data"
Sherlock Holmes

Abstract

Cloud services are known for their high availability. Despite this, failures do occur occasionally which may cause damage to a cloud service's reputation or financial earnings. Failures in the cloud are reported in one of two manners: by the cloud service provider themselves or through websites that gather user-reports. In this thesis, we investigate both types of cloud failure data from an archive of many sources and services collected between 2017 and 2020. A justification for the selection of these sources and services is provided along with suggestions for their expansion. We formulate a method for extracting relevant data and combining them into a single unified dataset that categorizes each cloud service failure into one of four types: operational, partial outages, major outages, or maintenance events. As far as we are aware, this is the first study of its kind that categorizes cloud service failures based on failure type. We find that cloud services live overwhelmingly in an operational state. We find that 95.52% of all status reports are operational, 4.36% are partial outages, 0.12% are major outages, and 0.002% are maintenance events. The mean time between failure is 23.59, 9.55, and 22.48 days for partial outages, major outages, and maintenance events. Following the same sequence of failure types, we find the mean time to repair to be 5.41, 6.80, and 2.23 hours. These same metrics are also calculated for each of the cloud services individually. Although we provide new insight into the cloud failure landscape, there is much room for expanding and improving upon our work.

Contents

List of Figures

List of Tables

1

Introduction

Cloud services are highly available, on-demand, and scalable internet applications (1). Examples of popular cloud services include the video streaming service *Netflix*, *Amazon Web Services*, and the online storage platform *Google Drive*, which surpassed one billion users as early as 2018 (2). The societal and economic impact of the cloud is continually increasing; for instance, in 2021 the global public cloud market grew by 29% to $408.6 billion (3) and is estimated to surpass one trillion dollars by 2024 (4). Cloud services are designed to be fault-tolerant, meaning that they should continue to operate even when one or multiple components exhibit a fault or failure. Although cloud services are designed with this fault-tolerance in mind, failures still occur in practice (5, 6). We believe that investigating cloud service failures provides insightful information that leads to a better understanding of how and why cloud services fail. The focus of this master thesis is to provide this insight by analyzing a cloud failures archive consisting of data collected over many years and from multiple sources.

Causes of cloud services failures include hardware problems, software bugs, server upgrades, planned maintenance, networking issues, and more (5). Although failure rates tend to fall below 1% (7), service downtime can still lead to significant revenue loss (8). Some cloud service providers share failure data that can be statistically analyzed, compared, and contrasted with additional sources of information and other cloud services. This leads to a better understanding of the cloud service failure landscape, which may help to mitigate failures in the future (and perhaps, minimize revenue loss).

There are two general sources of cloud service failure information: those provided by cloud service providers themselves, and those that collect user-reported data. Google, Microsoft, and Apple are three examples of cloud service providers that publish their own failure data. Although some cloud service providers publicize their failure data, they may not necessarily report all failures or provide sufficiently detailed information under certain circumstances (9, 10). This is probably due to cloud service providers striving to maintain their position as reliable, top-tier operators. On the other hand, websites exist that allow users to report on cloud service failures (*i.e. crowdsourcing*). Examples of crowdsourcing websites include Downdetector and Outage.Report. Crowdsourced failure data is valuable because it can be used to verify the transparency of failure reports from cloud service providers. Though valuable, user-reported data is not fully reliable as the cloud service status descriptions they provide are usually vague. Nonetheless, user-reported data provides a general indication of a cloud service's health.

In this paper, we analyze a data archive containing status data of 54 cloud services collected from 46 sources. The dataset consists of a collection of HTML documents in various formats. We first inspect a sample of these HTML documents for each cloud and its source to discover the common properties between their status reports. We determine which sources contain data that is suitable for compiling a single unified dataset. The dataset we compile contains information about the *global status* of a cloud service, which is classified as *operational*, *partial outage*, *major outage*, or *maintenance event*. Some sources breakdown status reports into sub-categories, such as geographical region, and in these cases we calculate the proportion of status reports that fall into each of the four global status classifications. This also means that the data does not always lead to a binary classification of the status of a cloud service at any given time. We use the results of the data classification to discover several properties of cloud service failures for each class, namely: the distribution of means, the mean time between failures, and the mean time to repair failures. We also compare failure metrics between self-reported and crowdsourced data when appropriate. This information provides an overview of the cloud service failure landscape. As far as we are aware, this method of comparing different classes of failures for cloud services is the first of its kind.

There are many challenges associated with unifying data from multiple sources and formats. We must first identify commonly reported status information between the sources before extracting them. This status information is reported at different levels of granularity depending on the pairing of a cloud service and its source of status reports. Some sources only report on the global status of a cloud service, while others break down status reports into sub-categories (e.g. by sub-service and/or geographical region). Averaging the occurrences of sub-categorical status data provides an indication of the global status of a cloud service. This aggregation is necessary as the global status is the common denominator between status reports; however, parsing sub-categorical status data requires much manual labor. Other challenges include dealing with status reports collected at different time granularity, and how to handle the presence of significant time gaps between data entries when performing our statistical analysis.

Overall, we find that the mean time between cloud service failures and repairs support the findings of a previous study on public cloud failures (11). Although previous studies classify a failure event with a possible root cause, our study is the first of its kind (that we are aware of) that categorizes cloud service statuses into failure classes. In addition, we present a justification for the selection of cloud services and their sources of status information in the dataset we analyze, since we do not select them ourselves. There is a fair mix of popular versus less-popular cloud services, and we provide suggestions for additional sources and services to include in future work.

1.1 Research Questions

The main research question this thesis attempts to answer is: how do we analyze cloud failure data that has been gathered from multiple sources and exist in widely different formats? We break this main research question down into four sub-questions. The approach to each is elaborated further in section 3.1. Our research sub-questions are as follows:

RQ.1 How do we combine cloud failure data from various sources into a single, uniform, and easy-to-use dataset?

The data archive contains HTML documents with information on the status of 46 unique cloud services collected from multiple sources. Each source reports status information in their own way, and thus needs to first be handled individually before combining into a single uniform dataset. The main challenge of this research question is how to achieve uniformity in the final dataset, for three reasons: **(i)** each source reports data in a unique format, **(ii)** the data across sources report status information differently (e.g. global versus regional statuses), and **(iii)** many cloud service providers breakdown statuses into subcategories, and these are inconsistent between different service providers.

RQ.2 What is a good process for selecting cloud services and their sources of status information and how does the current selection compare to this process?

The data archive contains a collection of status reports from many cloud service providers. Currently, there is no documented justification or reasoning why information for these specific cloud services or sources of user reports were selected. It is important that we determine the popularity, or ranking, for both the cloud services and the sources of user reports present in the dataset. In achieving this, we provide evidence that the selection is at least relevant to today's cloud-based culture. We also identify additional sources and/or cloud services that are relevant for future work.

RQ.3 What are the statistical properties of cloud service failures in relation to their type, the mean time between failures, and the mean time to repair them?
We compile a single uniform dataset where certain statistical properties are derived pertaining to the status information of cloud services. We find that statuses fall into one of four categories: *operational*, *partial outages*, *major outages*, and *maintenance events*. We use the Empirical Cumulative Distribution Function (ECDF) to determine the distribution of means for each of these categories. Additionally, calculating the mean time between failure events provides an indication of the availability of each cloud service, while the mean time to repair indicates a cloud service provider's effectiveness at recovering from failure.

RQ.4 How do self-reported sources of cloud service status data compare to user-reported sources?
The dataset we analyze contains cloud service status data reported by cloud service providers themselves as well as from websites which gather user-reported data. The question that arises is how do these two sources of data compare to each other with respect to the statistics gathered in RQ.3. Answering this is the goal of this research question. A potential issue here could be whether there is enough data from both types of sources to draw meaningful conclusions, but should nonetheless provide insight into how cloud providers versus users report their status information.

1.2 Approach

RQ.1 In order to combine the data from various sources into a single and uniform dataset, we first inspect a small sample of HTML documents from each source. For each sample, we take note of any relevant information that could be useful for performing a statistical analysis on cloud service failures. At this point we can determine which of the relevant data are common between all sources, and can thus be considered for inclusion in the final dataset we will compile. The creation of the final dataset is broken down into four steps and explained further in section 3.6: **(1)** data parsing, **(2)** status encoding, **(3)** data unification and aggregation, and **(4)** data validation.

RQ.2 The popularity of the cloud service providers and their sources of status information present in the data archive are approximated in several ways. For cloud services, we identify several website ranking services and then check where each cloud service ranks in their lists of top websites. For sources of status data, we perform search engine queries and note which sources appear on the first page. This is important to help determine the relevancy of the cloud services that we analyze.

RQ.3 We use the single, uniform dataset we compile as part of research question RQ.1 to carry out our statistical analysis in chapter 4. The Empirical Cumulative Distribution Function (ECDF) plots are created using the Python module *statsmodels*. We build our own Python functions to determine the following: **(1)** the mean time between failures (MTBF), and **(2)** the mean time to repair failures (MTTR).

RQ.4 This approach to this research question is an extension of research question RQ.3, except that we now compare the statistics based on the source of cloud service failure data, which is either self-reported or crowdsourced. An issue we encounter in this process is that the vast majority (nearly 85%) of reports are crowdsourced, making it difficult to draw a meaningful conclusion between the two types of reports. Although we cannot solve the issue regarding the imbalance in the number of self-versus-crowdsourced reports, we do provide suggestions to help correct this for future work.

1.3 Study Contributions

The primary *Conceptual*, *Experimental*, and *Technical* contributions of this master thesis, in relation to our research questions, are as follows:

1. *(Technical, RQ.1)* Data parsers, encoders, aggregators, and validation scripts (Chapter 3).

2. *(Technical, RQ.1)* Single unified dataset (Chapter 3).

3. *(Conceptual, RQ.2)* Analysis of the popularity of cloud services (Chapter 3).

4. *(Conceptual, RQ.2)* Analysis of the relevancy of sources of cloud service status data (Chapter 3).

5. *(Experimental, RQ.3)* Statistical analysis of cloud service failures (Chapter 4).

6. *(Experimental, RQ.4)* Comparison of self-reported versus crowdsourced cloud service failures (Chapter 4).

2

Background

Cloud service failures are generally categorized as either minor or major outages (12). The occurrence of outages can be decreased when a cloud service provider increases the number of geographical regions or zones where they operate, which means an individual failure does not result in a collapse of the entire system (13). This increase in redundancy should lead to an increase in minor outages and a decrease in major outages. As far as we are aware, this is the first study of its kind investigating cloud service failures that distinguish between different types of failure events. This section provides background information into the area of cloud service failures, including related work.

Cloud service failure data is either provided by the source itself or via crowdsourced user reports. Failure data provided by the source is usually presented the form of log files, tickets, failure records, system metrics, and/or manual reports which may not be made publicly available or open source (14); however, some providers also publish the status of their cloud service(s) online. A problem is that this information may not be accurate, which may be coincidental or by-design. For instance, a provider's status tracking system may not catch all failures all of the time, which could be considered coincidental. On the other hand, a provider may decide to withhold certain failure data or "sugar-coat" the details to prevent damage to the their reputation or revenue (9, 10). Sources of crowdsourced user reports do not suffer from this issue since cloud service providers have no control over such reports, and are thus a way to hold providers accountable. Nevertheless, crowdsourced data also has its flaws regarding the trustworthiness of the reports. For instance, an incident may be considered as a partial outage because only a small subset of users were affected, but if these users could not access the system for weeks then this could be considered a major outage. A case similar to this happened with Atlassian in April 2022 (15).

The most common properties extracted from cloud failure data are the root causes of failure, the mean time between failures (MTBF), and the mean time to {repair, resolve, recover, respond} to a failure (MTTR). Root causes include anything from failing hardware/software to planned maintenance and even natural disasters. The MTBF value represents the average amount of time between failures, where the failure is *repairable* (16). A higher MTBF value represents a more reliable and available system. In contrast, lower MTTR values indicate higher efficiency in solving outage events. Figure 2.1 illustrates several examples of how MTBF and MTTR metrics are applied.

The most prominent root causes of cloud failures are the result of service upgrades (16%), networking issues (15%), and software bugs (15%) (5). Research on over 12,000 public cloud servers finds that the MTBF for such systems is 12.6 days, with mean and median TTR values of 5.56 and 0.23 hours, respectively (17). Another study finds results for a hybrid cloud system which has an MTBF of 22.26 hours and an MTTR of 10.22 hours (18). Although the results from these two studies focus on different cloud system types, both report high levels of variance in their findings, which is a theme we also encounter in our research.

In this study, we analyze data provided by cloud service providers as well as crowdsourced data made publicly available online. The data has been collected from multiple sources, and thus consists of their own unique structure. These websites allow users to report when they are experiencing issues with a particular cloud service. These websites present cloud outage information in their own unique manner. The most common cloud status reporting methods we encounter are as follows: **(1)** a single, global status report, **(2)** multiple status reports broken down into, for example, geographical regions and/or sub-services, **(3)** a count of the number of users experiencing issues, and **(4)** detailed information in the form of web feeds (e.g. *JSON* or *RSS* feeds). In this study we focus on sources that use the first two reporting methods since these sources categorize on the type of failure by default. Method three is not used as it is difficult to classify the severity of a failure based on an arbitrary number of users reporting issues. The JSON/RSS feeds in method four provide detailed information regarding the *root cause* of failure along with a timeline of the recovery process; however, the format and type of information provided between the sources vary widely enough that they do not fit into the single unified dataset we compile.

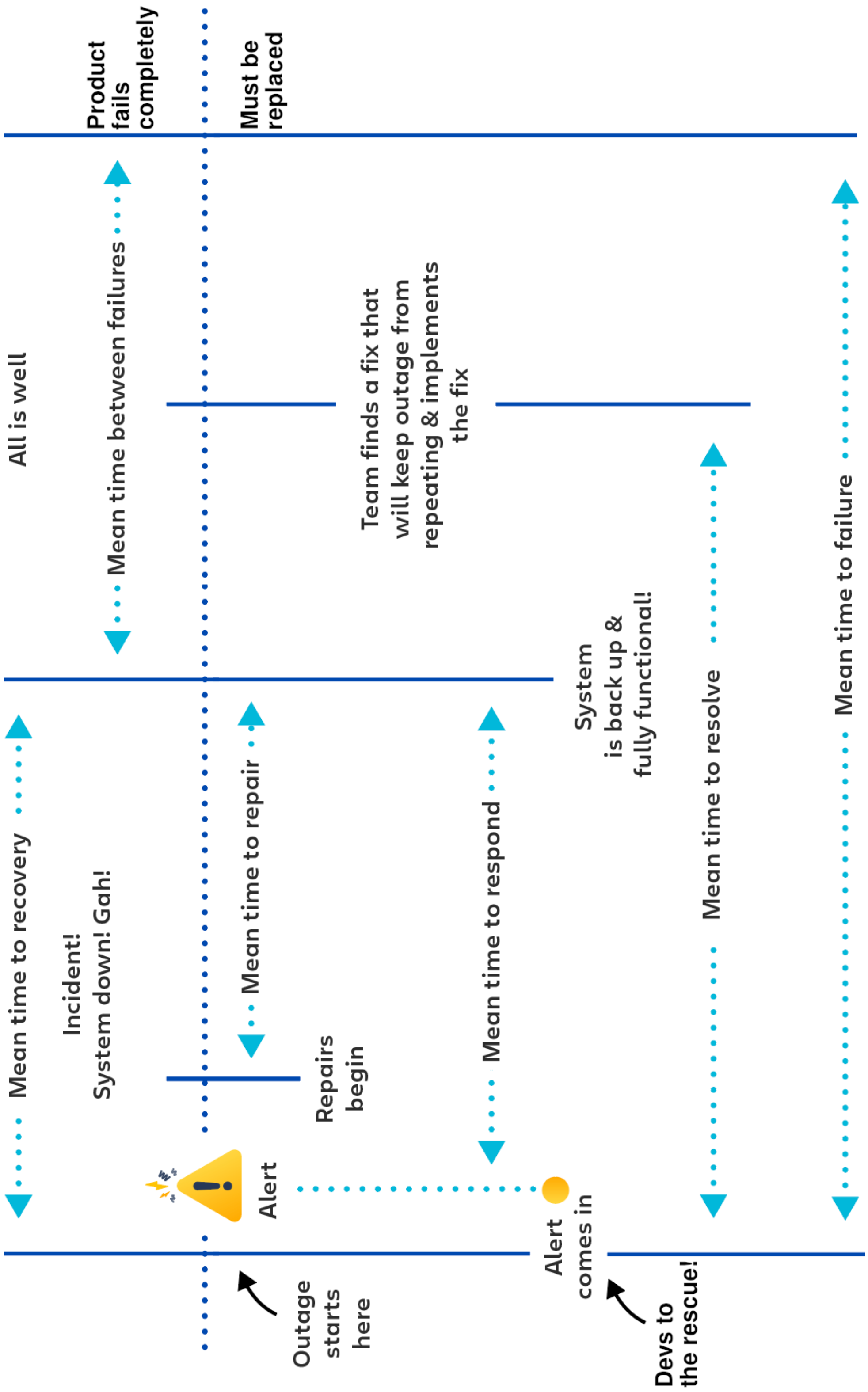


Figure 2.1: MTBF and the various types of MTTR. Image source: (16).

3

Study Design

3.1 Research Questions

The goal of this thesis is to provide new insight into the cloud failure landscape. To achieve this goal, we present our main research question: how do we analyze cloud failure data that has been gathered from multiple sources and exist in widely different formats? Our main research question can be broken down into several sub-questions which are listed in Table 3.1.

Sub-Question	Description
RQ.1	How do we combine cloud failure data from various sources into a single, uniform, and easy-to-use dataset?
RQ.2	What is a good process for selecting cloud services and their sources of status information and how does the current selection compare to this process?
RQ.3	What are the statistical properties of cloud service failures in relation to their type, the mean time between failures, and the mean time to repair them?
RQ.4	How do self-reported sources of cloud service status data compare to user-reported sources?

Table 3.1: Research questions.

3.2 Requirements Analysis

The purpose of this section is to identify stakeholders, use-cases, and (non-)functional requirements to satisfy research question RQ.1. This will provide an understanding of how to compile a single, unified, and easy-to-use dataset of cloud failure data from many sources, along with its intended use.

3.2.1 Stakeholders

We identify the primary stakeholders (*SH*) as scientific researchers and cloud service providers, which are described in further detail below.

SH.1 **Scientific Researchers** are those involved in academia. These stakeholders may use, reference, further analyze, and expand upon the dataset we compile. An important aspect of this dataset is that it is easy-to-use and open-source so that other researchers can contribute to it and/or verify its correctness.

SH.2 **Cloud Service Providers** are the companies or organizations that operate one or more cloud services. These stakeholders can reference the data we provide in our compiled dataset to their own data. The cloud service provider can compare their self-reported failure data to that of crowdsourced failure data to possibly identify discrepancies between reporting methods. The cloud service provider can determine whether they fail to report or even identify certain failure events, which may then be addressed.

3.2.2 Use Cases

This section describes use-cases related to research question RQ.1 and the stakeholders (*SH*) listed in section 3.2.1. We identify three specific use-cases, namely: *statistical failure analysis*, *performance optimization*, and *validating private failure reports*.

UC.1 **Statistical Failure Analysis** (SH.1): In this thesis we calculate the following two main metrics: the mean time between failures (MTBF) and the mean time to repair failures (MTTR). In addition, we plot the empirical cumulative distribution functions and also compare the metrics of self-reported versus user-reported data sources. There are further opportunities to explore and statistically analyze the dataset we compile.

UC.2 Performance Optimization (SH.1): The data processing pipeline we employ (i.e. data parsing, encoding, aggregation, and validation) is currently designed to run using Python3 multi-processing on a single, multi-core CPU. An optimization is to make this pipeline horizontally scalable and distributed in nature.

UC.2 Validating Private Failure Reports (SH.2): Although some cloud service providers publicize their failure data, many decide to keep this data private. For the providers who wish to keep their data private, they are still able to use our dataset to benchmark their failure data with the providers we analyze in our dataset. The benefit of using our single, unified dataset is that ours contains a fair mix of popular versus less-popular cloud services, as described in section 3.4, and classifies failures based on level of severity.

3.2.3 Functional Requirements

This section describes the functional requirements (*FR*) as they relate to the single, unified, and easy-to-use dataset we compile as part of research question RQ.1. The requirements also involve the stakeholders (*SH*) listed in section 3.2.1 and the use-cases (*UC*) listed in section 3.2.2. The description of each functional requirement begins with "The dataset must" and are the following:

FR.1 Maintain a consistent naming convention for cloud services and sources of status data (UC.1): The dataset must identify cloud services and sources of status data in a consistent manner in order to ensure correct data aggregation. For instance, if a cloud service name is reported by *source1* as "YouTube" and by *source2* as "youtube" (and so forth), they must be made consistent in their formatting.

FR.2 Report the status of cloud services by severity level (UC.1, UC.3): The dataset must classify cloud service statuses by level of severity. This must hold true for every record in the dataset and also demands that there shall be no data entries where the severity level is unknown (unless explicitly classified as so).

FR.3 Report timestamps in a uniform and consistent format (UC.1, UC.3): The dataset must, for each data record, report a timestamp in a format that is consistent across all records. For example, the timestamp could be of when the data was collected. This is required for time series analysis.

FR.4 Present quantified data in a normalized manner (UC.1): The dataset must, for each data record, store quantified data in a normalized format. This is important for performing a statistical analysis on the data.

FR.5 Saved in a commonly used data format (UC.1, UC.2, UC.3): The dataset must be saved in a popular and widely-used data format, such as a text CSV file. This relates to the *ease-of-use* of the dataset referred to in SH.1.

FR.6 Be easily usable by popular programming languages (UC.1, UC.2): The dataset must be easy to import, read from, and write to using popular programming languages. This relates back to FR.5, as most commonly used programming languages support performing the aforementioned actions on data formats such as text CSV files.

3.2.4 Non-Functional Requirements

Below lists the non-functional requirements (*NFR*) as they relate to the single, unified, and easy-to-use dataset we compile as part of research question RQ.1 and the use-cases (*UC*) described in section 3.2.2.

NFR.1 An easy to use dataset (UC.1, UC.3): The dataset should be in a format that is easy to understand, process, and analyze, making it easy for both researchers and third-parties to utilize.

NFR.2 An easy to expand dataset (UC.1): It should be easy for a researcher to add data to the dataset. Examples of such data include those from additional cloud services and/or sources of status information.

NFR.3 Can be made horizontally scalable (UC.2): The data processing pipeline can be scaled horizontally by a researcher with some experience in distributed systems/workloads, without having to extensively re-configure the data processing pipeline.

3.3 Data Archive

The data archive is a 19.46GB compressed .sqsh file consisting of 46 directories. The directory names represent either the name of a cloud service (e.g. Slack) or the source which tracks their status (e.g. Downdetector). Downdetector is the only source which appears more than once (30 times in total), where each directory contains data recorded from a particular country. The directories consist of a hierarchy of files and folders, illustrated in Figure 3.1, and contain HTML files which hold the status information of one or more cloud services. The directory containing the HTML files follow a naming convention that represents the timestamp in which the HTML data was collected.

```
slack
└── 2020
    ├── 20203.zip
    │   ├── 20200326.zip
    │   │   ├── 20200326T120001
    │   │   │   ├── global-status.html
    │   │   │   └── rss-feed.html
```

Figure 3.1: Example hierarchy of the compressed data archive.

In total we find data for 54 unique cloud services which are presented in Table 3.2. The data spans a range of two years and eight months from October 2017 to June 2020. Although some data sources span this entire duration, others only span a subset, the smallest being one month. The mean and median date ranges are 12.7 and 11 months, respectively. The amount of data present for each source ranges anywhere from 156KB to several gigabytes. This large difference may be explained by both the duration of the recorded data as well as the size of the individual files.

3.4 Justification for the selection of cloud services

The dataset we analyze contains the status information for 54 unique cloud service providers. For the purpose of this section, we consider similar cloud services such as *Jira-Align*, *Jira-Core*, and *Jira-Software*, as a single service (in this case, *Jira*), resulting in 46 unique cloud services. Currently, there is no documented justification or reasoning for the selection of these cloud services. Ideally, each of the 46 cloud services present in the dataset should appear in at least one list of most popular websites compiled by third-party sources. We analyze the popularity of each cloud service using five website ranking sources: *Ahrefs*, *Alexa*, *Moz*, *Rankranger*, and *Similarweb*. The website ranking sources were discovered via Google search akin to "top website rankings (by category)". We only consider *global ranking data* from *original sources* that provide this information *free of charge*.

Table 3.3 lists the total number of websites reported for each website ranking source, the date the ranks were collected, the rank duration (or *freshness*), and the metric used for ranking. The number of websites ranked ranges from 50 to 500, with the freshness of

Data Source	Date Start	Date End	# Months	Zip Size	Cloud Services Reported
Amazon Web Services	2017-11-01	2020-06-10	31	1.5 GB	Amazon Web Services
Apple	2017-11-22	2020-06-10	31	4.7 MB	Apple Consumer
Atlassian	2020-03-26	2020-06-10	3	536 MB	Access, Bitbucket, Confluence, Developers, Jira Align, Jira Core, Jira Service Desk, Jira Software, Opsgenie, Partners, Statuspage, Support, Trello
Cloudflare	2017-11-25	2020-06-10	31	27 MB	Cloudflare
Discord	2020-03-26	2020-06-10	3	38 MB	Discord
Docker	2020-03-26	2020-06-10	3	1.5 MB	Docker
Downdetector-*	2017-10-23	2018-09-18	11	7.2 GB	Airbnb, Amazon, Lyft, Netflix, Pinterest, Reddit, Slack, Snapchat, Spotify, YouTube, Zynga
DownRightNow	2017-10-23	2020-06-10	32	2.1 GB	Blogger, Facebook, Foursquare, Gmail, Hotmail, Linkedin, Livejournal, Netflix, Ning, Paypal, Skype, Tumblr, Twitter, Typepad, YahooMail, YouTube
GitHub	2019-05-07	2020-06-10	13	193 MB	GitHub
Google Apps	2017-11-22	2020-06-10	31	1.1 GB	Google Apps
Google Cloud	2017-11-01	2020-06-10	31	1.8 GB	Google Cloud Platform
gPanel	2019-06-25	2020-06-10	12	1.4 MB	gPanel
Microsoft Azure	2019-06-22	2020-06-10	12	222 MB	Microsoft Azure
Minecraft	2020-03-26	2020-06-10	3	9.7 MB	Minecraft
Nintendo	2020-05-10	2020-06-10	1	156 KB	Nintendo
Outage.Report	2019-04-15	2020-06-10	14	4.2 GB	Apple Servers, Facebook, Facebook Messenger, GitHub, Gmail, Instagram, Netflix, Snapchat, Skype, Twitter, WhatsApp, YouTube
Slack	2020-03-26	2020-06-10	3	3.3 MB	Slack

Table 3.2: Properties of the data archive.

the ranking data spanning from 1 day to 3 months. The websites are ranked based on certain metrics, some which need further explanation. *Organic traffic* refers to the number of direct website visits via search engines, *link-based* is how many other websites link to a page, and the *Domain Authority score* is a machine learning metric developed by Moz that predicts site rankings in search engine results. The ranking metric used by Similarweb is described only as *traffic*, as they do not explicitly state whether this includes non-organic traffic.

For each of the five website ranking sources, we perform a search to determine which cloud services from our dataset appear in any of the ranking lists. Figure 3.2 displays the top 100 and 500 overall global website rankings for the cloud services in our dataset. 17 of the cloud services are present in the top 50 (36.96%), 20 are in the top 100 (43.48%), and

Source	Top n	Date (2021)	Duration	Ranking Metric
Ahrefs	100	01 Jan	1 month	Organic traffic
Alexa	50	24 May	3 months	Average daily visitors, page views
Moz	500	24 May	1 day	Linked-based, Domain Authority score
Rankranger	100	23 May	1 day	Organic traffic
Similarweb	50	01 Apr	1 month	Traffic

Table 3.3: Website ranking sources.

25 reside in the top 500 (54.34%). This means that roughly half of the cloud services in the dataset are not present in any of the third-party rankings lists, indicating that there is a balanced mix of highly – versus less – popular cloud services. A cloud service that does not appear in any of the rankings does not necessarily mean it is unworthy of attention, but we are unable to justify their selection. We can safely approve of the selection of cloud services that appear in at least one of the top n ranking lists.

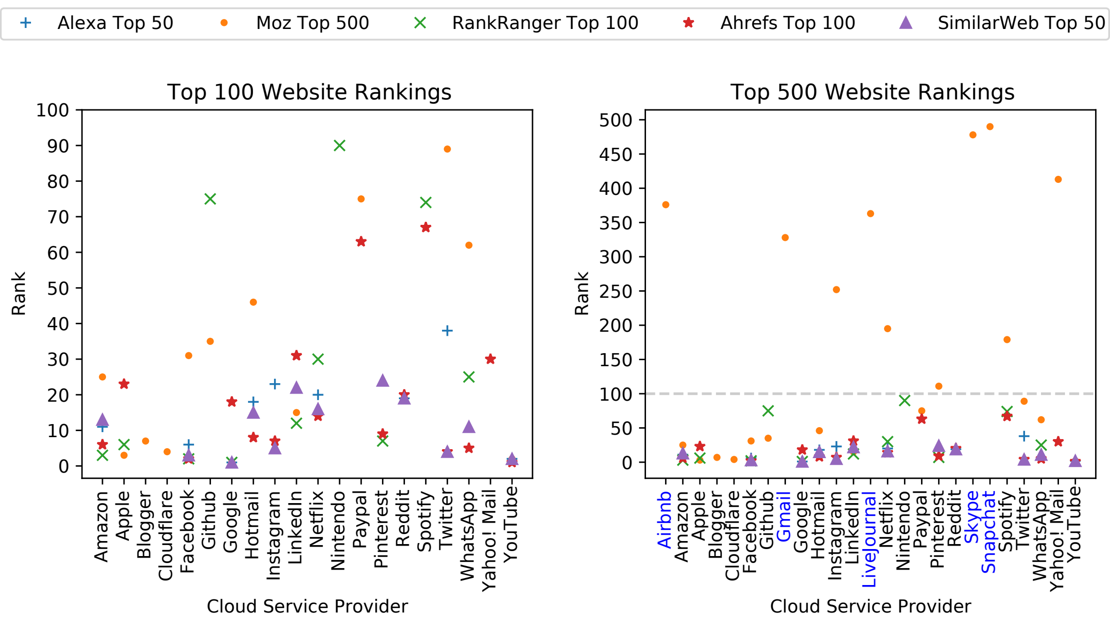


Figure 3.2: Website rankings for the cloud services present in our dataset. Blue represents those not present in the Top 100.

Some of the cloud service rankings are inconsistent across the website ranking sources. For instance, Blogger, Cloudflare, and Nintendo are ranked only by a single source; Netflix is ranked by each of the five sources, but these rankings range from 14 to 195. Possible explanations for these anomalies may involve the data freshness and/or the metric used to calculate the rank. Further investigation into *how* websites are ranked is out of the scope of this study.

Similarweb also provides top 50 global website rankings per category alongside their overall rankings. We determine which of our cloud services appear in this categorical ranking data, and they are displayed in Table 3.4. 21 of the cloud services in our dataset (45.65%) are ranked in the top 50 for at least one of the categories. Cloud services may also have a rank for multiple categories, as is the case with Airbnb and Lyft. We find the cloud services in our dataset existing in a total of seven out of 24 categories, indicating a lack of diversity across all categories. Table 3.5 lists these 24 categories along with the top three websites that fall into each, which may be useful for considering other cloud services in future work.

We can further justify the selection of the cloud service providers in our dataset by determining how many sources of user-reported data track these services. To achieve this, we perform the Google search query "<cloud_service_name> service status" for each of the cloud services in our dataset, and record the number of unique sources of user-reported data that appear on the first page of the query. Only the first page is considered for two reasons: **(1)** 75% of users only visit results on the first page (19), and **(2)** our goal is to get an indication of a cloud services popularity. The results are displayed in Figure 3.3. 39 out of 46 (84.78%) cloud services have at least one source of user-reported status data, as per our Google query, with 35 (76.09%) having at least three sources. The seven cloud services that do not appear in the figure may still be tracked by one or more sources, but in this case we only consider sources that appear on the first page of our search query.

3.5 Justification for the selection of user-reported sources

Our dataset contains cloud service status data from both cloud service providers and crowdsourced user data; however, there is no documented justification or reasoning for the selection of these sources. The dataset contains user-reported data from three sources: *Downdetector*, *Outage.Report*, and *DownRightNow*. We investigate how popular these three sources of user-reported data are and identify other popular sources for comparison. To achieve this, we first perform the Google search query "<cloud_service_name> service status" for each of the cloud services present in the dataset. For the same reasons as our query for cloud services, the first page is considered for two reasons: **(1)** 75% of users only visit results on the first page (19), and **(2)** our goal is to get an indication

CATEGORY	SERVICE	RANK
Arts and Entertainment	YouTube	1
	Netflix	2
	Spotify	7
Computers, Electronics, and Technology	Google	1
	Facebook	2
	Twitter	3
	Instagram	4
	WhatsApp	7
	Hotmail	9
	Reddit	10
	LinkedIn	13
	Pinterest	15
	Apple	31
	Github	34
	Tumblr	42
E-Commerce and Shopping	Amazon	1
Finance	Paypal	1
Games	Nintendo	33
	Minecraft	47
Travel and Tourism	Airbnb	4
	Lyft	46
Health	Airbnb	4
	Lyft	46

Table 3.4: Similarweb top 50 global rankings for cloud services in our dataset, per category.

of the popularity of user-reported sources. We record a count for each source of user-reported status information that appear on the first page of the Google search results, which are displayed in Figure 3.4. Downdetector, Outage.Report, and DownRightNow appear quite often in our query, indicating that they are popular sources of cloud status reports. Downdetector is the most popular, tracking 35 of the 46 cloud services in our dataset, with Outage.report (15/46) and DownRightNow (14/46) following. We consider these acceptable sources of crowdsourced data as each fall above the mean (9.71) and median (5) values of the number of services tracked among the sources. Other sources of user-

CATEGORY	RANK 1	RANK 2	RANK 3
Adult	xvideos.com	xnxx.com	pornhub.com
Arts and entertainment	youtube.com	netflix.com	fandom.com
Business and consumer services	canadapost-postescanada.ca	zillow.com	usps.com
Community and society	jw.org	livehdcams.com	tinder.com
Computers electronics and technology	google.com	facebook.com	twitter.com
E commerce and shopping	amazon.com	ebay.com	amazon.co.jp
Finance	paypal.com	binance.com	coinmarketcap.com
Food and drink	trilltrill.jp	cookpad.com	tabelog.com
Gambling	bet365.com	eshkol.io	caliente.mx
Games	twitch.tv	roblox.com	chess.com
Health	booking.com	tripadvisor.com	uber.com
Heavy industry and engineering	edf.fr	grainger.com	archdaily.com
Hobbies and leisure	shutterstock.com	flickr.com	ancestry.com
Home and garden	homedepot.com	ikea.com	lowes.com
Jobs and career	indeed.com	jooble.org	glassdoor.com
Law and government	gov.uk	irs.gov	service.gov.uk
Lifestyle	shein.com	linktr.ee	nike.com
News and media	yahoo.com	yahoo.co.jp	naver.com
Pets and animals	chewy.com	petfinder.com	ironsource.mobi
Reference materials	wikipedia.org	quora.com	worldometers.info
Science and education	weather.com	accuweather.com	brainly.co.id
Sports	espn.com	cricbuzz.com	marca.com
Travel and tourism	booking.com	tripadvisor.com	uber.com
Vehicles	drom.ru	motorbiscuit.com	auto.ru

Table 3.5: Top three globally ranked websites per category (Similarweb). Red indicates services present in our dataset.

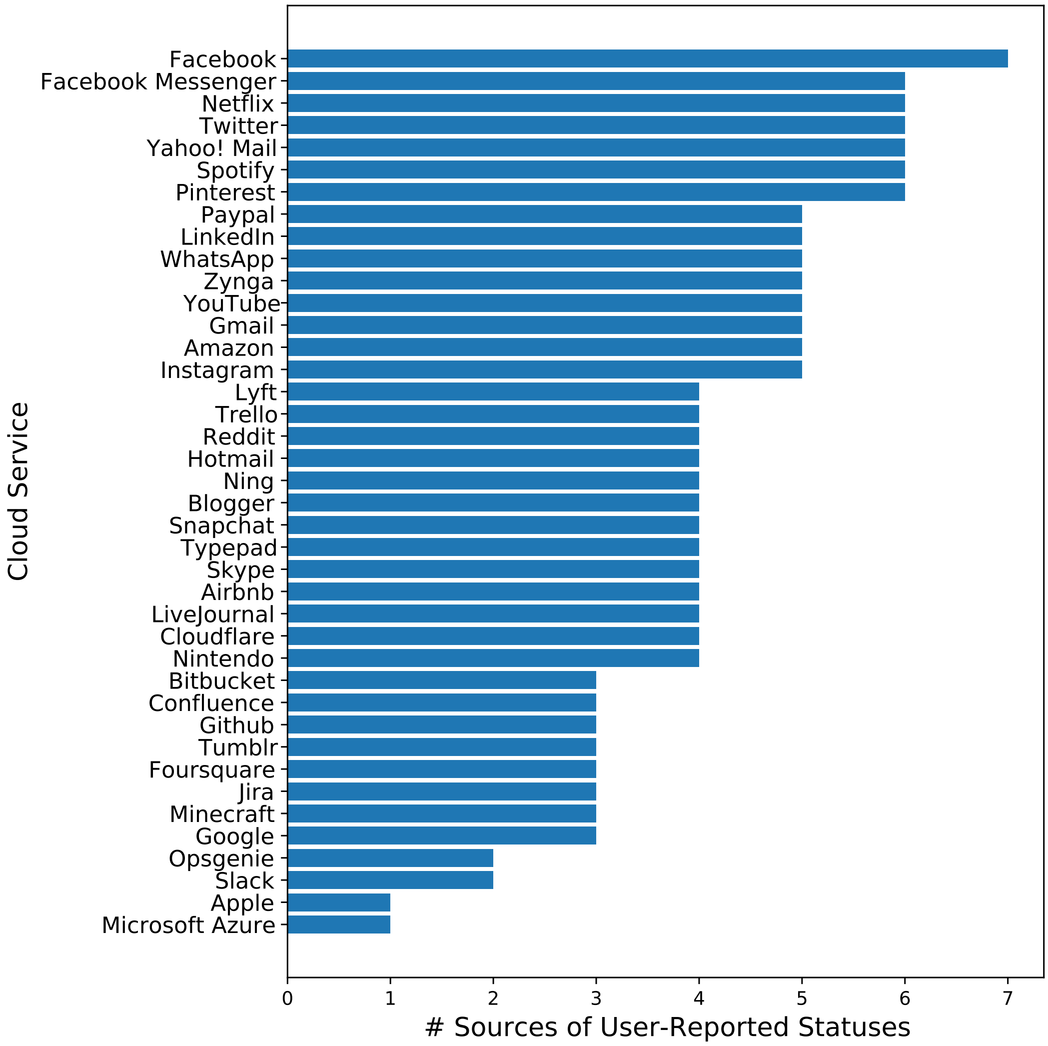


Figure 3.3: Number of unique sources of user-reported status data per cloud service.

reported data appear to be quite popular as well. The sources that track a higher number of cloud services than the mean of 9.71 include: *IsItDownRightNow*, *IsTheServiceDown*, *ServicesDown*, and *StatusGator*. These other sources could be considered as additional sources of cloud service status data for future work.

3.6 Creating a single dataset

The data archive we analyze contains status data for 54 unique cloud services collected from multiple sources. All files are in HTML format, although the structure and information they contain widely vary. One of our goals is to combine all of the data into a single,

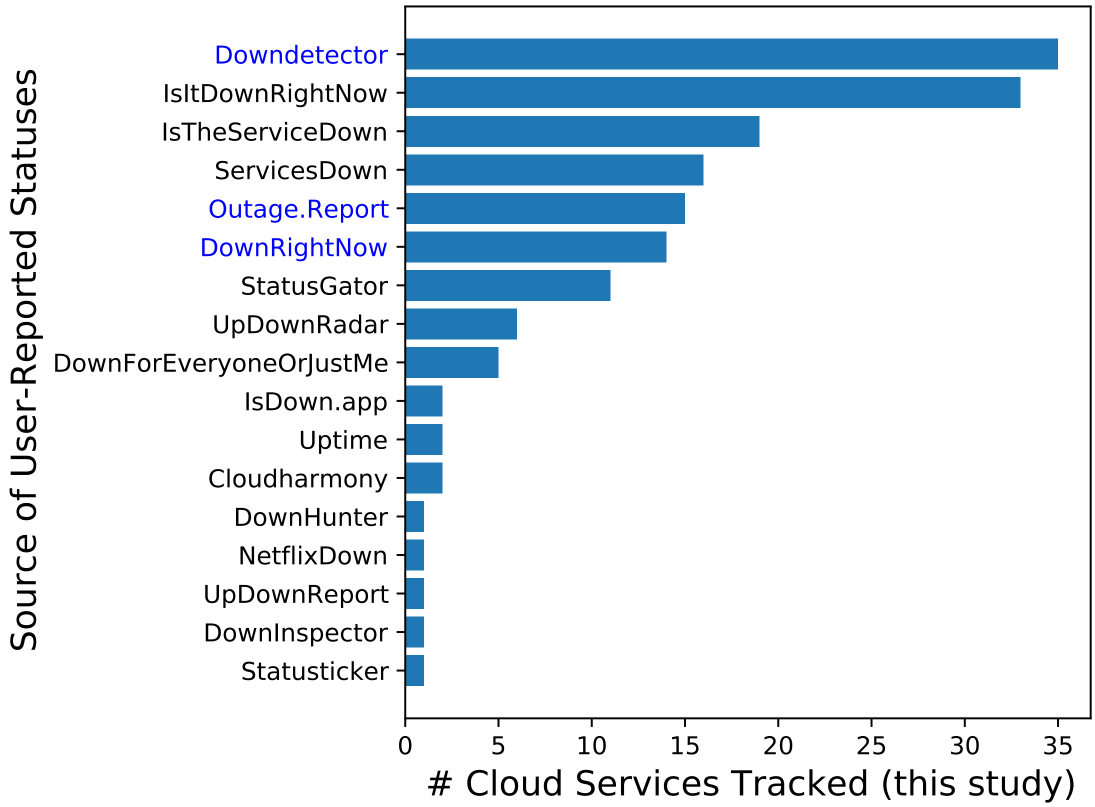


Figure 3.4: Number of cloud services in this study tracked by third-party websites. Blue represents sources present in our dataset.

uniform, easy-to-use dataset, and then perform a statistical analysis. To achieve these goals, we break down our approach into four steps: **(1)** *data parsing*, **(2)** *status encoding*, **(3)** *data unification and aggregation*, and **(4)** *data validation*.

3.6.1 Data parsing

The cloud service status information in our data archive exist as HTML files from many sources containing their own format/structure. One of our goals is to combine all of this data into a single, uniform, and easy-to-use dataset. The first step is to create parsers for each source individually, which is carried out using Python3 Jupyter Notebooks. Initial inspection reveals that the HTML files exist in one of two general formats: traditional HTML found on a typical webpage and those containing JSON-like web feeds. In this study, we focus on the data in traditional HTML formats, and refer to this as *HTML* for future context. In addition, we only consider data that has been *qualified* (failure classification) and not *quantified* (number of reports), as we do not have a method for qualifying the quantified data.

Our dataset consists of 46 top-level directories for which we find 37 containing data suitable for HTML parsing. These 37 sources include cloud service status data for 42 out of the 54 unique cloud services present in the original dataset. We manually inspect a small sample of HTML files for each source and take note of HTML tags containing information relevant to the status of the cloud service. The amount and type of information provided varies between sources. In the simplest cases, relevant data consists of a timestamp, the global status of the cloud service, and possibly an indication of when the problems began. The most complex data include status reports for dozens of sub-services per geographical region. For each source, we record the name of the cloud service, any relevant timestamps, and the current status of the (sub-)services reported. The timestamps recorded are those from the directory names in the data archive, which are formatted as timestamp, and website metadata (if provided). Sources report on either a single global status or those broken down into sub-categories. Examples of sub-categories include regional statuses and sub-services (e.g. databases, login systems, front-end, etc). The number of sub-categorical statuses reported by all sources ranges from one to 248. Each source also uses a set of tags to describe the status of a (sub-)service. Examples of tags include *operational*, *partial outage*, and *major outage*, with the number of possible status tags present for a source ranges from two to 25. Table 3.6 summarizes the above properties, including the number of data records present, for each source of traditional HTML data.

Each sources is parsed individually and saved as an intermediate dataset. From this point we begin the process of combining the intermediate datasets into a single, uniform, and easy-to-use dataset. First, we must encode the status tags so they are consistent across the data sources. This encoding process is described in the following section.

3.6.2 Status encoding

The intermediate datasets we obtain from the data parsing step (see 3.6.1) contain a total of 59 status descriptions for which 46 are unique. Upon inspecting the list of unique status descriptions, we determine that they fall into 5 general categories: **(1)** *operational*, **(2)** *partial outages*, **(3)** *major outages*, **(4)** *maintenance events*, and **(5)** *unknown*. Encoding each status into one of these five categories enables us to eventually aggregate and combine the data from all sources into a single dataset.

Each unique status description is manually assigned a numeric value based on the category they fall into. The mapping of numeric encoding to each status tag is displayed

Source	Services	Unique Statuses	Sub-Categorical Reports	Data Records
Atlassian	13	6	175	23,864
Cloudflare	1	8	248	8,433
Discord	1	4	21	1,836
Google Cloud	1	25	1	22,811
Downdetector-*	11	4	30	323,708
DownRightNow	16	5	1	357,900
GitHub	1	5	11	9,612
Slack	1	2	10	1,835
Total	45	**59**	**468**	**749,999**
Total (unique)	42	**46**	**458**	**749,999**

Table 3.6: General properties of HTML files for each source (* represents 30 geographical regions).

in Table 3.7. 71.74% of status tag descriptions fall into the partial outage encoding, follow by operational (13.04%), major outages (6.52%), maintenance (4.35%), and unknown (4.35%). Status tags for *maintenance events* are put into their own category as we cannot determine whether this maintenance resulted in an outage, and if so, what type of outage. The *unknown* encoding contains status tags for *NaN* and *?* values. A *NaN* value is present when there is no data reported for a particular column, which may or may not be problematic, are handled during the data validation process in Section 3.6.4. A *?* indicates that an error was caught in the parsing stage, which is always problematic; however, we only encounter 84 occurrences of *?* status tags out of 749,999 records (0.01%). Upon inspection of these files we discover faulty HTML data, so we discard them.

3.6.3 Data aggregation and unification

The intermediate datasets we parse and encode in Sections 3.6.1 and 3.6.2 can now be aggregated and unified. In the encoding step, we assign one of five numeric values based on the status of a cloud service at each data point. The number of sub-categorical reports that make up a single status report ranges from one to 248, and there are a total of 458 unique sub-categorical report types in total. Using Table 3.6 we determine that 50.76% of all data we encode only contain reports on the global status of the service itself (i.e. the number of sub-categorical reports is equal to one). Unifying the encoded data without aggregation would result in an unnecessarily large and sparse dataset, which is not useful when performing a general statistical analysis. Instead, we aggregate the encoded statuses across sub-categorical reports by counting the occurrences of each encoding for every data entry. This aggregation results in a dataset with a uniform number of features for each source. Concatenating the datasets results in a single, uniform, and easy-to-use dataset. A sample of the unified dataset is displayed in Figure 3.5. Even in this small data sample, the column *timestamp_site* shows inconsistent formatting and even NaN values. We address such issues by validating the unified dataset, as described in the following section.

3.6.4 Data validation

In Section 3.6.3 we compile a unified dataset containing cloud service status information from multiple sources. The dataset contains 749,999 record in total. A small sample of this dataset is shown in Figure 3.5. We must first validate the unified dataset before performing our statistical analysis. The validation step ensures that our data is consistent and sane.

We find a total of 24,646 NaN values in our dataset, and 100% of these NaN values reside in the *timestamp_site* column, referring to timestamp information included in website metadata. Upon further inspection, we confirm that not all sources report this information. Luckily the directory names of the data archive we use are timestamp values of when the data was collected, and are present in the *timestamp_dir* column. We discard the *timestamp_site* column and use *timestamp_dir* for our statistical analysis, as this column contains no NaN values.

The columns *operational_count*, *partial_outage_count*, *major_outage_count*, and *maint_count* report the number of occurrences of the status encodings we perform in Section 3.6.2. If a record in our unified dataset contains zeros in each of these columns, then there are no status reports for that record. We check that each record in our unified dataset contains at least one value greater than zero across these columns and find that eight records that do not. Manually inspecting the HTML documents for each of these records shows that they either contain missing HTML data or no data at all. These records are discarded. At this point, we have compiled a unified and validated dataset ready for statistical analysis.

3.7 Study Replicability

The dataset, parsers, scripts, and output are all open-source and has been made publicly available at: *https://github.com/shanemin/thesis-msc-cloud-failures*.

Encoding	Description	Status Tag
0	Operational	All services available
		No issues
		Operational
		Up
		alert-success
		operational
1	Partial outage	Cloud Developer Tools reporting issues
		Cloud Machine Learning reporting issues
		Cloud Run reporting issues
		Cloud Spanner reporting issues
		Degraded Performance
		Google $<service\_name>$ reporting issues (x17)
		Identity & Security reporting issues
		Likely Service Disruption
		Multiple services reporting issues
		Operations reporting issues
		Partial Outage
		Possible Service Trouble
		Recent Signs of Service Trouble
		Something's not quite right View details
		alert-warning
		degraded_performance
		partial_outage
2	Major outage	Major Outage
		alert-danger
		major_outage
3	Maintenance	Under Maintenance
		under_maintenance
9	Unknown	?
		nan

Table 3.7: Encoding of unique status tag descriptions.

	source	service	timestamp_dir	timestamp_site	operational_count	partial_outage_count	major_outage_count	maint_count
4162	github-status	github	2019-10-27 22:00:01	2019-10-27 21:57:27	6	0	0	0
264292	downdetector-nld-netherlands	snapchat	2018-09-02 04:00:01	2018-09-02T06:00:28.835837+02:00	1	0	0	0
184112	downdetector-deu-germany	amazon	2017-12-18 19:00:01	2017-12-18T20:01:28.212137+01:00	0	1	0	0
117416	downdetector-jpn-japan	youtube	2017-11-30 16:00:01	2017-12-01T01:01:51.670986+09:00	0	1	0	0
214936	downdetector-fra-france	netflix	2017-12-27 03:00:01	2017-12-27T04:01:22.040615+01:00	1	0	0	0
263053	downdetector-sgp-singapore	youtube	2018-09-01 20:00:01	2018-09-02T04:00:01.786446+08:00	1	0	0	0
116519	downdetector-pol-poland	netflix	2017-11-30 10:00:01	2017-11-30T11:00:46.676522+01:00	1	0	0	0
737	cloudflare-status	cloudflare	2019-07-26 08:00:01	2019-07-26 07:56:21	207	8	0	0

Figure 3.5: A random sample from the unified dataset.

4

Statistical Analysis of Cloud Service Failures

The data archive we analyze consists of HTML documents containing status reports for 54 cloud services across 46 sources, of which we find 42 services from 37 sources suitable for failure analysis. A status report is either a global/binary status or is broken down into many sub-reports, for example by geographical region or sub-service type. We determine that statuses fall into one of four general categories: **(1)** *operational*, **(2)** *partial outages*, **(3)** *major outages*, or **(4)** *maintenance events*. Every status report is encoded based on the category they belong to, and while global/binary statuses can only live in one category at any given time, sub-reported data can exist across multiple categories. For sources that include sub-reported status information, we normalize the values by dividing the number of occurrences per category by the total count across all categories (i.e. we take the mean values). Using these normalized values, we calculate certain statistical properties of the cloud services in the dataset.

4.1 General statistics

Table 4.1 displays both the the raw count of all reported statuses in each category and the corresponding normalized value, rounded to the nearest integer. The vast majority of cloud services live in the operational category (93.93%), followed by partial outages (5.94%), major outages (0.13%), and maintenance events (0.01%). After normalizing the data, we notice a rise in the percentage of operational statuses and drops in partial and major outages as follows: operational 95.52% (+1.59%), partial outages 4.36% (-1.58%), Major outages 0.12% (+0.01%), and maintenance events 0.002% (+0.008%). These changes are the result of averaging the count, which may be as high as 248, of a single category across the total count of all categories. The low occurrence of maintenance events suggest rarity and only 14 of the 42 cloud services (33%) report on this type of event.

Category	**Raw Count**	**%**	**Normalized Count**	**%**
Operational	2,962,914	93.93	433,139	95.52
Partial Outage	187,281	5.94	19,752	4.36
Major Outage	4,179	0.13	561	0.12
Maintenance Event	160	0.01	7	0.002

Table 4.1: Number of cloud service status reports in each category.

Figure 4.1 displays the distributions of cloud service status reports based on the daily averages per category and are broken down into three plots: all reports, self-reports only, and user-reports only. Daily averages (and not one of finer granularity) are selected to address the sources that report only on global statuses, which avoids reporting on strictly binary status data. We therefore take daily averages to obtain non-binary data similar to sources that report on the status of multiple sub-services. The distributions are displayed as letter-value plots. Letter-value plots are useful for large datasets (20, 21). A traditional box plot contains two *letter-values*, which represent the median and quartiles. In a letter-value plot, additional letter-values are created as long as they are reasonable estimations of their respective quantiles. This results in more detailed information in the tails of the distribution. Each distribution show that the majority of cloud service status reports are highly operational. The partial outage state is a near inverse of the operational state, with major outages and maintenance events appearing quite rarely.

We notice that self-reported statuses contain many more letter-values and display higher availability when compared to user-reports. This may indicate the existence of a bias in one of the two types of reporting methods used. When comparing the mean and standard deviations of self-reported versus user-reported statuses, as shown in Table 4.2, we find that self-reported data has a 2.38% higher operational percentage along with less than half of the variance in each category. In addition, the percentage of partial and major outages more than doubles for user-reported statuses (maintenance events are only present for self-reported statuses). One thing to consider when comparing the reporting types is that our dataset contains a much larger sample of user-reported data: after performing a daily aggregation we end up with a total of 18,667 records, of which 2,859 are self-reported and 15,808 user-reported. The difference in the number of records between reporting methods

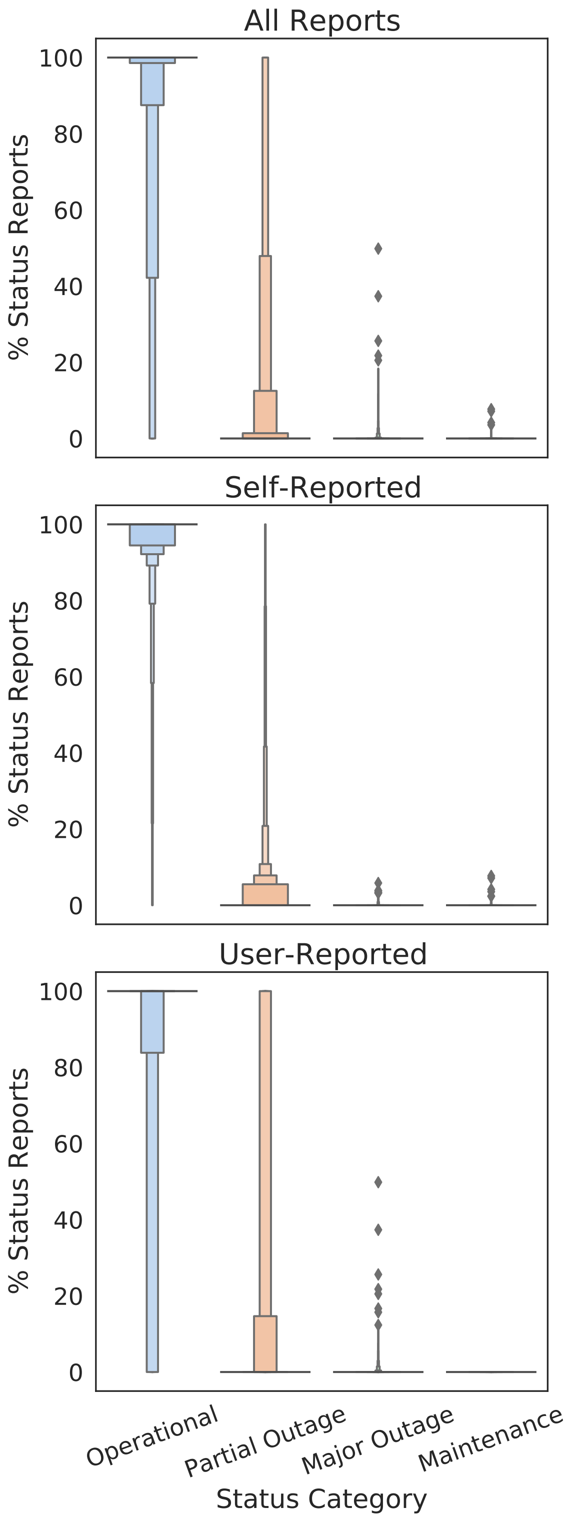


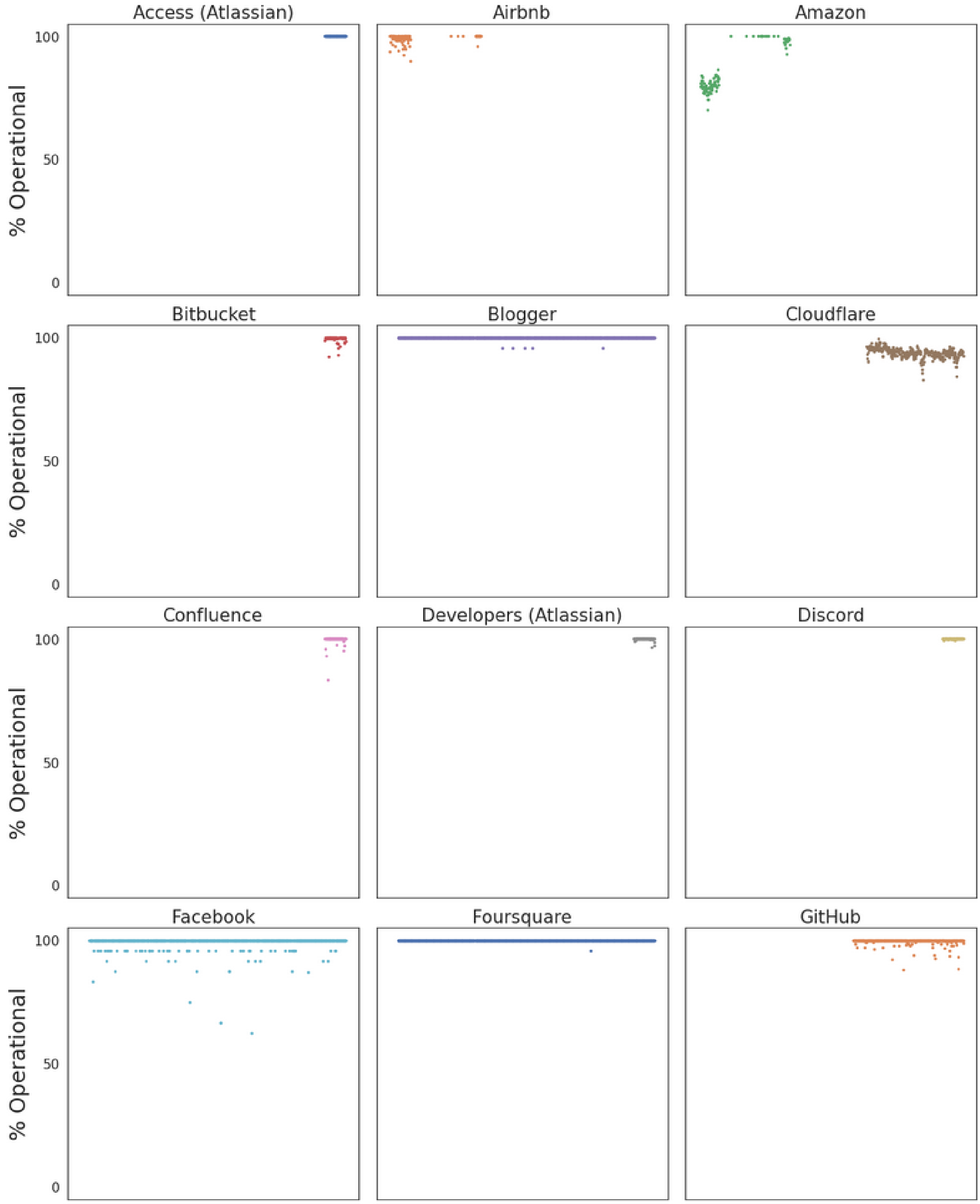
Figure 4.1: Distributions of cloud service status reports.

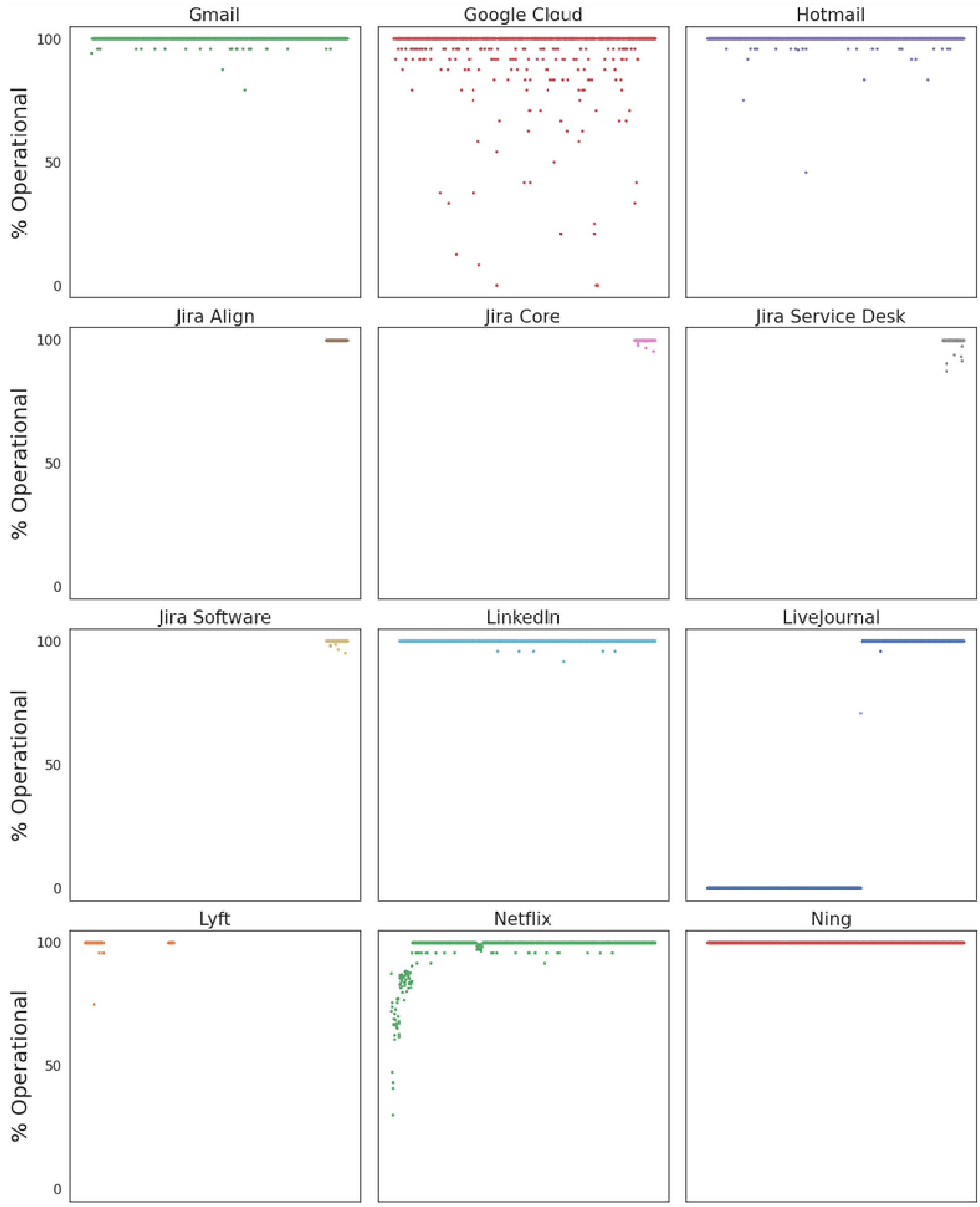
makes it difficult to draw conclusions on the resulting numbers. Furthermore, the median values are 100% for operational and 0% for each failure class. This makes sense since greater than 95% of status reports reside in the operational class across all reports, self-reports, and user-reports.

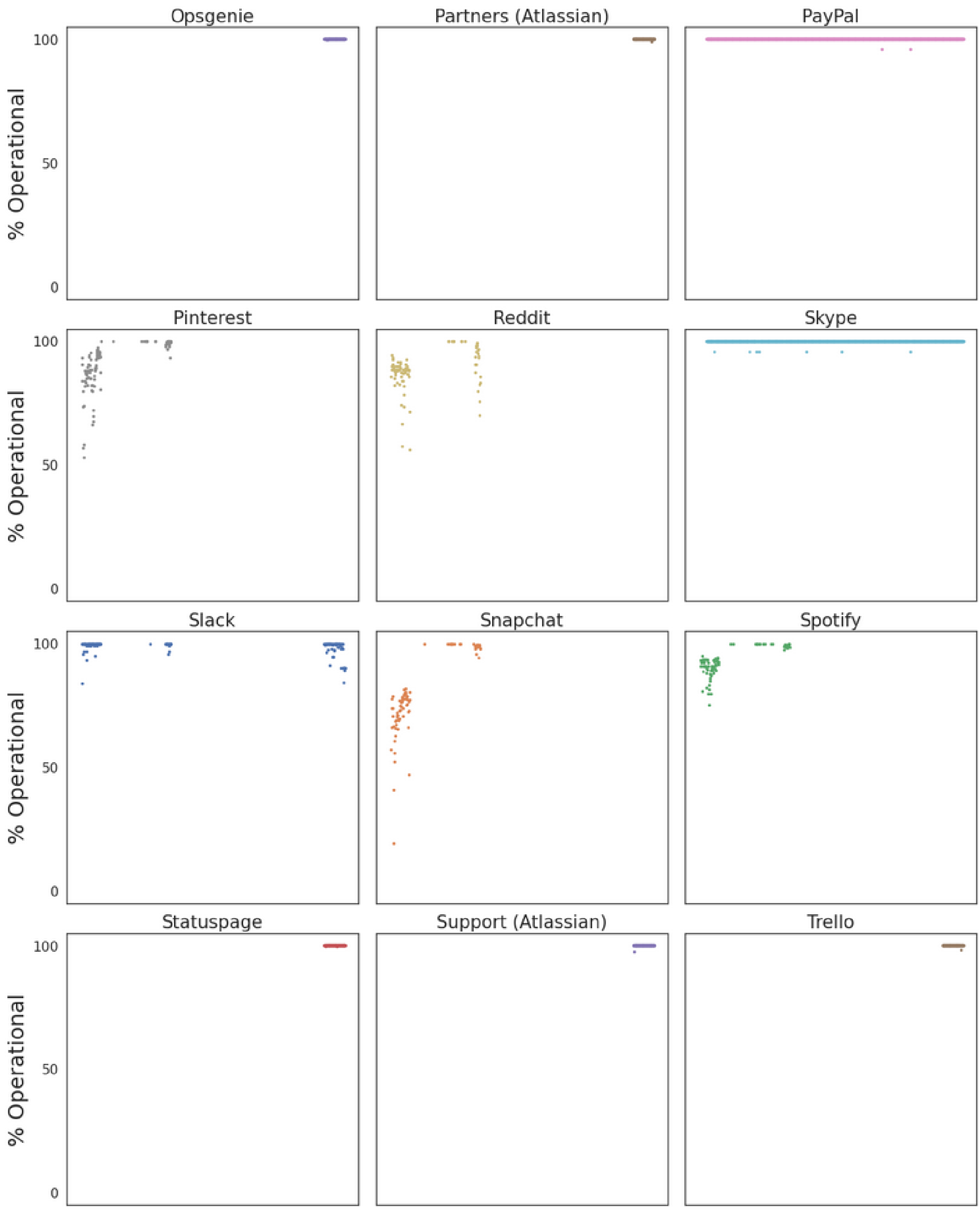
		All Reports	Self-Reported	User-Reported
Mean	Operational	95.796	97.902	95.415
	Partial Outages	4.160	2.074	4.537
	Major Outages	0.042	0.014	0.047
	Maintenance Events	0.002	0.011	-
Std	Operational	17.678	7.986	18.883
	Partial Outages	17.619	7.985	18.818
	Major Outages	0.675	0.203	0.728
	Maintenance Events	0.091	0.232	-

Table 4.2: Means and standard deviations of number of total reports versus self-reported and user-reported statuses.

Describing general distributions does not provide insight into the health of each individual cloud service. Figure 4.2 plots the daily operational averages for each of the 42 cloud services in our unified dataset over time. Many of the cloud services reside in a near 100% operational state, although there are exceptions. For example, two services that stand out visually are LiveJournal and Twitter. The operational status of LiveJournal hovers near 0% for more than the first half of the reported dates, but then jumps close to 100% for the remainder. Twitter seems to suffer from operational issues for roughly half a year, though this is not as pronounced as with LiveJournal. We also notice some plots containing clusters of points in the early months that do not follow later trends. We come up with two possible explanations for the appearance of these clusters. First, while most of the cloud service status data is collected at one-hour intervals, some were collected every five minutes (and these were scraped early on in the data collection process). Although we take daily averages, a greater number of observations may affect the mean values. Second, some cloud services contain status data from multiple sources, and we could be observing the nuances between their status reporting methods.


Access (Atlassian)
Airbnb
Amazon
Bitbucket
Blogger
Cloudflare
Confluence
Developers (Atlassian)
Discord
Facebook
Foursquare
GitHub
% Operational
100
50
0


Gmail
Google Cloud
Hotmail
Jira Align
Jira Core
Jira Service Desk
Jira Software
LinkedIn
LiveJournal
Lyft
Netflix
Ning
% Operational
100
50
0


Opsgenie
Partners (Atlassian)
PayPal
Pinterest
Reddit
Skype
Slack
Snapchat
Spotify
Statuspage
Support (Atlassian)
Trello
% Operational
100
50
0

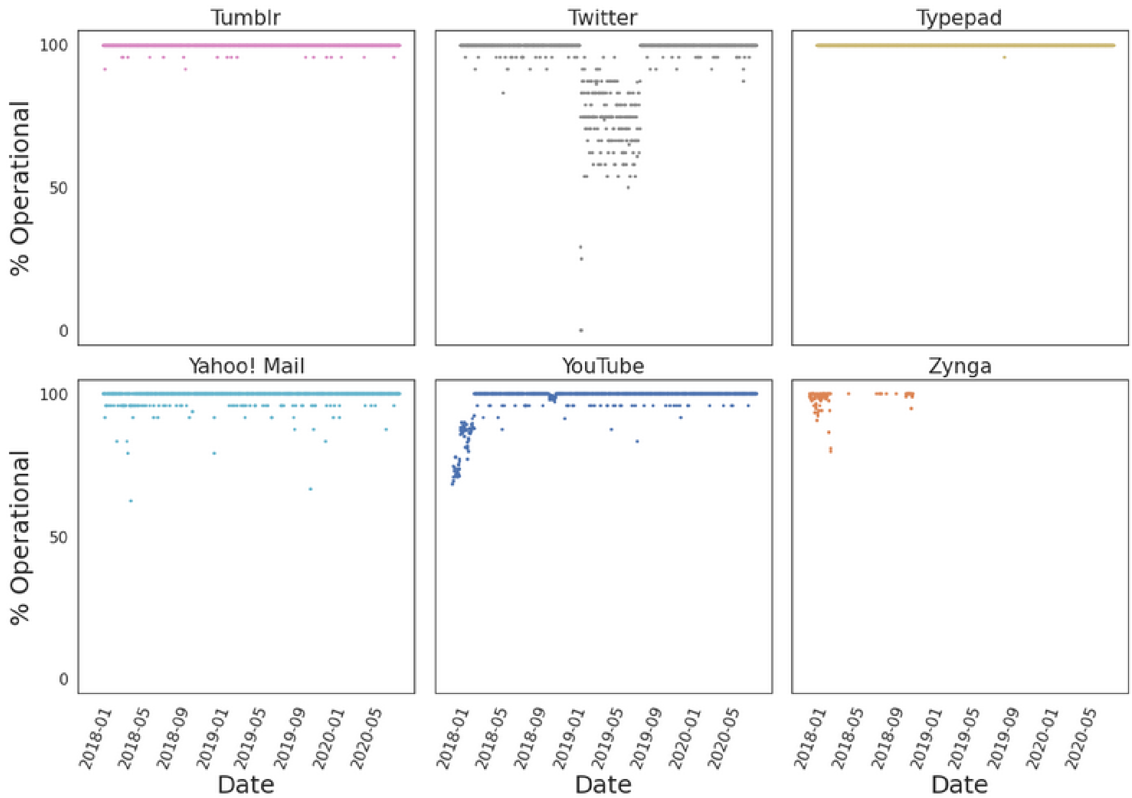


Figure 4.2: Daily operational averages of cloud services.

Table 4.3 lists the percentage that each cloud service resides in a given category. We compare these values to the raw percentages shown in Table 4.1. 37 of the 42 cloud services (88.1%) meet or exceed the average value of the operational category. The remaining five cloud services that fall below the operational state average are: Amazon, LiveJournal, Pinterest, Reddit, and Snapchat. These services are also the only ones that perform worse than average with respect to partial outages. Reddit and Snapchat perform worse than average for major outages. Bitbucket, Confluence, Developers, Partners, and Support (all Atlassian services) fair worse than average for maintenance events. These findings provide insight into the health of each cloud service. The sections that follow dive deeper into the behaviors of the cloud services individually. We investigate the distribution of means for each state, determine the mean time between failure events, and estimate the mean time to repair them.

4.2 Distribution of means

Analyzing the distribution of means for each category using an Empirical Cumulative Distribution Function (ECDF) provides insight into the behavior of the individual cloud services. The ECDF determines the proportion of observations that are less than a given value (22). In our case, this value is the mean percentage that a cloud service lives in each category. The mean percentage is a normalized value based on the number of statuses that fall into the four predefined categories: **(1)** operational, **(2)** partial outages, **(3)** major outages, and **(4)** maintenance events.

Many of the cloud services we analyze are shown to be highly operational, with 28 out of 42 having greater than than 99% of its status reports in this category. ECDF plots for these cloud services do not provide meaningful insight, since the number of observations are low for partial outages, major outages, and maintenance events. In Figure 4.3, we display eight ECDF plots for cloud services that exhibit a higher number of failure events (the remaining plots are shown in Appendix 6.1). The eight cloud services are: Amazon, Cloudflare, Livejournal, Netflix, Pinterest, Reddit, Snapchat, and Spotify. LiveJournal is a special case where nearly 60% of observations are partial outages with the rest being fully operational, which is clearly displayed in its corresponding ECDF plot. We identify the following trends for the remaining cloud services:

1. Mean operational percentages sharply increase at values around 90% and cover a large proportion of the data. This signifies that the cloud services we analyze are highly operational state, which support our previous findings. This trend exists across all cloud services, with the exception of LiveJournal.

2. If a cloud service exhibits a large number of partial outages, they tend to spike at a mean percentage of around 10%, and cover an approximated 80% proportion of the data.

3. Major outages and maintenance events are rare.

At this point we have reiterated that the majority of cloud services are highly operational, with a few exceptions. In the upcoming sections we focus on failures specifically, both overall and per cloud service.

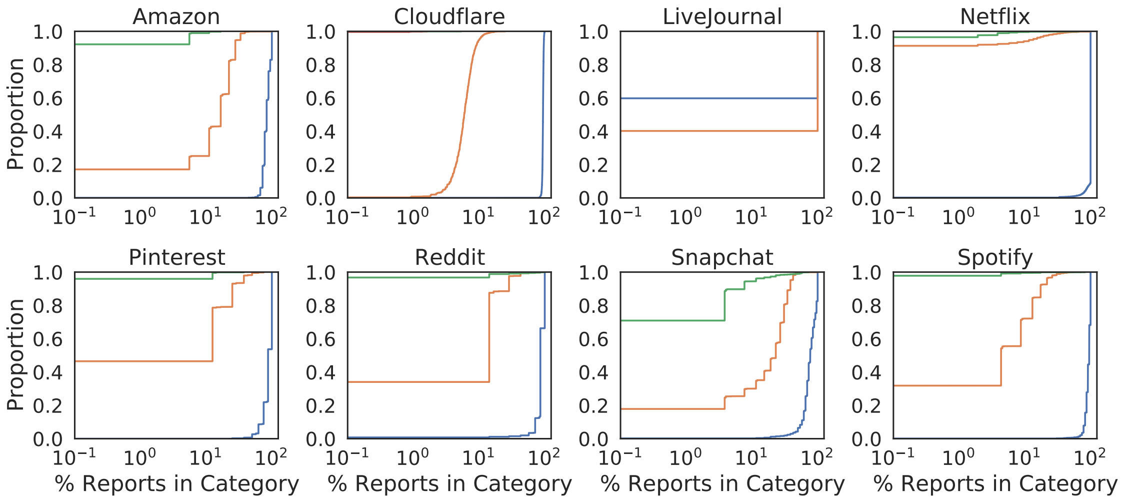


Figure 4.3: ECDF plots representing the proportion of status reports in a given category. blue = operational, orange = partial outages, green = major outages, red = maintenance events.

4.3 Mean time between failures

A failure is defined as a partial outage, major outage, or maintenance event. These types of failures make up 4.482% of all events in our dataset. The overall mean time between failures (MTBF) for the 42 cloud services are 22.59 days for partial outages, 9.55 days for major outages, and 22.48 days for maintenance events. The MTBF distributions of each failure category are shown as box plots in Figure 4.4. User-reported sources have a much higher range for partial outages which also include outliers, although their interquartile ranges are comparable between the two. One might expect a shorter MTBF for partial outages compared to major outages, since partial outages are not as severe; however, four outliers exist for partial outages: 85.31 days for LinkedIn, 91.14 days for Blogger, 103.25 days for PayPal, and 107.90 days for Skype. Major outages contain two outliers, namely 36.02 days for Cloudflare and 36.62 days for GitHub. It is difficult to determine the significance of these outliers since the 42 cloud services we analyze represent a very small portion of all cloud services. Another issue with performing a general MTBF analysis is that failures between cloud services are independent from one another. Thus, it is necessary and more meaningful to dive deeper into the MTBF of each cloud service individually.

Calculating the MTBF for individual cloud services requires recording the moments in time where we observe an increase in the number of failures of a specific type, which indicates the start of a failure event. We do not consider failure events that span time gaps

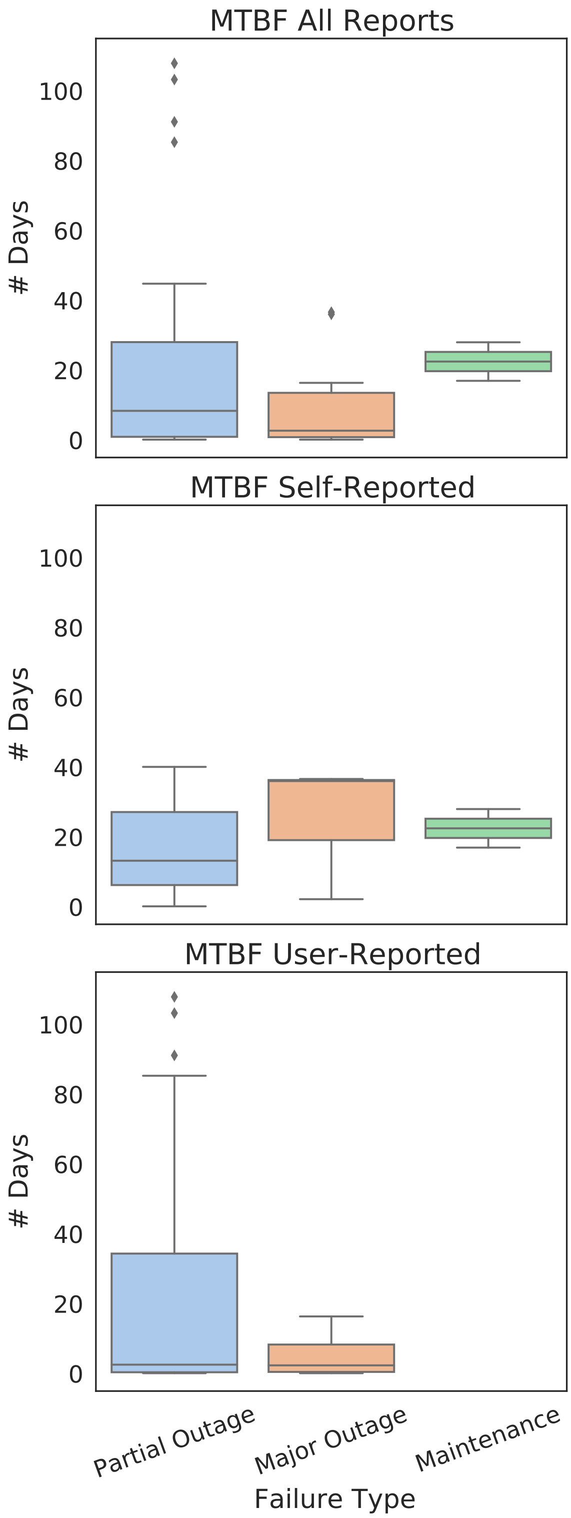


Figure 4.4: Box-plots: mean time between failures (MTBF).

where data has not been collected or missing for more than two days, and this phenomenon occurs only four times in total. Time-deltas are calculated between the start of failure events in sequential pairs, providing the amount of time between beginning of each failure event. The MTBF for each cloud service is the average of these values and are displayed in Table 4.5. Some cloud services have no MTBF value for one or more failure types, indicating that the cloud service never suffers from that type of failure event (maintenance events do not exist for user-reported sources of cloud service status data). If we exclude the four outliers for partial outages and the two for major outages, we find that 12/42 (28.57%) of the cloud services fall below the overall MTBF for partial outages, 5/42 (11.90%) for major outages, and 1/14 (7.14%) for maintenance events. Furthermore, the pair-wise standard correlation coefficient for the MTBF of partial and major outages produces a value of 0.440, signifying a moderate degree of correlation between them (23).

We note several observations when comparing self-reported versus user-reported sources of cloud service status data shown in Table 4.4. Major outages occur much more frequently for user-reported sources, and we offer possible explanations for why this *might* be misleading: **(1)** we reiterate that user-reported sources make up nearly 85% of all status reports, and thus we may not have an adequate sample size of self-reported data, **(2)** users who report an issue with a cloud service may over-exaggerate its severity, **(3)** sources may have a bias with their failure classification method or algorithm, **(4)** the cloud service providers between self-reported and user-reported sources do not intersect.

4.4 Mean time to repair

The average duration between the start of a failure event to its reparation is called the mean time to repair, or MTTR. When the number of failure status reports decreases for a cloud service, we consider it a reparation event. For the 42 cloud services in our dataset, we find a 5.41 hour MTTR for partial outages, a 6.80 hour MTTR for major outages, and a 2.23 hour MTTR for maintenance events. The distributions are shown in Figure 4.5 and the MTTR for each individual cloud service is listed in Table 4.6. The MTTR we find for partial and major outage events is similar to the 5.56 hour MTTR reported in a study on over 12,000 public cloud servers (11). Maintenance events are resolved at least twice as quickly on average versus outage events, which is probably due to most maintenance events being *planned events*.

We also investigate the differences in the MTTR between self-reported and user-reported sources of cloud service status data, which are shown in Table 4.7. The mean and standard

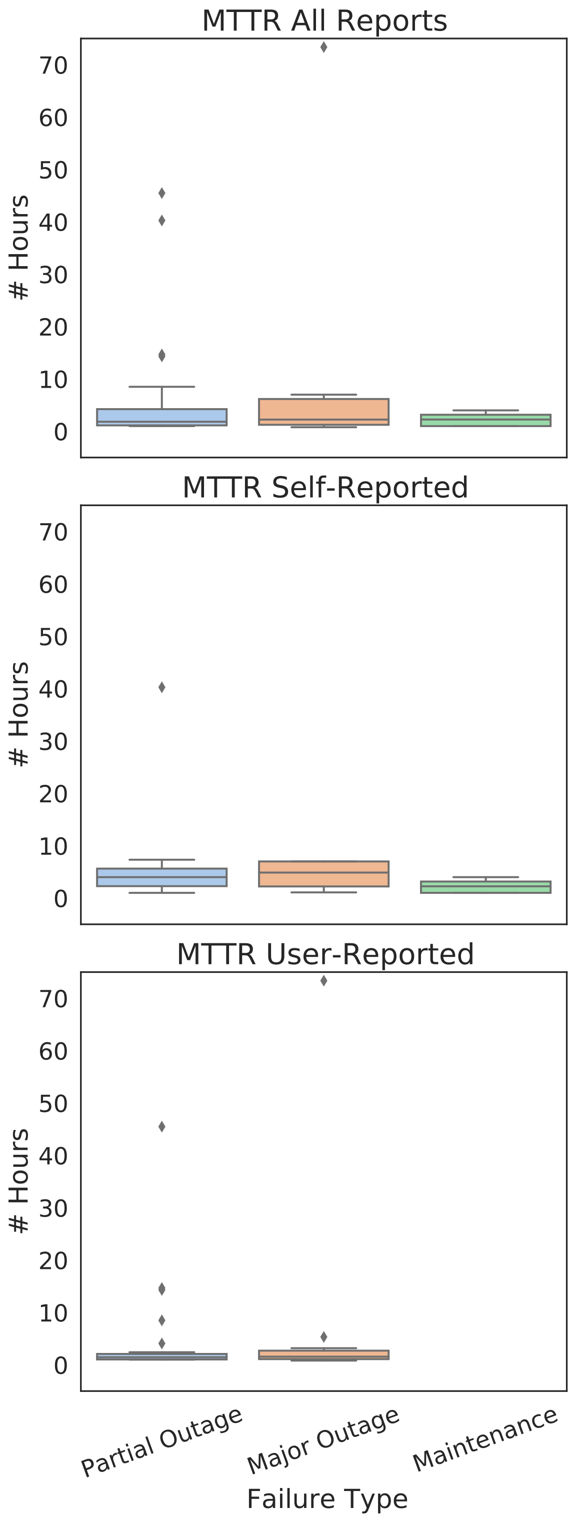


Figure 4.5: Box-plots: mean time to repair cloud service failures (MTTR).

deviations for partial outages are relatively similar, although they are both slightly higher for self-reported sources. In contrast, the respective mean and standard deviation for user-reported major outages are 1.87 and 8.01 times greater than its counterpart. This could imply that one of the types of sources tends to exaggerate their repair times, but this is difficult to determine due to the lack of ground truth in the data.

It is important to note that the reported MTTR values are an estimation of the true MTTR values. During the data unification process we transform multiple status reports into a single normalized value representing the global percentage of status reports for a given failure type. In this step we lose the one-to-one mappings of specific failure events to their resolution. In our case, the normalized values we calculate provide an indication of the general health of a cloud service, and are thus an indication of its true MTTR. In addition, none of the cloud service providers obtain an MTTR value below one hour in any state. The reason for this is that the vast majority of cloud service failure data is collected at hourly intervals. It is possible that the MTTR values reported at one hour intervals are lower in reality, and there are 13 cloud services that have this one hour value for at least one of the types.

Service	Operational	Partial Outage	Major Outage	Maintenance
Access (Atlassian)	100.00	0.00	0.00	0.00
Airbnb	98.94	1.01	0.05	0.00
Amazon	86.13	13.44	0.43	0.00
Bitbucket	99.48	0.17	0.08	0.27
Blogger	99.98	0.02	0.00	0.00
Cloudflare	93.82	6.18	0.00	0.00
Confluence	99.46	0.45	0.03	0.05
Developers (Atlassian)	99.88	0.07	0.04	0.01
Discord	99.97	0.03	0.00	0.00
Facebook	99.48	0.52	0.00	0.00
Foursquare	100.00	0.00	0.00	0.00
GitHub	99.70	0.27	0.03	0.00
Gmail	99.86	0.14	0.00	0.00
Google Cloud	96.45	3.55	0.00	0.00
Hotmail	99.76	0.24	0.00	0.00
Jira Align	100.00	0.00	0.00	0.00
Jira Core	99.84	0.11	0.04	0.00
Jira Service Desk	99.41	0.51	0.08	0.00
Jira Software	99.83	0.13	0.04	0.00
LinkedIn	99.97	0.03	0.00	0.00
LiveJournal	40.18	59.82	0.00	0.00
Lyft	99.53	0.42	0.05	0.00
Netflix	98.07	1.68	0.25	0.00
Ning	100.00	0.00	0.00	0.00
Opsgenie	100.00	0.00	0.00	0.00
Partners (Atlassian)	99.99	0.00	0.00	0.01
PayPal	99.99	0.01	0.00	0.00
Pinterest	89.99	9.50	0.51	0.00
Reddit	88.36	10.73	0.91	0.00
Skype	99.97	0.03	0.00	0.00
Slack	98.78	1.16	0.06	0.00
Snapchat	80.47	17.26	2.26	0.00
Spotify	93.25	6.55	0.20	0.00
Statuspage	100.00	0.00	0.00	0.00
Support (Atlassian)	99.97	0.00	0.00	0.03
Trello	99.98	0.02	0.00	0.00
Tumblr	99.91	0.09	0.00	0.00
Twitter	94.32	5.68	0.00	0.00
Typepad	100.00	0.00	0.00	0.00
Yahoo! Mail	99.33	0.67	0.00	0.00
YouTube	98.43	1.53	0.04	0.00
Zynga	98.37	1.42	0.21	0.00

Table 4.3: Percentage of status reports per category for each cloud service. Red indicates values that are worse than the raw mean for that category.

		All Reports	Self-Reported	User-Reported
Mean	Partial Outages	22.594	16.303	25.268
	Major Outages	9.552	24.943	4.935
	Maintenance Events	22.480	22.480	-
Std	Partial Outages	31.472	12.488	37.908
	Major Outages	13.012	19.707	6.065
	Maintenance Events	7.806	7.806	-

Table 4.4: MTBF statistics of all reports versus self-reported and user-reported statuses, in days.

Service	Partial Outage	Major Outage	Maintenance Event
Access (Atlassian)	-	-	-
Airbnb	1.06	13.52	-
Amazon	0.14	0.83	-
Bitbucket	6.28	2.19	16.96
Blogger	91.14	-	-
Cloudflare	0.14	36.02	28.00
Confluence	15.95	-	-
Developers (Atlassian)	22.97	-	-
Discord	10.46	-	-
Facebook	11.43	-	-
Foursquare	-	-	-
GitHub	7.52	36.62	-
Gmail	34.38	-	-
Google Cloud	6.09	-	-
Hotmail	28.14	-	-
Jira Align	-	-	-
Jira Core	27.15	-	-
Jira Service Desk	28.02	-	-
Jira Software	27.15	-	-
LinkedIn	85.31	-	-
LiveJournal	-	-	-
Lyft	8.26	-	-
Netflix	0.39	0.12	-
Ning	-	-	-
Opsgenie	-	-	-
Partners (Atlassian)	-	-	-
PayPal	103.25	-	-
Pinterest	0.19	2.95	-
Reddit	0.23	2.68	-
Skype	107.90	-	-
Slack	3.15	16.38	-
Snapchat	0.12	0.33	-
Spotify	0.14	2.05	-
Statuspage	40.08	-	-
Support (Atlassian)	-	-	-
Trello	-	-	-
Tumblr	44.78	-	-
Twitter	1.01	-	-
Typepad	-	-	-
Yahoo! Mail	8.47	-	-
YouTube	0.65	0.37	-
Zynga	1.06	10.12	-

Table 4.5: Mean time between failures (in days) for each cloud service. Red indicates values that are worse than the mean for that failure type.

Service	Partial Outage	Major Outage	Maintenance Event
Access (Atlassian)	-	-	-
Airbnb	1.16	<1.00	-
Amazon	14.31	2.24	-
Bitbucket	3.08	2.75	2.25
Blogger	<1.00	-	-
Cloudflare	40.26	2.30	2.33
Confluence	4.67	7.00	<1.00
Developers (Atlassian)	5.00	2.00	4.00
Discord	1.80	-	-
Facebook	1.50	-	-
Foursquare	<1.00	-	-
GitHub	2.28	1.09	-
Gmail	1.24	-	-
Google Cloud	5.63	-	-
Hotmail	1.86	-	-
Jira Align	-	-	-
Jira Core	3.33	7.00	-
Jira Service Desk	7.33	7.00	-
Jira Software	4.00	7.00	-
LinkedIn	1.17	-	-
LiveJournal	<1.00	-	-
Lyft	1.80	<1.00	-
Netflix	1.95	0.78	-
Ning	-	-	-
Opsgenie	-	-	<1.00
Partners (Atlassian)	-	-	<1.00
PayPal	<1.00	-	-
Pinterest	4.09	3.19	-
Reddit	45.49	73.36	-
Skype	<1.00	-	-
Slack	4.02	1.20	-
Snapchat	14.68	5.34	-
Spotify	8.50	2.03	-
Statuspage	<1.00	-	-
Support (Atlassian)	-	-	4.00
Trello	<1.00	-	-
Tumblr	1.05	-	-
Twitter	1.46	-	-
Typepad	<1.00	-	-
Yahoo! Mail	1.37	-	-
YouTube	2.41	1.29	-
Zynga	1.47	1.57	-

Table 4.6: Mean time to repair (in hours) for each cloud service. Red indicates values that are worse than the mean for that failure type.

		All Reports	Self-Reported	User-Reported
Mean	Partial Outages	5.414	6.634	4.719
	Major Outages	6.797	4.518	8.463
	Maintenance Events	2.226	2.226	-
Std	Partial Outages	9.821	10.308	9.517
	Major Outages	16.287	2.693	21.564
	Maintenance Events	1.343	1.343	-

Table 4.7: MTTR statistics of all reports versus self-reported and user-reported statuses, in hours.

5

Discussion

In this research paper we explored a data archive containing cloud service status data from multiple sources and providers. The status reports present in the dataset were either provided by the cloud service provider (self-reported) or via crowdsourcing (user-reported). The workload was split into two main areas: combining all of the data into a single and easy-to-use dataset and then extracting relevant information and statistics from said dataset. Combining the data into a single dataset was achieved by parsing the data for cloud service statuses, categorizing these statuses into one of four general types, and normalizing the resulting data. The statistics showed that roughly 94% of status reports are non-failures and the remaining reports are mostly partial outages (i.e. major outages and maintenance events are rare). For both the mean time between failures and the mean time to repair, user-reported sources showed favorable results for partial outages and significantly unfavorable results for major outages.

The findings of this study are relevant for individuals or companies that want to understand the behavior of cloud services with respect to the severity of the failure. As far as we are aware, this is the first study of its kind that categorizes cloud statuses into multiple failure types. In this case, these failure types are partial outages, major outages, and maintenance events. Most other studies we have encountered consider all failures as equivalent, but our research enables us to gain insight into the severity level of cloud failures. In addition, we compared the types of failures reported by cloud service providers themselves to user-reported sources, which is also something we have not come across in other literature.

5.1 Study limitations

The first limitation we encountered was that not all sources were easily addable to the single unified dataset that we end up compiling. For our intents and purposes, sources should report an already *qualified* status of cloud services such that we are able to classify the status accordingly. Some sources report on the status of a cloud service in a *quantified* manner. For example, the source Outage.Report counts the number of users that have reported a problem with a cloud service. This makes it difficult to draw conclusions regarding the severity of the problem, which is the focus of our research. Furthermore, some sources provide data with a complexity level that is either too difficult, time-consuming, or impossible to properly incorporate into the final dataset.

The second limitation relates to the composition of the raw data available for analysis. This is especially true for the ratio of user-reported and self-reported cloud service status data, of which the vast majority resides as user-reported data. Furthermore, the intersection of cloud service providers between self-reported and user-reported sources should be larger so that comparisons can be drawn between the two; however, this is dependent on whether a cloud service provider makes this information public.

A third limitation regards the normalization process used to construct the single unified dataset. During this process we end up losing the one-to-one mapping of a failure event to its reparation, and thus the values we report for the mean time between failure and the mean time to repair become *estimations* of their true values. Nonetheless, we justify our decision to use the normalized data since we are concerned with the general health of a cloud service (i.e. we are not concerned with *which* geographical regions or sub-services are down at any given time, we are only concerned with *how many*).

5.2 Future work

The data in the archive we analyze is being collected on a continuous basis, which presents several opportunities for future research. The first area of expansion could be to update the single unified dataset we compile in real-time and build a pipeline to automate the processing of the data. This could be achieved by creating, for example, a *crontab* (24) entry that executes one or more scripts at a set time. The immediate benefit is that the manual labor for processing the data would be eliminated. Since each source of cloud service status data provide reports in a unique manner, there would need to be a method to identify what the source is to ensure it reaches the appropriate parser. Through this automation process we would no longer need to manually run the parsing, encoding, and validation steps. Re-compiling the dataset would also no longer be required, and instead the newly processed data can be appended to the existing dataset.

Another area for future work is to expand on the number and variety of cloud services and the sources that are collected from. We identify several promising candidates in sections 3.4 and 3.5. A parser must be created for each new source of cloud service status data, but not for each cloud services being reported on for a single source. This makes it tempting to add sources of user-reported data over self-reported, as a single parser covers a wide variety of cloud services. On the other hand, a parser must be created for each source of self-reported data, which is more labor intensive, but necessary in order draw comparisons between the two types of data reporting methods. In addition, creating a greater intersection of the cloud services between user-reported and self-reported sources would allow us to more accurately identify the biases in their reporting methods, if existent.

6

Conclusion

In this study we analyze data from a cloud failures archive. This archive contains operational status information for many cloud services reported by multiple sources in the form of HTML documents. There are two main sources of status information: those provided by cloud service provider themselves, and those consisting of user-reported data (i.e. crowdsourcing). In section 3.1 we present four research questions, which are subsequently answered in chapters 3 and 4.

RQ.1 How do we combine cloud failure data from various sources into a single, uniform dataset?

We parse the HTML documents to extract relevant cloud service status information, which is then classified and encoded into one of four general categories: **(1)** operational, **(2)** partial outage, **(3)** major outage, or **(4)** maintenance event. This is detailed in chapter 3. Some sources break down statuses into sub-categories, such as by geographical region, and for these sources we normalize the number of occurrences by computing the mean values. This provides us with an indication of the global status of a cloud service. The normalization process allows us to combine the cloud service status information from multiple sources into a single uniform dataset.

RQ.2 What is a good process for selecting cloud services and their sources of status information and how does the current selection compare to this process?

Our research provides a reasoning for the selection of cloud services and their sources of status information, which is detailed in chapter 3. We use five website ranking services that determine the popularity of cloud services. We find that 54.34% of the cloud services in our dataset fall into the top 500 overall websites, and 45.65% fall into the top 50 for at least one category (e.g. finance or games). The popularity of the sources of cloud service status information is also provided. In addition, we list other potential cloud services and sources of their status information for future work.

RQ.3 What are the statistical properties of cloud service failures in relation to their type, the mean time between failures, and the mean time to repair them?
In chapter 4 we find that cloud services live overwhelmingly in an operational state. After normalization we find that 95.52% of all status reports are operational, 4.36% are partial outages, 0.12% are major outages, and 0.002% are maintenance events. The mean time between failures (MTBF) are 23.59, 9.55, and 22.48 days for partial outages, major outages, and maintenance events. Following the same sequence of failure types, we find the mean time to repair (MTTR) to be 5.41, 6.80, and 2.23 hours. We encountered high variance with our failure metrics, especially with respect to the MTBF results, which is a reoccurring theme in other research. We also found our MTTR values to be similar to those of other studies.

RQ.4 How do self-reported sources of cloud service status data compare to user-reported sources?
We conclude that this is difficult to determine in chapter 4. Failures tend to occur more frequently with user-reported data, but there could be several explanations for this: **(1)** approximately 85% of the dataset is user-reported, **(2)** users who report an issue with a cloud service may over-exaggerate its severity, **(3)** sources may have a bias with their failure classification method or algorithm, and **(4)** there is too small of an intersection between self-reported and user-reported sources. For the mean time to repair failures, we find that the mean and standard deviations for partial outages are fairly similar for user- and self-reported statuses; however, user-reported major outages values are much higher, which might indicate that one of the sources is exaggerating their status reports.

Finally, the 42 cloud services we analyze represent a small proportion of all cloud services, and thus more work is necessary to determine whether our results are consistent with a larger sample size.

Appendix

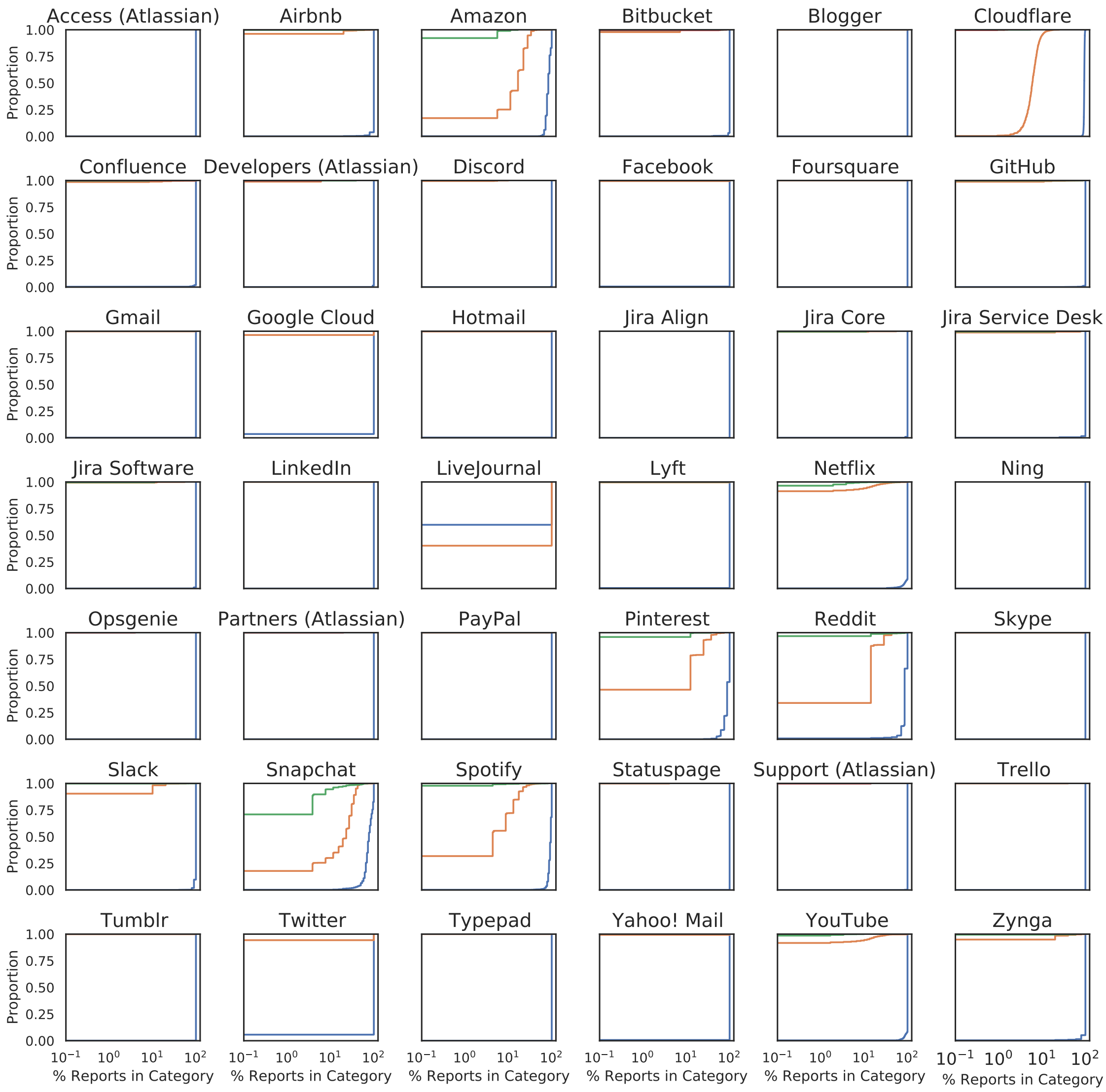


Figure 6.1: ECDF plots representing the proportion of status reports in a given category. blue = operational, orange = partial outages, green = major outages, red = maintenance events.

References

[1] **What is a Cloud Service – Cloud Service Definition**. https://www.citrix.com/solutions/digital-workspace/what-is-a-cloud-service.html, 2021. Accessed: 26-06-2021. 1

[2] **26 Cloud Computing Statistics, Facts Trends for 2022**. https://www.cloudwards.net/cloud-computing-statistics, 6 2022. Accessed: 16-08-2022. 1

[3] **Worldwide Public Cloud Services Revenues Grew 29.0% to $408.6 Billion in 2021, According to IDC**. https://marketresearchmedia.com/worldwide-public-cloud-services-revenues-grew-29-0-to-408-6-billion-in\-2021-according-to-idc/, 06 2022. Accessed: 16-08-2022. 1

[4] **The Societal Impact of Cloud**. https://deloitte.wsj.com/articles/the-societal-impact-of-cloud-1542247330, 11 2018. Accessed: 16-08-2022. 1

[5] Haryadi Gunawi, Mingzhe Hao, Riza Suminto, Agung Laksono, Anang Satria, Jeffry Adityatama, and Kurnia Eliazar. **Why Does the Cloud Stop Computing?: Lessons from Hundreds of Service Outages**. pages 1–16, 10 2016. 1, 8

[6] Sacheendra Talluri, Leon Overweel, Laurens Versluis, Animesh Trivedi, and Alexandru Iosup. **Empirical Characterization of User Reports on Cloud Failures**. 2021. 1

[7] R. Gracia-Tinedo, M. S. Artigas, A. Moreno-Martínez, C. Cotes, and P. G. López. **Actively Measuring Personal Cloud Storage**. In *2013 IEEE Sixth International Conference on Cloud Computing*, pages 301–308, 2013. 1

[8] Maurice Gagnaire, Felipe Diaz, Camille Coti, Christophe C, erin, Kazuhiko Shiozaki, Yingjie Xu, Pierre Delort, Jean-Paul Smets,

Jonathan Le, Lous, Stephen Lubiarz, and Pierrick Leclerc. **Downtime statistics of current cloud solutions**. In *Downtime statistics of current cloud solutions. International Working Group on Cloud Computing Resiliency, Technical report*, page 2, 2012. 1

[9] **Hacker News: AWS Cognito is having issues and health dashboards are still green**. `https://news.ycombinator.com/item?id=25209773`, 2020. Accessed: 27-06-2021. 2, 7

[10] **Most websites back online after global cloud outage temporarily breaks the Internet**. `https://www.france24.com/en/americas/20210608-most-websites-back-online-after-global-cloud-outage-\temporarily-breaks-the-internet`, 06 2021. 2, 7

[11] P. Garraghan, P. Townend, and J. Xu. **An Empirical Failure-Analysis of a Large-Scale Cloud Computing Environment**. In *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering*, pages 113–120, 2014. 3, 38

[12] **The 10 Biggest Cloud Outages Of 2021 (So Far)**. `https://www.crn.com/slide-shows/cloud/the-10-biggest-cloud-outages-of-2021-so-far-`, 07 2021. 7

[13] **Redundancy in Cloud Computing Means Checking Four Areas**. `https://sg.atsg.net/blog/redundancy-in-cloud-computing-means-checking-four-areas`, 07 2018. 7

[14] S. Minnema. **An Overview of the Cloud-Based Failure Landscape**. 2021. 7

[15] **Post-Incident Review on the Atlassian April 2022 outage**. `https://www.atlassian.com/engineering/post-incident-review-april-2022-outage`, 04 2022. 7

[16] **MTBF, MTTR, MTTF, MTTA: Understanding incident metrics**. `https://www.atlassian.com/incident-management/kpis/common-metrics`, 2021. Accessed: 18-08-2021. 8, 9

[17] P. Garraghan, P. Townend, and J. Xu. **An Empirical Failure-Analysis of a Large-Scale Cloud Computing Environment**. In *2014 IEEE 15th International Symposium on High-Assurance Systems Engineering*, pages 113–120, 2014. 8

[18] B. Javadi, J. Abawajy, and R. O. Sinnott. **Hybrid Cloud resource provisioning policy in the presence of resource failures**. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 10–17, 2012. 8

[19] **95 SEO Statistics From This Year That'll Transform Your Strategy**. https://www.webfx.com/seo/statistics/, 02 2022. 17

[20] Hofmann Heike, Hadley Wickham, and Karen Kafadar. **Letter-Value Plots: Boxplots for Large Data**. *Journal of Computational and Graphical Statistics*, **26**, 03 2017. 28

[21] Heike Hofmann, Hadley Wickham, and Karen Kafadar. **Letter-Value Plots: Boxplots for Large Data**. *Journal of Computational and Graphical Statistics*, **26**(3):469–477, 2017. 28

[22] **What, Why, and How to Read Empirical CDF**. https://towardsdatascience.com/what-why-and-how-to-read-empirical-cdf-123e2b922480, 08 2020. 35

[23] **Pearson's Correlation Coefficient**. https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/pearsons-correlation-coefficient/, 06 2021. 38

[24] **crontab(5) — Linux manual page**. https://man7.org/linux/man-pages/man5/crontab.5.html, 2021. Accessed: 07-02-2022. 47