A Large Scale Study of Data Center Network Reliability

Justin Meza
Carnegie Mellon University
Facebook, Inc.
jjm@fb.com

Tianyin Xu
University of Illinois Urbana-Champaign
Facebook, Inc.
tyxu@illinois.edu

Kaushik Veeraraghavan
Facebook, Inc.
kaushikv@fb.com

Onur Mutlu
ETH Zürich
Carnegie Mellon University
onur.mutlu@inf.ethz.ch

ABSTRACT

The ability to tolerate, remediate, and recover from network incidents (caused by device failures and fiber cuts, for example) is critical for building and operating highly-available web services. Achieving fault tolerance and failure preparedness requires system architects, software developers, and site operators to have a deep understanding of network reliability at scale, along with its implications on the software systems that run in data centers. Unfortunately, little has been reported on the reliability characteristics of large scale data center network infrastructure, let alone its impact on the availability of services powered by software running on that network infrastructure.

This paper fills the gap by presenting a large scale, longitudinal study of data center network reliability based on operational data collected from the production network infrastructure at Facebook, one of the largest web service providers in the world. Our study covers reliability characteristics of both intra and inter data center networks. For intra data center networks, we study seven years of operation data comprising thousands of network incidents across two different data center network designs, a cluster network design and a state-of-the-art fabric network design. For inter data center networks, we study eighteen months of recent repair tickets from the field to understand reliability of Wide Area Network (WAN) backbones. In contrast to prior work, we study the effects of network reliability on software systems, and how these reliability characteristics evolve over time. We discuss the implications of network reliability on the design, implementation, and operation of large scale data center systems and how it affects highly-available web services. We hope our study forms a foundation for understanding the reliability of large scale network infrastructure, and inspires new reliability solutions to network incidents.

KEYWORDS

data centers, networks, reliability, fault tolerance

IMC '18, October 31-November 2, 2018, Boston, MA, USA

ACM ISBN 978-1-4503-5619-0/18/10.
https://doi.org/10.1145/3278532.3278566

1 INTRODUCTION

Data center network infrastructure, consisting of both intra and inter data center networks, forms the cornerstone of large scale, geographically-distributed web services. In the last two decades, data center network infrastructure has evolved rapidly, driven by the scalability challenges of ever-increasing traffic demand and the resulting desire for ever-bigger and more plentiful data centers across the planet. New data center network technologies [2–4, 24, 35, 37, 38, 45, 50, 57, 62, 76] and WAN backbones [39, 43, 44, 47, 64] have been designed, deployed, and operated in the field. At the same time, both data center and backbone networks are transitioning from older *cluster network designs* with proprietary switches [24] toward newer *fabric network designs* [4] built from simple commodity hardware with automated repair mechanisms.

Despite the significance of network infrastructure for data center operation, we find that three aspects of data center network infrastructure are *not* well understood in the literature. First, little has been reported on the *overall* reliability characteristics of data center network infrastructure – from the networks *within* data centers, to the networks that connect *different* data centers. Second, no previous study that we are aware of has examined the reliability trends that appear with the transition away from cluster network designs toward fabric network designs [76]. Third, there is little discussion about how network design decisions affect software systems that run on the networks.

In the past, a lack of understanding hindered researchers and practitioners while combating network incidents [8]. Today, network incidents have become a major root cause of data center outages, as shown in recent studies [9, 28, 36, 65] and news reports [20, 23, 32, 33, 79]. For example, Gunawi et al. [36] studied 597 unplanned web and cloud service outages from headline news and public postmortem reports in a seven-year span from 2009 to 2015. Their study showed that network incidents contributed to 15% of these outages, forming the second largest root cause category. Therefore, understanding network reliability and its implications on software systems is of critical importance to the design, implementation, and operation of large scale data centers.

Prior studies examined data center networks and device reliability [30, 31, 34, 67, 68]. However, most of these studies focus on the failure characteristics of network devices and links, *without* connecting their impact to *software systems*. Network incidents often manifest as failures and anomalies in production software systems running on data center network infrastructure. In fact, all device- and link-level failures are *not* created equal – many failures are masked by hardware redundancy, path diversity, and other

fault-tolerance logic [51]. Compared to network device failures,[1] *network incidents*, i.e., misbehavior of the network as a whole that impairs the availability or quality of large scale web services, are among the least understood types of failures in the literature.

This paper fills the gap by presenting a large scale, longitudinal study of the reliability of data center network infrastructure. Our study is based on seven years of *intra* data center operation data and eighteen months of *inter* data center operation data collected from the production systems of Facebook, one of the largest web service providers in the world. Facebook serves 2.23 billion monthly users and operates twelve geographically distributed data centers [1] with multiple generations of data center network design. These data centers are interconnected via a *Wide Area Network (WAN)* backbone designed for serving different traffic types (§3 provides more detail on these traffic types).

We study the reliability characteristics of intra and inter data center networks from the perspective of *software system incidents in large scale systems*, and how these reliability characteristics evolve over time. Specifically, we discuss in depth how data center network reliability influences the design, implementation, and operation of large scale software systems that run highly-available web services. We make the following major observations:

- We observe that most failures that software cannot repair involve maintenance, faulty hardware, and misconfiguration. We also find 2× more human errors than hardware errors as devices and routing configurations become more complex and challenging to maintain (§5.1).
- Network devices with higher bandwidth have a higher likelihood of affecting software systems. Network devices built from commodity chips have much lower incident rates compared to devices from third-party vendors due to the devices' integration with automated failover and repair software. Rack switch incidents are increasing over time and are currently around 28% of all network incidents (§5.2).
- Although high bandwidth *core* network devices have the most incidents, the incidents they have are low severity. Fabric network devices cause incidents of lower severity than cluster network devices (§5.3).
- Cluster network incidents increased steadily over time until the adoption of fabric networks, with cluster networks currently having 2.8× the incidents compared to fabric networks (§5.4).
- While high reliability is essential for widely-deployed devices, such as rack switches, incident rates vary by three orders of magnitude across device types. Larger networks tend to have longer incident resolution times (§5.5).
- We develop models for the reliability of Facebook's WAN, which consists of a diverse set of edges and links that form a backbone. We find that time to failure and time to repair closely follow *exponential* functions. We provide models for these phenomena so that future studies can build on our models and use them to understand the nature of backbone failures (§6.1–§6.2).
- Backbone edge nodes that convey traffic between data centers fail on the order of months and recover on the order of hours. However, there is high variance in edge node failure rate and recovery rate. Path diversity in the backbone topology ensures that large scale networks can tolerate failures with long repair times (§6.1).
- Links supplied by backbone vendors typically fail on the order of months, with links in big cities failing less frequently. Both failure rate and recovery rate for links span multiple orders of magnitude among vendors (§6.2).
- Edge failure rate varies by months across continents in the world. Edges recover within 1 day on average on all continents (§6.3).

[1] We use "failures" to refer to any network device misbehavior. The root cause of a failure includes not only hardware faults, but also misconfigurations, maintenance mistakes, firmware bugs, and other issues.

2 MOTIVATION

The reliability of data center network infrastructure is critically important for building and operating highly available and scalable web services [8, 16]. Despite an abundance of device- and link-level monitoring, the effects of network infrastructure reliability on the software systems that run on them is not well understood. The fundamental problem lies in *the difficulty of correlating device- and link-level failures with software system impact*. First, many network failures do *not* cause software system issues due to network infrastructure redundancy (including device, path, and protocol redundancy). Second, large scale network infrastructure is typically equipped with automated repair mechanisms that take action to resolve failures when they occur.

To understand the behavior of network failures, we must be able to answer questions such as: *"How long do network failures affect software when they occur?"*, *"What are the root causes of the network failures that affect software?"*, and *"How do network failures manifest themselves in software systems?"*. Unfortunately, past efforts to understand network incidents in large scale network infrastructure are limited to informal surveys and a small number of public postmortem reports [8, 36], which could be biased toward certain types of failures and not comprehensive. As noted in [8], due to scant evidence and even less data, it is hard to discuss the reliability of software systems in the face of network incidents because *"much of what we believe about the failure modes is founded on guesswork and rumor."*

Even for large web and cloud service providers, understanding the reliability of network infrastructure is challenging, given the complex, dynamic, and heterogeneous nature of large scale networks. With complex and constantly evolving network designs built from a wide variety of devices, it is hard to reason about the end-to-end reliability of network infrastructure under different failure modes, let alone how the network affects the software that uses it. As far as we know, from what limited information has been publicly discussed, this is a common challenge across the industry.

Facebook has attempted to address this challenge by seeking to understand the reliability of its data center network infrastructure. At Facebook, software system events that affect reliability (known as *SEVs*) are rigorously documented and reviewed to uncover their root causes, duration, software system impact, as well as mitigation and recovery procedures [55]. These postmortem reports form an invaluable source of information for analyzing and understanding network reliability from the perspective of large scale web services.

Our goal is to shed light on the network reliability incidents, both *within* and *between* data centers, which affect the software systems that power large scale web services. We hope that our work helps researchers and practitioners anticipate and prepare for network incidents, and inspires new network reliability solutions.

3 FACEBOOK'S NETWORK ARCHITECTURE

Figure 1 shows Facebook's network architecture [4, 24, 44, 71]. Facebook's network consists of interconnected data center *regions*. Each region contains buildings called *data centers*. Facebook operates both data center and backbone networks. We call the network *within* data centers the *intra* data center network and the backbone network *between* data centers the *inter* data center network.

The diversity of Facebook's network provides an opportunity to compare reliability across different network designs. Though diverse, Facebook's network is by no means unique. Published network architectures from Google and Microsoft use similar design principles and building blocks [31, 34, 43, 67, 68, 76]. We expect that our findings and implications would apply to other large scale data center networks.

3.1 Intra Data Center Networks

Facebook uses two intra data center network designs: an older *cluster-based* design [19, 24] and a newer *data center fabric* design [4]. We call these the *cluster network* and the *fabric network*. Unlike the cluster network, the fabric network uses a five-stage Folded Clos [2] design, built from simple commodity hardware, with software-controlled automated repairs.

Cluster network design. In Facebook's older network (Figure 1, Region A), a *cluster* is the basic unit of network deployment. Each cluster comprises four *cluster switches (CSWs, ①)*, each of which aggregates physically contiguous *rack switches (RSW, ②)* via 10 Gb/s Ethernet links. In turn, a *cluster switch aggregator (CSA, ③)* aggregates CSWs and keeps inter cluster traffic within the data center. Inter data center traffic flows through *core network devices (core devices, ④)*, which aggregate CSAs.

A cluster network has two main limitations:

(1) ***Vendor devices*** limit data center scalability. Connecting more devices requires waiting for vendors to produce larger switches. This is a fundamental limiting factor for data center size in cluster networks.
(2) ***Proprietary software*** is challenging to maintain and customize. Proprietary software on switches makes customization difficult or impossible. Once deployed, proprietary switches must be repaired in-place. When a device becomes unresponsive, a human must power cycle the device. Compared to software, humans perform slow repairs. Slow repairs mean fewer switches to route requests, more traffic on the remaining switches, and more congestion in the network.

Despite its limitations, the cluster networks remain in use in a dwindling fraction of Facebook's data centers. Ultimately, these data centers will join new data centers in using the fabric network design.

Fabric network design. Facebook's newer network (Figure 1, Region B) addresses the cluster network's limitations. A *pod* is the basic unit of network deployment in a fabric network. Unlike the physically contiguous RSWs in a cluster, RSWs in a pod have no physical constraints within a data center. Each RSW (⑥) connects to four *fabric switches (FSWs, ⑦)*. The 1:4 ratio of RSWs to FSWs maintains the connectivity benefits of the cluster network. *Spine switches (SSWs, ⑧)* aggregate a dynamic number of FSWs, defined by software. Each SSW connects to a set of *edge switches (ESWs, ⑨)*. Core devices (⑩) connect ESWs between data centers.

Facebook's fabric networks are managed largely by software and differ from its cluster networks in four ways:

(1) ***Simple, custom switches.*** Fabric devices contain simple, commodity chips and eschew proprietary firmware and software.
(2) ***Fungible resources.*** Fabric devices are *not* connected in a strict hierarchy. Control software manages FSWs, SSWs, and ESWs as a fungible pool of resources. Resources dynamically expand or contract based on network bandwidth and reliability needs.
(3) ***Automated repair mechanisms.*** Failures on data center fabric devices can be repaired automatically by software [70]. Centralized management software continuously checks for device misbehavior. A skipped heartbeat or an inconsistent network setting raises alarms for management software to handle. Management software triages the problem and attempts to perform automated repairs. Repairs include restarting device interfaces, restarting the device itself, and deleting and restoring a device's persistent storage. If the repair fails, management software opens a support ticket for investigation by a human.
(4) ***Stacked devices.*** The same type of fabric device can be stacked in the same rack to create a higher bandwidth virtual device [26]. Stacking enables port density in fabric networks to scale faster than in proprietary network devices [5–7, 75].

Both cluster networks and fabric networks use *backbone routers (BBRs)* located in *edge nodes* (⑤) to communicate across the WAN backbone and Internet.

3.2 Inter Data Center Networks

Facebook's WAN backbone consists of *edge nodes* connected by *fiber links* (⑪ in Figure 1). Edge nodes are locations where Facebook deploys hardware to route backbone traffic. Fiber links are optical fibers that connect edge nodes, formed by optical *circuits* made of optical *segments*. An optical segment corresponds to a fiber optic cable and carries multiple channels, where each channel corresponds to a wavelength mapped to a router port.

Fiber link reliability is important to software systems that run in multiple data centers, especially those requiring consistency and high-availability [8, 16]. Without careful planning, fiber cuts (e.g., due to natural disasters) can partition entire data centers or regions from the rest of the network. At Facebook, we have not witnessed disastrous network partitions of data centers or regions. This is in part due to network planning decisions made from simulations using models like those presented in this paper. Common results of fiber cuts include lost capacity from edge nodes to regions or lost capacity between regions. In these cases, we reroute backbone traffic using other links, possibly with increased latency.

On top of the physical fiber-based backbone, multiple WAN backbone networks satisfy the distinct requirements of two types of traffic labeled in Figure 1:

(1) ***User-facing traffic*** (⑫ in Figure 1) connects a person using Facebook applications like those hosted at `facebook.com`, to software systems running in Facebook data centers. To reach a Facebook data center, user-facing traffic goes through the Internet via a *peering* [82] process. There, *Internet Service Providers (ISPs)* exchange traffic among Internet domains. User-facing traffic uses the *Domain Name System (DNS)* to connect users to geographically local servers operated by Facebook called *edge nodes* (also known as *points of presence*) [72, 82]. From edge

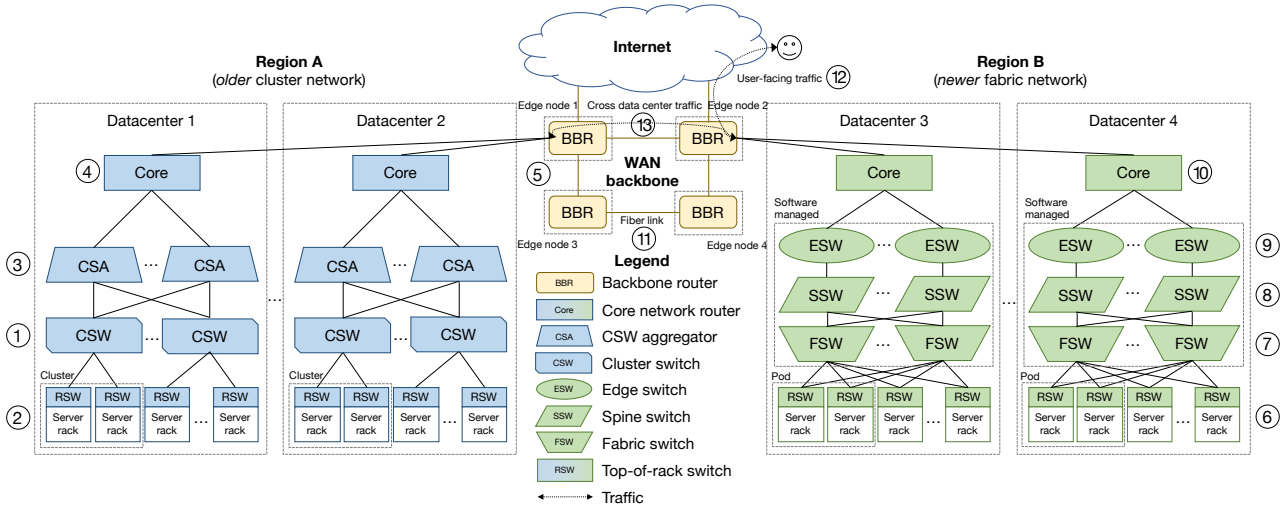

Figure 1: Facebook's network architecture, explained in §3. Data centers use either an older cluster network or a newer fabric network. Cluster networks and fabric networks communicate through the WAN backbone and Internet.

nodes, user traffic arrives at Facebook's data center regions through the backbone network.

(2) ***Cross data center traffic*** (⑬ in Figure 1) connects a software service in one Facebook data center to a software service in another Facebook data center. The backbone network interconnects both cluster networks and fabric networks. By volume, cross data center traffic consists primarily of *bulk data transfer* streams for replication and consistency. Bulk transfer streams are generated by *backend* services that perform batch processing [21, 53], distributed storage [10, 60], and real-time processing [18, 40].

To serve user-facing traffic, backbone networks support a range of protocols and standards to connect a variety of external networks from different ISPs. Facebook uses a traditional WAN backbone design consisting of backbone routers placed in every edge node (e.g., the BBRs in Edge 1 through 4 in the WAN backbone, ⑤ in Figure 1). In contrast, cross data center traffic is partitioned at the optical layer into four *planes* where each plane has one backbone router per data center [44]. Cross data center traffic is managed by software systems that route traffic between backbone routers built from commodity chips. The design is, in principle, similar to Google's B2 and B4 that are described in [34, 43].

4 BASE RESULTS AND METHODOLOGY

We describe how we measure and analyze the reliability of intra and inter data center networks, including the scope of our study (§4.1), our network incident dataset (§4.2), our analytical methodology (§4.3), and limitations and conflating factors in our study (§4.4).

4.1 Network Incidents and Automated Repairs

We call network failures that disrupt software systems *network incidents*. Network incidents affect software systems, causing data corruption, connection time outs, and excessive latency, for example. Software systems at Facebook include frontend web servers [22], cache systems [17, 63], storage systems [10, 60], data processing systems [18, 40], and real-time monitoring systems [46, 66].

Facebook shields software systems from common network failures with *automated repair software* [70]. Automated repair software prevents common network *failures* from causing network *incidents*. It runs on RSWs, FSWs, and core devices. We list automated repair software data from April 1 to May 1, 2018 in Table 1. During this time, automated repair software fixed 99.7% of RSW failures, 99.5% of FSW failures, and 75% of core device failures.[2]

Device	Repair Ratio	Avg Priority / Wait / Repair Time
Core	75%	0 (highest priority) / 4 m / 30.1 s
FSW	99.5%	2.25 / 3 d / 4.45 s
RSW	99.7%	2.22 / 1 d / 2.91 s

Table 1: The repair ratio (fraction of issues repaired with automated repair versus all issues), average priority (0 = highest, 3 = lowest), average wait time, and average repair time for the network device types that automated repair software supports.

Automated repair software schedules a repair based on its *priority*: low priority repairs wait longer than high priority repairs. Engineers assign repairs a priority from 3 (the lowest priority) to 0 (the highest priority). Core device repairs have the highest priority, and wait only minutes on average, because core devices connect data centers. FSW and RSW repairs have lower priorities on average, 2.25 and 2.22, respectively, and wait *days*. Repairs happen relatively fast once they run, taking less than a minute on average. Core device repairs take around 30.1 s on average; FSW and RSW repairs take around 4.45 s and 2.91 s on average, respectively.

If automated repair software cannot fix a device's failure, the software alerts a human technician to investigate the device. Four

[2] Automated repair software is less effective for core devices because many core devices run vendor software that is *incompatible* with automated repairs.

root causes constitute the top 90.9% of failures handled by automated repairs. (1) 50% of repairs fix device port ping failures by turning the port off and on again. (2) 32.4% of repairs fix configuration file backup failures by restarting the configuration service and reestablishing a secure shell connection. (3) 4.5% of repairs handle fan failures by extracting failure details and alerting a technician to examine the faulty fan. (4) 4.0% of repairs handle entire device ping failures by collecting device details and assigning a task to a technician.

We analyze the failures automated repair software *neither detects nor fixes*. By doing so, we hope to portray the non-trivial network failures that cause network incidents. We describe next our methodology for analyzing network incidents.

4.2 Site Events (SEVs)

Facebook engineers document incidents that affect software systems in reports called *Site Events (SEVs)*.[3] SEVs fall into three severity categories ranging from SEV3 (the lowest severity, no external outage) to SEV1 (the highest severity, widespread external outage). Engineers who responded to a SEV, or whose service the SEV affected, write the SEV's *report*. The report contains the incident's root cause, the root cause's effect on software systems, and steps to prevent the incident from happening again [55]. Each SEV goes through a review process to verify the accuracy and completeness of the report. SEV reports help engineers at Facebook prevent similar incidents from happening again.

We analyze a SEV dataset collected over seven years, from 2011 to 2018. The dataset comprises thousands of SEVs and resides in a MySQL database. Network SEV reports contain details on the network incident: the network device implicated in the incident, the duration of the incident (measured from when the root cause manifested until when engineers fixed the root cause), and the incident's effect on software systems (for example, increased load from lost capacity, message retries from corrupted packets, downtime from partitioned connectivity, and increased latency from congested links). We use SQL queries to analyze the SEV report dataset for our study.

SEVs come in many shapes and sizes. We summarize three representative example SEVs in increasing site event severity:

SEV3 ***Switch crash from software bug.*** A bug in the switching software triggered an RSW to crash whenever the software disabled a port. The incident occurred on August 17, 2017 at 11:52 am PDT after an engineer updated the software on a RSW and noticed the behavior. The engineer identified the root cause by reproducing the crash and debugging the software: an attempt to allocate a new hardware counter failed, triggering a hardware fault. On August 22, 2017 at 11:51 am PDT the engineer fixed the bug and confirmed the fix in production.

SEV2 ***Traffic drop from faulty hardware module.*** A faulty hardware module in a CSA caused traffic to drop on October 25, 2013 between 7:39 am PDT and 7:44 am PDT. After the drop, traffic shifted rapidly to alternate network devices. Web servers and cache servers, unable to handle the influx of load, exhausted their CPU and failed 2.4% of requests. Service resumed normally after five minutes when web servers and cache servers recovered. An on-site technician diagnosed the problem, replaced the faulty hardware module, verified the fix, and closed the SEV on October 26, 2013 at 8:22 am PDT.

SEV1 ***Data center outage from incorrect load balancing.*** A core device with an incorrectly configured load balancing policy caused a data center network outage on January 25, 2012 at 3:46 am PST. Following a software upgrade, a core device began routing traffic on a single path, overloading the ports associated with the path. The overload at the core device level caused a data center outage. Site reliability engineers detected the incident with alarms. Engineers working on the core device immediately attempted to downgrade the software. Despite the downgrade, core device load remained imbalanced. An engineer resolved the incident by manually resetting the load balancer and configuring a particular load balancer setting. The engineer closed the SEV on January 25, 2012 at 7:47 am PST.

4.3 Analytical Methodology

We use two sets of data for our study. For *intra* data center reliability, we examine seven years of service-level event data collected from the SEV database we discussed in §4.2. For *inter* data center reliability we use eighteen months of data collected from vendors on fiber repairs the vendors performed between October 2016 and April 2018. We describe the analysis for each data source below.

Intra data center networks. For intra data center reliability, we study the network incidents in three aspects:

(1) ***Root cause.*** We use the root causes chosen by the engineers who authored the corresponding SEV reports. The root cause category (listed in Table 2) is a mandatory field in our SEV authoring workflow, although the root cause may be *undetermined*.

(2) ***Device type.*** To classify a network incident by the implicated device's type, we rely on the naming convention enforced by Facebook where each network device is named with a unique, machine-understandable string prefixed with the device type. For example, every rack switch has a name prefixed with "`rsw`". By parsing the prefix of the name of the offending device, we are able to classify SEVs based on device type.

(3) ***Network design.*** We also classify network incidents based on network architecture. Recall from Figure 1 that CSA and CSW devices belong to cluster networks, while ESW, SSW, and FSW devices are a part of fabric networks.

Inter data center networks. For inter data center reliability, we study the reliability of edge nodes and fiber links based on repair tickets from fiber vendors whose links form Facebook's backbone networks that connect the data centers. Facebook has monitoring systems that check the health of every fiber link, as unavailability of the links could significantly affect the traffic or partition a data center from the rest of Facebook's infrastructure.

When a vendor starts repairing a link (when the link is severed) or performing maintenance on a fiber link, Facebook is notified via email. The email is in a structured form, including the logical IDs of the fiber link, the physical location of the affected fiber circuits, the starting time of the repair or maintenance, and the estimated duration. Similarly, when the vendor completes the repair or maintenance of a fiber link, the vendor sends a confirmation email. The emails are automatically parsed and stored in a database

[3] Pronounced [sɛv] in the *International Phonetic Alphabet* [42], rhyming with "rev."

for later analysis. We examine eighteen months of repair data in this database from October 2016 to April 2018. From this data, we measure fiber link mean time between failures (MTBF) and mean time to repair (MTTR).

4.4 Limitations and Conflating Factors

We found it challenging to control for all variables in a longitudinal study of failures at a company of Facebook's scale. So, our study has limitations and conflating factors, some of which we briefly discuss below. Throughout our analysis, we state when a factor that is *not* under our control may affect our conclusions.

- ***Absolute versus relative number of failures.*** We cannot report the absolute number of failures. Instead, we report failure rates using a fixed baseline when the trend of the absolute failures aids our discussion.
- ***Logged versus unlogged failures.*** Our intra data center network study relies on SEVs reported by employees. While Facebook fosters a culture of opening SEVs for all incidents affecting production, we cannot guarantee our incident dataset is exhaustive.
- ***Technology changes over time.*** Switch hardware consists of a variety of devices sourced and assembled from different vendors. We do not account for these factors in our study. Instead, we analyze trends by switch type when a switch's architecture significantly deviates from others.
- ***Switch maturity.*** Switch architectures vary in their lifecycle, from newly-introduced switches to switches ready for retirement. We do not differentiate the effect a switch's maturity has in Facebook's fleet in our analyses.
- ***More engineers making changes.*** As Facebook has grown, so has the number of engineers performing network operations. While all network software and configuration changes go through code review to reduce the chances of network incidents, more engineers can potentially lead to more opportunities for failure.

5 INTRA DATA CENTER RELIABILITY

In this section, we study the reliability of data center networks. We analyze network incidents within Facebook data centers over the course of seven years, from 2011 to 2018, comprising thousands of real world events. A network incident occurs when the root cause of a SEV relates to a network device. We analyze root causes (§5.1), incident rate and distribution (§5.2), incident severity (§5.3), network design (§5.4), and device reliability (§5.5).

5.1 Root Causes

Key Takeaways
- *Maintenance failures contribute the most documented incidents*
- *2× higher rate of human errors than hardware errors*

Table 2 lists network incident root causes.[4] If a SEV has *multiple* root causes, we count the SEV toward multiple categories. Human classification of root causes implies SEVs can be misclassified [56, 69]. While the rest of our analysis does *not* depend on the accuracy of root cause classification, we find it instructive to examine the types of root causes that occur in Facebook's networks.

We find the root cause of 29% of network incidents is *undetermined*. We observe these SEVs correspond typically to transient and isolated incidents where engineers only reported on the incident's symptoms. Wu et al. note a similar fraction of unknown issues (23%, [81], Table 1), while Turner et al. report a smaller fraction (5%, [80], Table 5).

Maintenance failures contribute the most documented root causes (17%). This suggests that in the network infrastructure of a large web service provider like Facebook, despite the best efforts to automate and standardize the maintenance procedures, maintenance failures still occur and lead to disruptions. Therefore, it is important to build mechanisms for quickly and reliably routing around faulty devices or devices under maintenance.

Hardware failures represent 13% of the root causes, while human-induced misconfiguration and software bugs occur at nearly double the rate (25%) of those caused by hardware failures. Turner et al. and Wu et al. observe hardware incident rates similar to the incident rates we observe (18% in Table 1 in [81] and 20% in Table 5 in [80]), suggesting that hardware incidents remain a fundamental root cause. Misconfiguration causes as many incidents as faulty hardware. This corroborates the findings of prior works that report misconfiguration as a large source of network failures in data centers [14, 15, 34, 49, 52, 67, 81], and shows the importance of emulation, verification, and automated repair techniques to reduce the number of incidents [11–13, 25, 27, 29, 48, 49].

We observe a similar rate of misconfiguration incidents (13%) as Turner et al. (9% in Table 5 in [80]), and a lower rate of misconfiguration incidents than Wu et al. (38% in Table 1 in [81]). We suspect network operators play a large role in determining how misconfiguration causes network incidents. At Facebook, for example, all configuration changes require code review and typically get tested on a small number of devices before being deployed to the fleet. These practices may contribute to the lower misconfiguration incident rate we observe compared to Wu et al..

[4] We use Govindan et al. [34]'s definition of *root cause*: *"A failure event's root-cause is one that, if it had not occurred, the failure event would not have manifested."*

Category	Fraction	Description
Maintenance	17%	Routine maintenance (for example, upgrading the software and firmware of network devices).
Hardware	13%	Failing devices (for example, faulty memory modules, processors, and ports).
Misconfiguration	13%	Incorrect or unintended configurations (for example, routing rules blocking production traffic).
Bug	12%	Logical errors in network device software or firmware.
Accidents	11%	Unintended actions (for example, disconnecting or power cycling the wrong network device).
Capacity planning	5%	High load due to insufficient capacity planning.
Undetermined	29%	Inconclusive root cause.

Table 2: Common root causes of intra data center network incidents at Facebook from 2011 to 2018.

A potpourri of accidents and capacity planning issues makes up the last 16% of incidents (cf. Table 2). This is a testament to the many sources of entropy in large scale production data center networks. Designing network devices to tolerate all of these issues is prohibitively difficult (if not impossible) in practice. Therefore, one reliability engineering principle is to *prepare for the unexpected* in large scale data center networks.

Figure 2 breaks down each root cause across the types of network devices it affects. Note that the major root cause categories, including *undetermined, maintenance, hardware, misconfiguration, bugs, accidents, and capacity planning* affect most network device types. Some root cause categories are represented unequally among devices. For example, capacity planning issues tend to affect more ESWs and maintenance issues tend to affect more FSWs. ESWs do *not* have SEVs due to bugs or maintenance, not because ESWs are immune to bugs and maintenance issues, but because the population size is small and such incidents have not yet been observed.

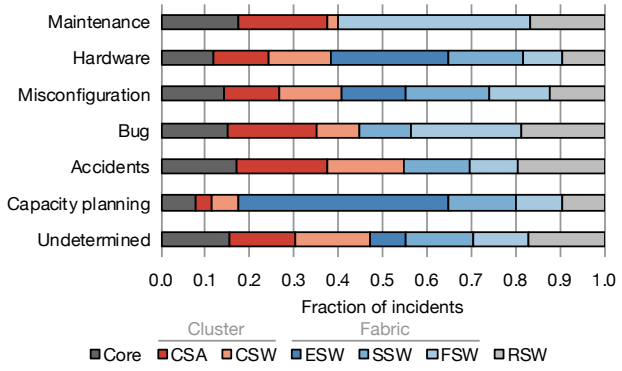

Figure 2: Breakdown of each root cause across the device types it affects.

We conclude that maintenance failures contribute the most documented network incidents (17%) and human-induced failures (misconfiguration and bugs) occur twice as much as hardware-induced failures.

5.2 Incident Rate and Distribution

Key Takeaways
- *Higher incident rates occur on higher bandwidth devices*
- *Lower incident rates occur on fabric network devices*
- *RSW incidents are increasing over time*

Incident rate. The overall reliability of data center networks is determined by the reliability of each interconnected network device. To measure the frequency of incidents related to each device type, we define *incident rate* of a device type as $r = \frac{i}{n}$, where i denotes the number of incidents caused by this type of network device and n is the number of active devices in the network of that type (the *population*). Note that the incident rate could be larger than 1.0, meaning that each device of that type caused more than one network incident, on average.

Figure 3 shows the incident rate of each type of network device in Facebook's data centers over the seven-year span of our study. From Figure 3, we make four observations:

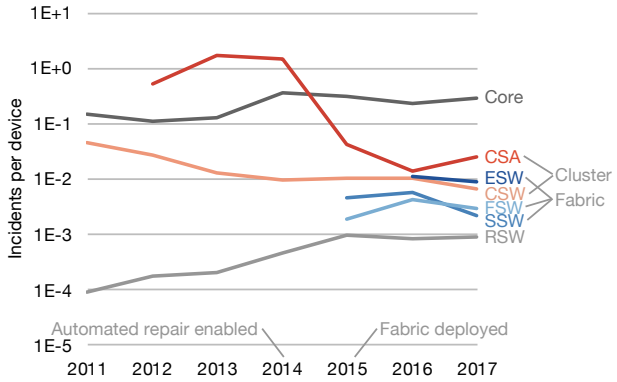

Figure 3: Yearly incident rate of each device type. Note that the y axis is in logarithmic scale and some devices have an incident rate of 0, which occurs if they did not exist in the fleet in a year.

(1) *Network devices with higher bisection bandwidth (e.g., core devices and CSAs in Figure 1) generally have higher incident rates, in comparison to the devices with lower bisection bandwidth (e.g., RSWs).* Intuitively, devices with higher bisection bandwidth tend to affect a larger number of connected devices and are thus correlated with more widespread impact when they fail. The annual incidence rate for ESWs, SSWs, FSWs, RSWs, and CSWs in 2017 is less than 1%.

(2) *Fabric network devices (ESWs, SSWs, and FSWs) have lower incident rates compared to cluster network devices (CSAs and CSWs).* There are two differences between fabric network devices and cluster network devices: (a) fabric network devices are built from commodity chips [5, 6], while cluster network devices are purchased from third-party vendors and (b) fabric networks employ automated repair software to handle common sources of failures [70].

(3) The fact that fabric network devices are less frequently associated with failures verifies that a *fabric network design, equipped with automated failover and repair, is more resilient to device failures.* Specifically, we can see a large rate of CSA-related incidents during 2013 and 2014, where the number of incidents exceeds the number of CSAs (with the incident rate as high as 1.7 and 1.5, respectively). Such high incidence rates were part of the motivation to transition from the cluster network to fabric network.

(4) The CSA-related incident rate decreased in 2015, while the core device-related incident rate has generally increased from pre-2015 levels. This trend can be attributed to two causes: (1) the decreasing size of the CSA population, and (2) new repair practices that were adopted around the time. For example, prior to 2014, network device repairs were often performed *without* draining the traffic on their links. This meant that in the worst case, when things went wrong, maintenance could affect a large volume of traffic. Draining devices prior to maintenance provided a simple but effective way to limit the likelihood of a repair affecting production traffic.

(5) RSW incident rate is increasing over time. We analyze this trend when we discuss Figure 4.

These reliability characteristics influence Facebook's fault tolerant data center network design. For example, Facebook provisions eight core devices in each data center, which allows Facebook data centers to tolerate one unavailable core device (e.g., if it must be removed from operation for maintenance) without any impact on the data center network. Note that nearly all of the core devices and CSAs are third-party vendor devices. In principle, if we do *not* have direct control of the proprietary software on these devices, the network design and implementation must take this lack of control into consideration.[5] For example, it may be more challenging to diagnose, debug, and repair devices that rely on firmware whose source code is unavailable. In these cases, it may make sense to increase the redundancy of devices in case some must be removed for repair by a vendor.

Incident distribution. Figure 4 shows the distribution of incidents caused by each type of network device on a yearly basis. From Figure 4, we make two observations:

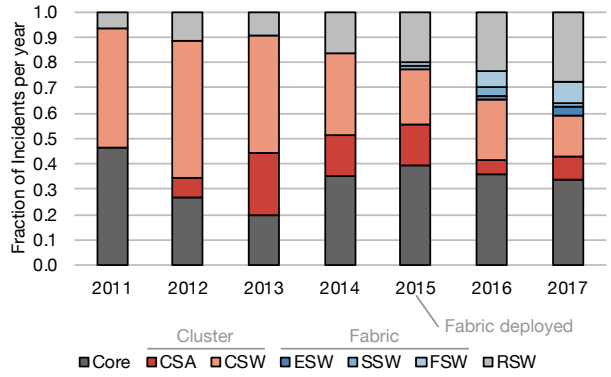

Figure 4: Fraction of network incidents per year broken down by device type.

(1) RSW-related incidents have been steadily increasing over time (a finding that corroborates that of Potharaju et al. [67, 68]). This is partially driven by an increase in the size of the rack population over time. In addition, this is also a result of Facebook's data center network design, where Facebook uses one RSW as the Top-Of-Rack (TOR) switch. Other companies, such as some cloud service providers and enterprises, use two TORs with each server connected to both TORs for redundancy. At Facebook, we find that replicating and distributing server resources leads to low RSW incident rates and is more efficient than using redundant RSWs in every rack.
(2) Devices in fabric networks have not demonstrated a large increase in incidents over time. This again suggests that fabric-based data center designs with automated failover provide good fault tolerance. We analyze this trend further in Section 5.4.

We conclude that (1) higher bandwidth devices have a higher likelihood of causing network incidents, (2) network devices built from commodity chips have much lower incident rates compared to devices from third-party vendors due to automated failover and software repairs, (3) better repair practices lead to lower incident rates, and (4) RSW incidents are increasing over time, but they are still relatively low.

[5] Facebook has been manufacturing custom RSWs and modular switches since 2013. Please refer to the details in [5–7, 75].

5.3 Incident Severity

Key Takeaways
- *Core devices have the most incidents, but they are low severity*
- *Fabric networks have less severe incidents than cluster networks*

Not all incidents are created equal. Facebook classifies incidents into three severity levels from SEV3 (lowest severity) to SEV1 (highest severity). A SEV level reflects the high watermark for an incident. A SEV's level is never downgraded to reflect progress in resolving the SEV. Table 3 provides examples of incidents for each SEV level.

Level	Incident Examples
SEV3	Redundant or contained system failures, system impairments that do not affect or only minimally affect customer experience, internal tool failures.
SEV2	Service outages that affect a particular Facebook feature, regional network impairment, critical internal tool outages that put the site at risk.
SEV1	Entire Facebook product or service outage, data center outage, major portions of the site are unavailable, outages that affect multiple products or services.

Table 3: SEV levels and incident examples.

Figure 5 shows how each type of network SEV in 2017 was distributed among network devices. We make two observations from Figure 5 that complement our raw incident rate findings from §5.2:

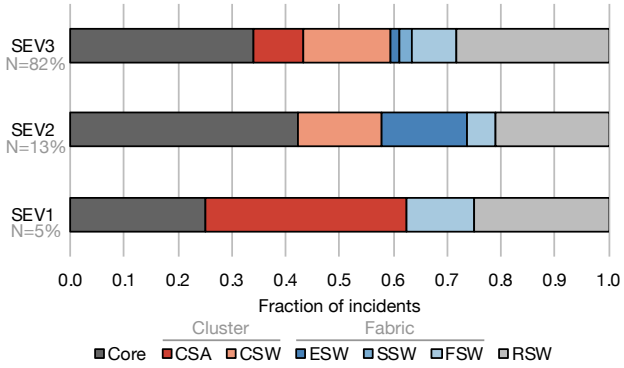

Figure 5: Breakdown of each SEV type across different network devices in 2017.

(1) While core devices have the highest number of SEVs, the severity of core device SEVs is typically low, with around 81% of SEVs at level 3, 15% at level 2, and 4% at level 1. RSWs have nearly as many incidents as core devices, with severity distributed in roughly the same proportion (85%, 10%, and 5% for SEV levels 3, 2, and 1, respectively).
(2) Compared to cluster network devices (CSAs and CSWs), fabric network devices typically have lower severity, with 66% fewer SEV1s, 33% *more* SEV2s (though the overall rate is still relatively

low), and 52% fewer SEV3s. The lower severity is due to the automatic failover and repair support in fabric network devices.

Figure 6 shows how the rate of each SEV level has changed over the years, normalized to the total number of devices in the population during that year. While we cannot disclose the absolute size of the population, we note that it is multiple orders of magnitude larger than similar studies, such as Turner et al. [80]. The main conclusion we draw from Figure 6 is that the overall rate of SEVs per device had an inflection point in 2015, corresponding to the deployment of fabric networks. This was a significant turnaround, as, prior to 2015, the rate of SEV3s grew at a nearly exponential rate.

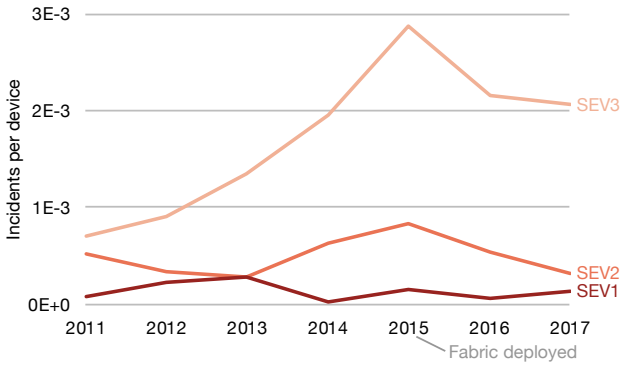

Figure 6: The number of network SEVs over time normalized to the number of deployed network devices. Note that the y axis is in logarithmic scale.

5.4 Network Design

Key Takeaways

- *Cluster network incidents increased steadily over time*
- *Cluster networks have* 2.8× *the incidents versus fabric networks*

We start by describing the composition of Facebook's fleet of network devices. We plot the population breakdown of devices deployed in Facebook's data centers from 2011 to 2017 in Figure 7. Aside from showing the proliferation of RSWs in the fleet, Figure 7 shows that an inflection point occurred in 2015, when the populations of CSWs and CSAs begin to decrease and the populations of FSWs, SSWs, and ESWs begin to increase. This is due to the adoption of fabric networks across more Facebook data centers. In 2017, fabric network device deployment surpassed cluster network device deployment, with 1.5 fabric network devices for every 1 cluster network device in Facebook data centers.

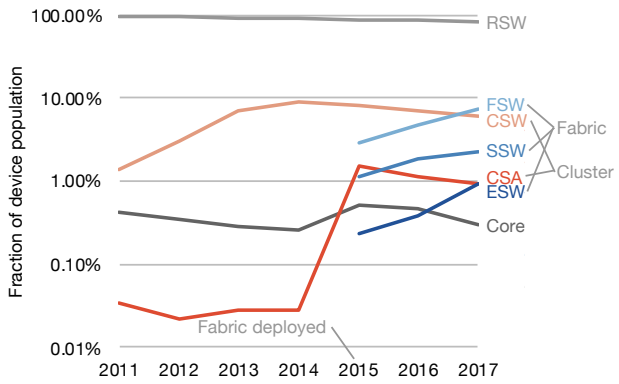

Figure 7: Population breakdown by network device type over the seven-year span of our study. Note that the y axis is in logarithmic scale.

Data center network design plays an important role in network reliability. Figure 8 shows how the fraction of network incidents from the *older* cluster network design and the *newer* fabric network design has changed over time. Cluster network devices are CSAs and CSWs; fabric network devices are ESWs, SSWs, and FSWs. The fraction is calculated by summing the network incidents across *all* of the device types in each network design and dividing it by a common baseline, the number of incidents in 2017. Focusing on 2015, for example, the year fabric networks started being deployed, cluster networks caused nearly the same number of incidents as *all* network incidents in 2017. From Figure 8, we make two observations:

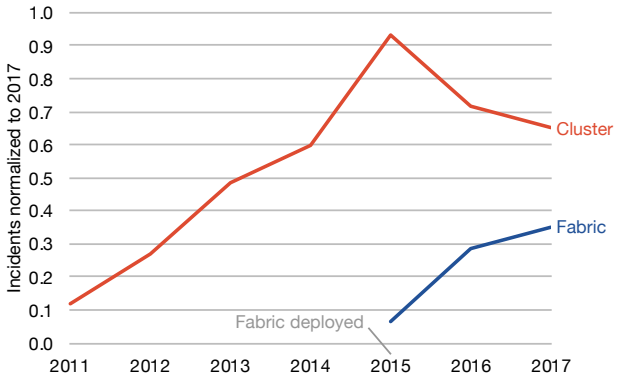

Figure 8: Number of incidents for each network design normalized to a fixed baseline, the total number of SEVs in 2017.

(1) *Cluster network incidents increased steadily* over time until around 2015, when it became challenging to make additional reliability improvements to the cluster network design.
(2) In 2017, the number of incidents for cluster network devices was 1.87× that of fabric network devices, despite fabric networks having 50% more devices. Thus, normalized by number of devices, *cluster network devices have* $1.87 \times 1.5 = 2.8\times$ *as many incidents as fabric network devices.* This is because the

software-managed fault tolerance and automated repair provided by fabric networks can *mask* some failures that would cause incidents in cluster networks.

We conclude cluster networks have around 2× the number of network incidents as fabric networks. We find that fabric networks are more reliable due to their simpler, commodity-chip based switches and automated repair software that dynamically adapts to tolerate device failures.

5.5 Device Reliability

Key Takeaways
- *Incident rates vary by 3 orders of magnitude across device types*
- *Larger networks have longer incident resolution times*

We analyze the reliability of Facebook data center network devices. We use the incident start time and incident resolution time from SEVs to measure *mean time between incidents (MTBI)* and *75th percentile (p75) incident resolution time (p75IRT)*. p75IRT deserves additional explanation. Engineers at Facebook document *resolution time*, not *repair time*, in a SEV. Resolution time exceeds repair time and includes time engineers spend on developing and releasing fixes. To prevent occasional months-long incident resolution times from dominating the mean, we examine the 75th percentile incident resolution time.

MTBI. We measure the average time between the start of two consecutive incidents for MTBI. Figure 9 plots MTBI for each switch type by year. We draw two conclusions from the data:

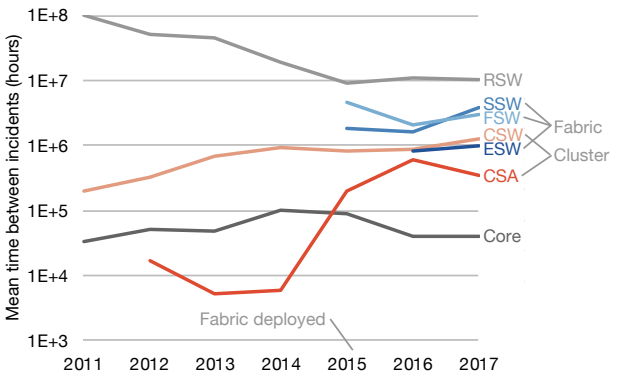

Figure 9: Mean time between incidents in hours for different network device types. Note that the y axis is in logarithmic scale.

First, we find that, from 2011 to 2017, MTBI did not change by more than 10× across each switch type, except CSAs. In 2015, in response to frequent CSA maintenance incidents, engineers strengthened CSA operational procedure guidelines, adding checks to ensure that operators drained CSAs before performing maintenance, for example. These operational improvements increased CSA MTBI by two orders of magnitude between 2014 and 2016.

Second, we find that, in 2017, *MTBI varied by three orders of magnitude across switch types:* from 39,495 device-hours for core devices to 9,958,828 device-hours for RSWs. If we compare MTBI to switch type population size in 2017 (shown in Figure 7), we find that devices with larger population sizes tend to have larger MTBIs. This is because engineers at Facebook have focused on deploying techniques like automated repair mechanisms to devices with large population sizes.

p75IRT. We measure the average time between the start and the resolution of incidents for p75IRT. Figure 10 plots p75IRT for each device type by year.

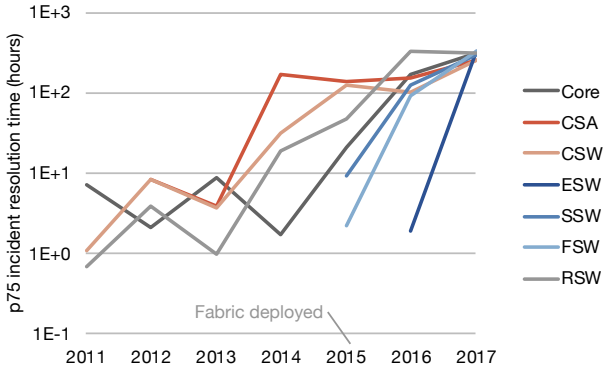

Figure 10: 75th percentile incident resolution time in hours for different network device types. Note that the y axis is in logarithmic scale.

We find that, from 2011 to 2017, p75IRT *increased similarly across device types.* The increase happened without significant changes to individual device design, operation, and management. To explain the overall increase in p75IRT, we plot p75IRT versus the normalized number of devices at Facebook in Figure 11.

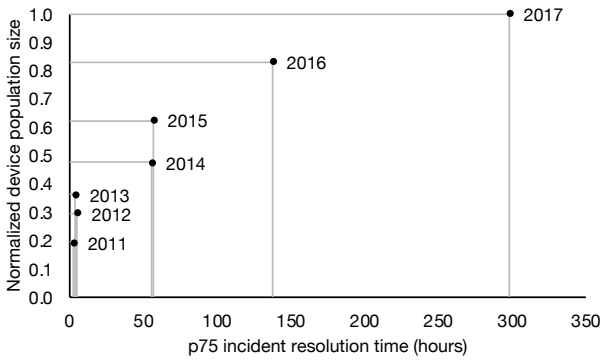

Figure 11: Average p75IRT per year compared to the population size of network devices in Facebook's data centers during that year.

We observe a positive correlation between p75IRT and number of devices. At Facebook, we find that *larger networks increase the time humans take to resolve network incidents.* We attribute part of the increased resolution time to more standardized processes for releasing fixes to production infrastructure. Today, device configuration and software changes go through more thorough review processes, testing, and deployment than in the past.

We conclude that in terms of device reliability, incident rates vary by 3 orders of magnitude across device types in Facebook's data centers and incidents that happen in larger networks tend to have longer incident resolution times.

6 INTER DATA CENTER RELIABILITY

In this section, we study the reliability of backbone networks. We analyze network failures *between* Facebook's data centers over the course of eighteen months, from October 2016 to April 2018, comprising tens of thousands of real world events, comparable in size to Turner et al. [80] and over three times as long of a timescale as Wu et al. [81]. We analyze two types of backbone network failures:

- ***Link failures,*** where an individual bundle of optical fiber linking two edge nodes (Figure 1, ⑤) fails.
- ***Edge node failures,*** where multiple link failures cause an edge node to fail. An edge node connects to the backbone and Internet using at least three links. When all of an edge node's links fail, the edge node fails.

Our backbone network dataset does not contain root causes. We measure *mean time between failures (MTBF)* and *mean time to recovery (MTTR)* for edge nodes and links. We analyze edge node reliability (§6.1), link reliability by fiber vendor (§6.2), and edge node reliability by geography (§6.3).

6.1 Edge Node Reliability

Key Takeaways
- *Typical edge node failure rate is on the order of months*
- *Typical edge node recovery time is on the order of hours*
- *There is high variance in both edge node MTBF and MTTR*

We first analyze the MTBF and MTTR of the edge nodes in Facebook's backbone network. An edge node fails when a combination of planned fiber maintenance or unplanned fiber cuts sever its backbone and Internet connectivity. An edge node recovers when repairs restore its backbone and Internet connectivity.

MTBF. The solid line in Figure 12 plots edge node MTBF in hours as a function of the percentage of edge nodes with that MTBF or lower. Most edge nodes fail infrequently because fiber vendors strive to maintain reliable links. 50% of edge nodes fail less than once every 1710 h, or 2.3 months. And 90% of edge nodes fail less than once every 3521 h, or 4.8 months.

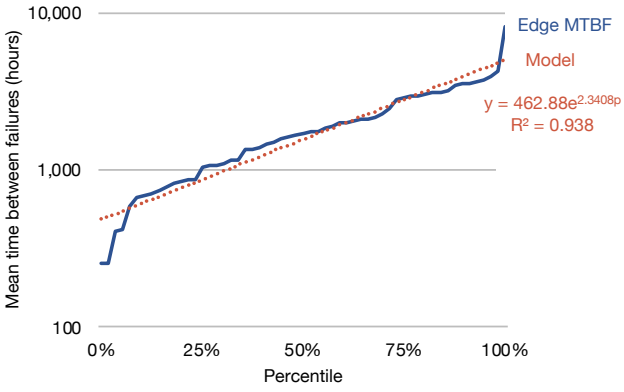

Figure 12: MTBF as a function of percentage of edge nodes connecting Facebook data centers with that MTBF or lower.

Edge nodes exhibit high variance in MTBF due to their diverse fiber vendor makeup and geographic locations (observations we explore in §6.2 and §6.3). The standard deviation of edge node MTBF is 1320 h, with the least reliable edge node failing, on average, once every 253 h and the most reliable edge node failing, on average, once every 8025 h.

We model $MTBF_{edge}(p)$ as an exponential function of the percentage of edge nodes, $0 \leq p \leq 1$, with that MTBF or lower. We built the models in this section by fitting an exponential function using the least squares method. At Facebook, we use these models in capacity planning to calculate *conditional risk*, the probability of an edge node or link being unavailable or overloaded. We plan edge node and link capacity to ensure conditional risk is below 0.0001. We find that $MTBF_{edge}(p) = 462.88e^{2.3408p}$ (the dotted line in Figure 12) with $R^2 \approx 0.94$.

MTTR. The solid line in Figure 13 plots edge node MTTR in hours as a function of the percentage of edge nodes with that MTTR or lower. Edge node recovery occurs much faster than the time between failures because edge nodes contain multiple links (at least three) and fiber vendors work to repair link failures rapidly. 50% of edge nodes recover within 10 h of a failure; 90% within 71 h.

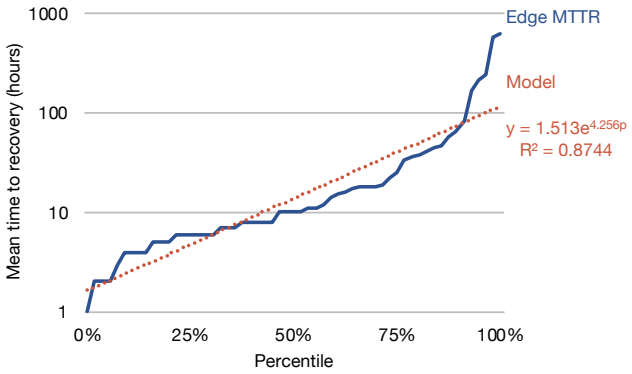

Figure 13: MTTR as a function of percentage of edge nodes connecting Facebook data centers with that MTTR or lower.

Edge nodes exhibit high variance in MTTR because some edge nodes are easier to repair than others. Imagine the differences between an edge node on a remote island compared to an edge node in a big city. Weather conditions, physical terrain, and travel time affect the time it takes a fiber vendor to repair an edge node's links. The standard deviation of edge node MTTR is 112 h, with the slowest edge node to recover taking 608 h and the fastest edge node to recover taking 1 h.

We model $MTTR_{edge}(p)$ as an exponential function of the percentage of edge nodes, $0 \leq p \leq 1$ with that MTTR or lower. We find that $MTTR_{edge}(p) = 1.513e^{4.256p}$ (the dotted line in Figure 13) with $R^2 \approx 0.87$.

The high variances in edge node MTBF and MTTR motivate us to study the reliability characteristics of the links connecting edge nodes in §6.2 and the geographic location of edge nodes in §6.3.

6.2 Link Reliability by Fiber Vendor

Key Takeaways
- *Typical vendor link failure rate is on the order of months*
- *We observe higher MTBF for edge nodes in big cities*
- *Vendor MTBF and MTTR each span multiple orders of magnitude*

We analyze the MTBF and MTTR for fiber vendors based on when the links they operate fail or recover. For brevity, we shorten *"the MTBF/MTTR of the links operated by a fiber vendor"* to *"fiber vendor MTBF/MTTR."*

MTBF. The solid line in Figure 14 plots the fiber vendor MTBF in hours as a function of the percentage of fiber vendors with that MTBF or lower. For most vendors, link failure happens only occasionally due to regular maintenance and monitoring. 50% of vendors have links that fail less than once every 2326 h, or 3.2 months. And 90% of vendors have links that fail less than once every 5709 h, or 7.8 months.

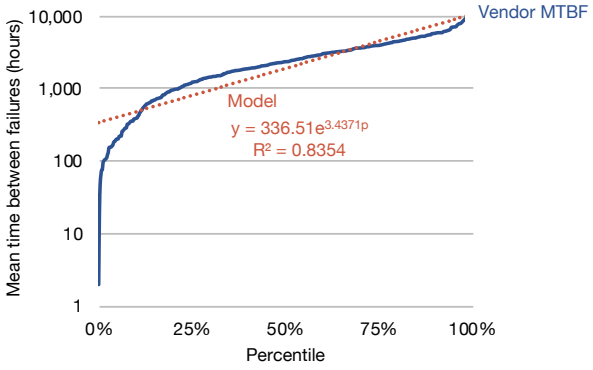

Figure 14: MTBF as a function of percentage of fiber vendors with that MTBF or lower.

Fiber vendor MTBF varies by orders of magnitude. The standard deviation of fiber vendor MTBF is 2207 h, with the least reliable vendor's links failing, on average, once every 2 h and the most reliable vendor's links failing, on average, once every 11,721 h. Anecdotally, we observe that fiber markets with high competition lead to more incentive for fiber vendors to increase reliability. For example, the most reliable vendor operates in a big city in the USA.

We model $MTBF_{vendor}(p)$ as an exponential function of the percentage of vendors, $0 \leq p \leq 1$ with that MTBF or lower. We find that $MTBF_{vendor}(p) = 336.51e^{3.4371p}$ (the dotted line in Figure 14) with $R^2 \approx 0.84$.

MTTR. The solid line in Figure 15 plots fiber vendor MTTR as a function of the percentage of fiber vendors with that MTTR or lower. Most vendors repair links promptly. 50% of vendors repair links within 13 h of a failure; 90% within 60 h.

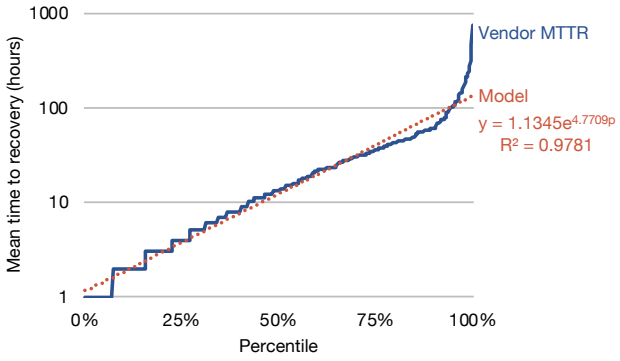

Figure 15: MTTR as a function of percentage of fiber vendors with that MTTR or lower.

Fiber vendors exhibit high variance in MTTR because some fiber vendors operate in areas where they can more easily repair links (an observation we analyze in §6.3). The standard deviation of fiber vendor MTTR is 56 h, with the slowest vendor taking on average 744 h to repair their links and the fastest vendor taking on average 1 h to repair their links.

We model $MTTR_{vendor}(p)$ as an exponential function of the percentage of vendors, $0 \leq p \leq 1$ with that MTTR or lower. We find that $MTTR_{vendor}(p) = 1.1345e^{4.7709p}$ (the dotted line in Figure 15) with $R^2 \approx 0.98$.

We conclude that *not all fiber vendors operate equally*. We summarize the reliability models we developed in Table 4. We next explore how the geographic location of edge nodes affects their reliability.

Reliability Model	Exponential Function	R^2
Edge node MTBF	$462.88e^{2.3408p}$	0.94
Edge node MTTR	$1.513e^{4.256p}$	0.87
Vendor MTBF	$336.51e^{3.4371p}$	0.84
Vendor MTTR	$1.1345e^{4.7709p}$	0.98

Table 4: Each reliability model is an exponential function expressing the MTBF or MTTR for a given percentile, $0 \leq p \leq 1$, of edge nodes (or vendors).

6.3 Edge Node Reliability by Geography

Key Takeaways
- *Edge node failure rate is similar across most continents*
- *Edge nodes recover within* 1 day *on average on all continents*

We analyze the reliability of edge nodes by their geographic location (the continent they reside in). Table 5 shows the distribution of edge nodes in Facebook's network across continents. Most edge nodes reside in North America, followed closely by Europe. The continents with the fewest edge nodes are Africa and Australia.

Continent	Distribution	MTBF (h)	MTTR (h)
North America	37%	1848	17
Europe	33%	2029	19
Asia	14%	2352	11
South America	10%	1579	9
Africa	4%	5400	22
Australia	2%	1642	2

Table 5: The distribution and reliability of edge nodes in Facebook's network across continents.

MTBF. We show the average MTBF for the edge nodes in each continent in Table 5. Edge nodes in Africa are outliers, with an average MTBF of 5400 h, or 7.4 months. Edge node reliability in Africa is important because edge nodes in Africa are few and connect Europe to Asia. Edge nodes in North America, South America, Europe, Asia, and Australia have average MTBFs ranging from 1579 h (2.2 months, for South America) to 2352 h (3.2 months, for Asia). The standard deviation of edge node MTBF across continents is 1333 h, or 1.8 months.

MTTR. We show the average MTTR for the edge nodes in each continent in Table 5. Across continents, edge nodes recover within 1 day on average. Edge nodes in Africa, despite their long uptime, take the longest time, on average, to recover (22 h), in part due to their submarine links. Edge nodes in Australia take the shortest time, on average, to recover (2 h), due to their locations in big cities. We observe a 7 h standard deviation in edge node MTTR across continents.

We conclude that edge node failure rate varies by *months* depending on the continent that edge nodes reside in, and edge nodes typically recover in 1 day across continents.

7 RELATED WORK

To our knowledge, this paper provides the first comprehensive study of network incidents from the perspective of large scale web services. Prior large scale data center failure studies [9, 16, 36, 65] report that network incidents are among the major causes of web service outages. However, none of these studies systematically analyze network incidents at a large scale, focusing on the availability of an *entire web service*, across both *inter* and *intra* data center networks, in a long term, longitudinal study.

Prior studies examine the failure characteristics of network links and devices in *different* types of networks studied in this paper, including data center networks [31, 67, 68, 83] and optical backbones [30, 54, 68]. Specifically, Potharaju and Jain [68] and Turner et al. [80] study data center network infrastructure by characterizing device and link failures in intra and inter data center networks. Their studies characterize the failure impact, including connectivity losses, high latency, packet drops, and so on. These studies significantly boost the understanding of network failure characteristics, and provide insights for network engineers and operators for improving the fault tolerance of existing networks and for designing more robust networks.

While our work is closely related to these prior studies, it is also *fundamentally* different and *complementary* in the following three aspects. First, our work has a different goal. Unlike these prior studies that focus on understanding fine-grained per-device, per-link failures and their impact on the system-level services above the network stack, our work focuses on how network incidents affect *the availability of an Internet service*. Our goal is to reveal and quantify the incidents that *cannot* be tolerated despite industry best practices, and shed light on how large scale systems can operate reliably in the presence of these incidents. Second, prior studies only cover the data center and backbone networks with traditional cluster network designs, whereas our work presents a comparative study of the reliability characteristics of data center network infrastructure with *both* a traditional cluster network design and a contemporary fabric network design with smaller, commodity switches. As introduced in §3, we achieve this due to the heterogeneity of the data center network infrastructure of Facebook where networks with different designs co-exist and co-operate. Third, we present a long-term (seven years for intra data center networks and eighteen months for inter data center networks) longitudinal analysis to reveal the evolution of network reliability characteristics, while prior studies typically provide only aggregated results, often over a much shorter period or with orders of magnitude fewer devices [80].

Govindan et al. [34] study over 100 failure events in Google WAN and data center networks, offering insights into why maintaining high levels of availability is challenging for content providers. Their study, similar to [9, 16, 36, 65], focuses on network management and the design principles for building robust networks. Many of the high-level design principles mentioned in [34], such as using multiple layers of fallback (defense in depth), continuous prevention, and fast recovery, are applicable to large scale software systems to protect against network incidents.

Other large scale studies examined failures of DRAM devices [41, 59, 74, 77, 78] and SSDs [58, 61, 73] in clusters and data centers. Our paper complements these studies.

8 CONCLUSIONS

At large scale Internet companies like Facebook, it is important to maintain a reliable network infrastructure both within and between data centers. In this study, we presented a large scale, longitudinal study of data center network reliability based on operational data collected from the production network infrastructure at Facebook. Our study spans thousands of intra data center network incidents across seven years, and eighteen months of inter data center network incidents. We show how the reliability characteristics of different network designs and different network device types manifest as network incidents and affect the software systems that use the network.

As software systems grow in complexity, interconnectedness, and geographic distribution, unwanted behavior from network infrastructure has the potential to become a key limiting factor in the ability to reliably operate distributed software systems at a large scale. It is our hope that the research community can build upon our comprehensive study to better characterize, understand, and improve the reliability of large scale data center networks and systems.

ACKNOWLEDGMENTS

We would like to thank Alexander Gilgur, Alexander Nikolaidis, Boliu Xu, Codey Oxley, Hans Ragas, James Zeng, Jimmy Williams, and Omar Baldonado for their feedback and suggestions on this paper.

REFERENCES

[1] Facebook Newsroom: Company Info. https://newsroom.fb.com/company-info/, Sept. 2018.

[2] Al-Fares, M., Loukissas, A., and Vahdat, A. A Scalable, Commodity Data Center Network Architecture. In *Proceedings of the 2008 ACM SIGCOMM Conference* (Seattle, WA, USA, 2008).

[3] Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and Sridharan, M. Data Center TCP (DCTCP). In *Proceedings of the 2010 ACM SIGCOMM Conference* (New Delhi, India, 2010).

[4] Andreyev, A. Introducing data center fabric, the next-generation Facebook data center network. https://code.facebook.com/posts/360346274145943/introducing-data-center-fabric-the-next-generation-facebook-data-center-network/, Nov. 2014.

[5] Bachar, Y. Introducing "6-pack": the first open hardware modular switch. https://code.facebook.com/posts/843620439027582/facebook-open-switching-system-fboss-and-wedge-in-the-open/, Feb. 2015.

[6] Bachar, Y., and Simpkins, A. Introducing "Wedge" and "FBOSS," the next steps toward a disaggregated network. https://code.facebook.com/posts/681382905244727/introducing-wedge-and-fboss-the-next-steps-toward-a-disaggregated-network/, June 2014.

[7] Bagga, J., and Yao, Z. Open networking advances with Wedge and FBOSS. https://code.facebook.com/posts/145488969140934/open-networking-advances-with-wedge-and-fboss/, Nov. 2015.

[8] Bailis, P., and Kingsbury, K. The Network is Reliable: An informal survey of real-world communications failures. *Communications of the ACM (CACM) 57*, 9 (Sept. 2014), 48–55.

[9] Barroso, L. A., Clidaras, J., and Hölzle, U. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-scale Machines*, 2 ed. Morgan and Claypool Publishers, 2013.

[10] Beaver, D., Kumar, S., Li, H. C., Sobel, J., and Vajgel, P. Finding a Needle in Haystack: Facebook's Photo Storage. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation* (Vancouver, BC, Canada, 2010).

[11] Beckett, R., Gupta, A., Mahajan, R., and Walker, D. A General Approach to Network Configuration Verification. In *Proceedings of the 2017 ACM SIGCOMM Conference* (Los Angeles, CA, USA, 2017).

[12] Beckett, R., Mahajan, R., Millstein, T., Padhye, J., and Walker, D. Don't Mind the Gap: Bridging Network-wide Objectives and Device-level Configurations. In *Proceedings of the 2016 ACM SIGCOMM Conference* (Florianópolis, Brazil, 2016).

[13] Beckett, R., Mahajan, R., Millstein, T., Padhye, J., and Walker, D. Network Configuration Synthesis with Abstract Topologies. In *Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Barcelona, Spain, 2017).

[14] Benson, T., Akella, A., and Maltz, D. Unraveling the Complexity of Network Management. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation* (Boston, MA, USA, 2009).

[15] Benson, T., Akella, A., and Shaikh, A. Demystifying Configuration Challenges and Trade-offs in Network-based ISP Services. In *Proceedings of the 2011 ACM SIGCOMM Conference* (Toronto, ON, Canada, 2011).

[16] Brewer, E. Spanner, TrueTime and the CAP Theorem. Tech. rep., Google Inc., Feb. 2017. https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/45855.pdf.

[17] Bronson, N., Amsden, Z., Cabrera, G., Chakka, P., Dimov, P., Ding, H., Ferris, J., Giardullo, A., Kulkarni, S., Li, H. C., Marchukov, M., Petrov, D., Puzar, L., Song, Y. J., and Venkataramani, V. TAO: Facebook's Distributed Data Store for the Social Graph. In *Proceedings of the 2013 USENIX Annual Technical Conference* (San Jose, CA, June 2013).

[18] Chen, G. J., Wiener, J. L., Iyer, S., Jaiswal, A., Lei, R., Simha, N., Wang, W., Wilfong, K., Williamson, T., and Yilmaz, S. Realtime Data Processing at Facebook. In *Proceedings of the 2016 ACM SIGMOD/PODS Conference* (San Francisco, CA, USA, 2016).

[19] Clos, C. A study of non-blocking switching networks. *The Bell System Technical Journal 32*, 2 (Mar. 1953), 406–424.

[20] Darrow, B. Superstorm Sandy wreaks havoc on internet infrastructure. https://gigaom.com/2012/10/30/superstorm-sandy-wreaks-havoc-on-internet-infrastructure/, 2012.

[21] Dean, J., and Ghemawat, S. MapReduce: Simplified Data Processing on Large Clusters. In *Proceedings of the 6th Symposium on Operating Systems Design and Implementation* (San Francisco, CA, USA, 2004).

[22] Evans, J. The HipHop Virtual Machine. https://code.facebook.com/posts/495167483903373/the-hiphop-virtual-machine/, Dec. 2011.

[23] Farmer, S. More On Gmail's Delivery Delays. https://cloud.googleblog.com/2013/09/more-on-gmails-delivery-delays.html, 2013. Google Cloud Official Blog.

[24] Farrington, N., and Andreyev, A. Facebook's Data Center Network Architecture. In *Proceedings of the 2013 IEEE Optical Interconnects Conference* (Santa Fe, New Mexico, May 2013).

[25] Fayaz, S. K., Sharma, T., Fogel, A., Mahajan, R., Millstein, T., Sekar, V., and Varghese, G. Efficient Network Reachability Analysis using a Succinct Control Plane Representation. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation* (Savannah, GA, USA, 2016).

[26] Ferreira, J., Hasani, N., Sankar, S., Williams, J., and Schiff, N. Fabric Aggregator: A flexible solution to our traffic demand. https://code.fb.com/data-center-engineering/fabric-aggregator-a-flexible-solution-to-our-traffic-demand/, 2018.

[27] Fogel, A., Fung, S., Pedrosa, L., Walraed-Sullivan, M., Govindan, R., Mahajan, R., and Millstein, T. A General Approach to Network Configuration Analysis. In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation* (Oakland, CA, USA, 2015).

[28] Ford, D., Labelle, F., Popovici, F. I., Stokely, M., Truong, V.-A., Barroso, L., Grimes, C., and Quinlan, S. Availability in Globally Distributed Storage Systems. In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation* (Vancouver, BC, Canada, Oct. 2010).

[29] Gember-Jacobson, A., Akella, A., Mahajan, R., and Liu, H. H. Automatically Repairing Network Control Planes Using an Abstract Representation. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China, 2017).

[30] Ghobadi, M., and Mahajan, R. Optical Layer Failures in a Large Backbone. In *Proceedings of the 2016 Internet Measurement Conference* (Santa Monica, CA, USA, 2016), IMC '16.

[31] Gill, P., Jain, N., and Nagappan, N. Understanding Network Failures in Data Centers: Measurement, Analysis, and Implications. In *Proceedings of the 2011 ACM SIGCOMM Conference* (Toronto, ON, Canada, 2011).

[32] Google Cloud Platform. Google Compute Engine Incident #16007. https://status.cloud.google.com/incident/compute/16007, 2016.

[33] Google Cloud Platform. Google Cloud Networking Incident #17002. https://status.cloud.google.com/incident/cloud-networking/17002, 2017.

[34] Govindan, R., Minei, I., Kallahalla, M., Koley, B., and Vahdat, A. Evolve or Die: High-Availability Design Principles Drawn from Google's Network Infrastructure. In *Proceedings of the 2016 ACM SIGCOMM Conference* (Florianópolis, Brazil, Aug. 2016).

[35] Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. A., Patel, P., and Sengupta, S. VL2: A Scalable and Flexible Data Center Network. In *Proceedings of the 2009 ACM SIGCOMM Conference* (Barcelona, Spain, 2009).

[36] Gunawi, H. S., Hao, M., Suminto, R. O., Laksono, A., Satria, A. D., Adityatama, J., and Eliazar, K. J. Why Does the Cloud Stop Computing? Lessons from Hundreds of Service Outages. In *Proceedings of the 7th ACM Symposium on Cloud Computing* (Santa Clara, CA, Oct. 2016).

[37] Guo, C., Lu, G., Li, D., Wu, H., Zhang, X., Shi, Y., Tian, C., Zhang, Y., and Lu, S. BCube: A High Performance, Server-centric Network Architecture for Modular Data Centers. In *Proceedings of the 2009 ACM SIGCOMM Conference* (Barcelona, Spain, 2009).

[38] Guo, C., Wu, H., Tan, K., Shi, L., Zhang, Y., and Lu, S. DCell: A Scalable and Fault-tolerant Network Structure for Data Centers. In *Proceedings of the 2008 ACM SIGCOMM Conference* (Seattle, WA, USA, 2008).

[39] Hong, C.-Y., Kandula, S., Mahajan, R., Zhang, M., Gill, V., Nanduri, M., and Wattenhofer, R. Achieving High Utilization with Software-driven WAN. In *Proceedings of the 2013 ACM SIGCOMM Conference* (Hong Kong, China, 2013).

[40] Huang, Q., Ang, P., Knowles, P., Nykiel, T., Tverdokhlib, I., Yajurvedi, A., Dapolito, IV, P., Yan, X., Bykov, M., Liang, C., Talwar, M., Mathur, A., Kulkarni, S., Burke, M., and Lloyd, W. SVE: Distributed Video Processing at Facebook Scale. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China, 2017).

[41] Hwang, A., Stefanovici, I., and Schroeder, B. Cosmic Rays Don't Strike Twice: Understanding the Characteristics of DRAM Errors and the Implications for System Design. In *ASPLOS* (2012).

[42] International Phonetic Association. *Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet.* Cambridge University Press, 1999.

[43] Jain, S., Kumar, A., Mandal, S., Ong, J., Poutievski, L., Singh, A., Venkata, S., Wanderer, J., Zhou, J., Zhu, M., Zolla, J., Hölzle, U., Stuart, S., and Vahdat, A. B4: Experience with a Globally-deployed Software Defined WAN. In *Proceedings of the 2013 ACM SIGCOMM Conference* (Hong Kong, China, 2013).

[44] Jimenez, M., and Kwok, H. Building Express Backbone: Facebook's new long-haul network. https://code.facebook.com/posts/1782709872057497/building-express-backbone-facebook-s-new-long-haul-network/, May 2017.

[45] Joseph, D. A., Tavakoli, A., and Stoica, I. A Policy-aware Switching Layer for Data Centers. In *Proceedings of the 2008 ACM SIGCOMM Conference* (Seattle, WA, USA, 2008).

[46] Kaldor, J., Mace, J., Bejda, M., Gao, E., Kuropatwa, W., O'Neill, J., Ong, K. W., Schaller, B., Shan, P., Viscomi, B., Venkataraman, V., Veeraraghavan, K., and Song, Y. J. Canopy: An End-to-End Performance Tracing And Analysis System. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China, Oct. 2017).

[47] Kandula, S., Menache, I., Schwartz, R., and Babbula, S. R. Calendaring for Wide Area Networks. In *Proceedings of the 2014 ACM SIGCOMM Conference* (Chicago, IL, USA, 2014).

[48] Khurshid, A., Zou, X., Zhou, W., Caesar, M., and Godfrey, P. B. VeriFlow: Verifying Network-Wide Invariants in Real Time. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation* (Lombard,

IL, USA, 2013).

[49] Liu, H. H., Zhu, Y., Padhye, J., Cao, J., Tallapragada, S., Lopes, N. P., Rybalchenko, A., Lu, G., and Yuan, L. CrystalNet: Faithfully Emulating Large Production Networks. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China, 2017).

[50] Liu, V., Halperin, D., Krishnamurthy, A., and Anderson, T. F10: A Fault-Tolerant Engineered Network. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation* (Lombard, IL, USA, 2013).

[51] Luo, Y., Govindan, S., Sharma, B., Santaniello, M., Meza, J., Kansal, A., Liu, J., Khessib, B., Vaid, K., and Mutlu, O. Characterizing application memory error vulnerability to optimize datacenter cost via heterogeneous-reliability memory. In *Proceedings of the 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks* (June 2014), pp. 467–478.

[52] Mahajan, R., Wetherall, D., and Anderson, T. Understanding BGP Misconfiguration. In *Proceedings of the ACM 2002 SIGCOMM Conference* (Pittsburgh, PA, USA, 2002).

[53] Malewicz, G., Austern, M. H., Bik, A. J., Dehnert, J. C., Horn, I., Leiser, N., and Czajkowski, G. Pregel: A System for Large-scale Graph Processing. In *Proceedings of the 2010 ACM SIGMOD/PODS Conference* (Indianapolis, IN, USA, 2010).

[54] Markopoulou, A., Iannaccone, G., Bhattacharyya, S., Chuah, C.-N., Ganjali, Y., and Diot, C. Characterization of Failures in an Operational IP Backbone Network. *IEEE/ACM Transactions on Networking 16*, 4 (Aug. 2008), 749–762.

[55] Maurer, B. Fail at Scale: Reliability in the Face of Rapid Change. *Communications of the ACM (CACM) 58*, 11 (Nov. 2015), 44–49.

[56] Medem, A., Akodjenou, M.-I., and Teixeira, R. TroubleMiner: Mining Network Trouble Tickets. In *Proceedings of the 11th IFIP/IEEE International Symposium on Integrated Network Management* (New York, NY, USA, 2009).

[57] Mellette, W. M., McGuinness, R., Roy, A., Forencich, A., Papen, G., Snoeren, A. C., and Porter, G. RotorNet: A Scalable, Low-complexity, Optical Datacenter Network. In *Proceedings of the 2017 ACM SIGCOMM Conference* (Los Angeles, CA, USA, 2017).

[58] Meza, J., Wu, Q., Kumar, S., and Mutlu, O. A large-scale study of flash memory errors in the field. In *ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (2015).

[59] Meza, J., Wu, Q., Kumar, S., and Mutlu, O. Revisiting memory errors in large-scale production data centers: analysis and modeling of new trends from the field. In *IEEE/IFIP International Conference on Dependable Systems and Networks* (2015).

[60] Muralidhar, S., Lloyd, W., Roy, S., Hill, C., Lin, E., Liu, W., Pan, S., Shankar, S., Sivakumar, V., Tang, L., and Kumar, S. f4: Facebook's Warm BLOB Storage System. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation* (Broomfield, CO, USA, 2014).

[61] Narayanan, I., Wang, D., Jeon, M., Sharma, B., Caulfield, L., Sivasubramaniam, A., Cutler, B., Liu, J., Khessib, B., and Vaid, K. SSD Failures in Datacenters: What? When? And Why? In *SYSTOR* (2016).

[62] Niranjan Mysore, R., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., Subramanya, V., and Vahdat, A. PortLand: A Scalable Fault-tolerant Layer 2 Data Center Network Fabric. In *Proceedings of the 2009 ACM SIGCOMM Conference* (Barcelona, Spain, 2009).

[63] Nishtala, R., Fugal, H., Grimm, S., Kwiatkowski, M., Lee, H., Li, H., McElroy, R., Paleczny, M., Peek, D., Saab, P., Stafford, D., Tung, T., and Venkataramani, V. Scaling Memcache at Facebook. In *Proceedings of the 10th USENIX Symposium on Networked Systems Design and Implementation* (Lombard, IL, USA, 2013).

[64] Noormohammadpour, M., Raghavendra, C. S., Rao, S., and Kandula, S. DCCast: Efficient Point to Multipoint Transfers Across Datacenters. In *Proceedings of the 9th USENIX Workshop on Hot Topics in Cloud Computing* (Santa Clara, CA, USA, 2017).

[65] Oppenheimer, D., Ganapathi, A., and Patterson, D. A. Why Do Internet Services Fail, and What Can Be Done About It? In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems* (Seattle, WA, USA, Mar. 2003).

[66] Pelkonen, T., Franklin, S., Teller, J., Cavallaro, P., Huang, Q., Meza, J., and Veeraraghavan, K. Gorilla: A Fast, Scalable, In-Memory Time Series Database. In *Proceedings of the 41st International Conference on Very Large Data Bases* (Kohala Coast, HI, USA, Aug. 2015).

[67] Potharaju, R., and Jain, N. Demystifying the Dark Side of the Middle: A Field Study of Middlebox Failures in Datacenters. In *Proceedings of the 2013 Conference on Internet Measurement Conference* (Barcelona, Spain, 2013).

[68] Potharaju, R., and Jain, N. When the Network Crumbles: An Empirical Study of Cloud Network Failures and Their Impact on Services. In *Proceedings of the 4th Annual Symposium on Cloud Computing* (Santa Clara, CA, USA, 2013).

[69] Potharaju, R., Jain, N., and Nita-Rotaru, C. Juggling the Jigsaw: Towards Automated Problem Inference from Network Trouble Tickets. In *Proceedings of the 10th USENIX Symposium on Networked Systems Designa nd Implementation* (Lombard, IL, USA, 2013).

[70] Power, A. Making facebook self-healing. https://www.facebook.com/notes/facebook-engineering/making-facebook-self-healing/10150275248698920/, 2011.

[71] Roy, A., Zeng, H., Bagga, J., Porter, G., and Snoeren, A. C. Inside the Social Network's (Datacenter) Network. In *Proceedings of the 2015 ACM SIGCOMM Conference* (London, United Kingdom, 2015).

[72] Schlinker, B., Kim, H., Cui, T., Katz-Bassett, E., Madhyastha, H. V., Cunha, I., Quinn, J., Hasan, S., Lapukhov, P., and Zeng, H. Engineering Egress with Edge Fabric: Steering Oceans of Content to the World. In *Proceedings of the 2017 ACM SIGCOMM Conference* (Los Angeles, CA, USA, 2017).

[73] Schroeder, B., and Gibson, G. Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You? In *FAST* (2007).

[74] Schroeder, B., Pinheiro, E., and Weber, W.-D. DRAM Errors in the Wild: A Large-Scale Field Study. In *SIGMETRICS/Performance* (2009).

[75] Simpkins, A. Facebook Open Switching System ("FBOSS") and Wedge in the open. https://code.facebook.com/posts/843620439027582/facebook-open-switching-system-fboss-and-wedge-in-the-open/, Mar. 2015.

[76] Singh, A., Ong, J., Agarwal, A., Anderson, G., Armistead, A., Bannon, R., Boving, S., Desai, G., Felderman, B., Germano, P., Kanagala, A., Provost, J., Simmons, J., Tanda, E., Wanderer, J., Hölzle, U., Stuart, S., and Vahdat, A. Jupiter Rising: A Decade of Clos Topologies and Centralized Control in Google's Datacenter Network. In *Proceedings of the 2015 ACM SIGCOMM Conference* (London, United Kingdom, 2015).

[77] Sridharan, V., and Liberty, D. A Study of DRAM Failures in the Field. In *SC* (2012).

[78] Sridharan, V., Stearley, J., DeBardeleben, N., Blanchard, S., and Gurumurthi, S. Feng Shui of Supercomputer Memory: Positional Effects in DRAM and SRAM Faults. In *SC* (2013).

[79] The AWS Team. Summary of the October 22, 2012 AWS Service Event in the US-East Region. https://aws.amazon.com/message/680342/, 2012.

[80] Turner, D., Levchenko, K., Snoeren, A. C., and Savage, S. California Fault Lines: Understanding the Causes and Impact of Network Failures. In *Proceedings of the 2010 ACM SIGCOMM Conference* (New Delhi, India, 2010).

[81] Wu, X., Turner, D., Chen, C.-C., Maltz, D. A., Yang, X., Yuan, L., and Zhang, M. NetPilot: Automating Datacenter Network Failure Mitigation. In *Proceedings of the 2012 ACM SIGCOMM Conference* (Helsinki, Finland, 2012).

[82] Yap, K.-K., Motiwala, M., Rahe, J., Padgett, S., Holliman, M., Baldus, G., Hines, M., Kim, T., Narayanan, A., Jain, A., Lin, V., Rice, C., Rogan, B., Singh, A., Tanaka, B., Verma, M., Sood, P., Tariq, M., Tierney, M., Trumic, D., Valancius, V., Ying, C., Kallahalla, M., Koley, B., and Vahdat, A. Taking the edge off with espresso: Scale, reliability and programmability for global internet peering. In *Proceedings of the 2017 ACM SIGCOMM Conference* (Los Angeles, CA, USA, 2017).

[83] Zhuo, D., Ghobadi, M., Mahajan, R., Förster, K.-T., Krishnamurthy, A., and Anderson, T. Understanding and Mitigating Packet Corruption in Data Center Networks. In *Proceedings of the 2017 ACM SIGCOMM Conference* (Los Angeles, CA, USA, 2017).