Public Review for

Surviving switch failures in cloud datacenters

Rachee Singh, Muqeet Mukhtar, Ashay Krishna, Aniruddha Parkhi, Jitendra Padhye, David Maltz

In this paper, the authors aim at estimating the expected time-to failure of cloud datacenter switches and quantify the factors that impact it negatively. To do so, they studied over 180,000 switches in datacenters across 130 geographical locations for a period of 3 months by collecting a combination of signals from their control, data and management plane.

They also applied the non-parametric Kaplan-Meier and the semi-parametric Cox Proportional Hazard models to identify the impact of switch hardware/software characteristics on its time-to failure likelihood. The main findings are that 32% of switch failures can be attributed to hardware faults, there is a 2% chance of having a switch failure within 3 months of being deployed in production and 17% of switch failures occur due to software bugs in vendor switch OSes.

The reviewers appreciated the analysis performed by the authors which has been considered very solid. Furthermore, the insights provided in this paper can be of great help for our community to better understand problems associated to production networks.

Public review written by
Gianni Antichi
Queen Mary University of London, UK

Surviving switch failures in cloud datacenters

Rachee Singh, Muqeet Mukhtar, Ashay Krishna, Aniruddha Parkhi,
Jitendra Padhye, David Maltz
Microsoft
rachee.singh@microsoft.com

ABSTRACT

Switch failures can hamper access to client services, cause link congestion and blackhole network traffic. In this study, we examine the nature of switch failures in the datacenters of a large commercial cloud provider through the lens of survival theory. We study a cohort of over 180,000 switches with a variety of hardware and software configurations and find that datacenter switches have a 98% likelihood of functioning uninterrupted for over 3 months since deployment in production. However, there is significant heterogeneity in switch survival rates with respect to their hardware and software: the switches of one vendor are twice as likely to fail compared to the others. We attribute the majority of switch failures to hardware impairments and unplanned power losses. We find that the in-house switch operating system, SONiC, boosts the survival likelihood of switches in datacenters by 1% by eliminating switch failures caused by software bugs in vendor switch OSes.

CCS CONCEPTS

• **Networks** → **Data center networks**; **Network reliability**; **Routers**;

KEYWORDS

Data center networks, Router failures, Survival theory

1 INTRODUCTION

Cloud providers support a large variety of applications with strong performance and reliability guarantees. A key enabler of reliable access to these applications is the design and architecture of modern datacenters. Network switches that form tiers of the clos-like datacenter topology are central to the functionality and reliability of datacenters [12]. Their failures can interrupt client services, increase the load on other switches and congest datacenter links.

In this work, our goal is to estimate the expected time-to-failure, or *survival time*, of datacenter switches and quantify the factors that impact it negatively. We examine the root causes of switch failures and propose ways to mitigate them. Our study spans over 180,000 switches in datacenters across 130 geographical locations for a period of 3 months. We collect a combination of signals from the control, data and management plane of the switches to infer the time and duration of failure events and augment these failure logs with hardware and software characteristics of switches (§2).

The network evolves during our study as new switches get added to datacenters and existing switches undergo proactive maintenance reboots, making it challenging to estimate the survival time of switches. Survival theory [7] provides a framework to estimate the survival time from failure logs of a continuously evolving network. Using the non-parametric Kaplan-Meier [6] and semi-parametric Cox Proportional Hazard [4] models, we identify the impact of switch characteristics, like their hardware SKU and operating system (OS), on the switch survival likelihood (§3). Based on this analysis, our key findings are as follows:

- Our root-cause analysis shows that 32% of switch failures can be attributed to hardware faults and 27% to unplanned power outages. These are the two leading causes of switch failures in datacenters. The majority of failed switches recover in less than 6 minutes (§2).
- Datacenter switches have a 2% chance of suffering a failure within 3 months of being in production. There is significant heterogeneity in the survival likelihood of switches manufactured by the three major switch vendors, with one vendor being twice as likely to fail compared to the others (§3 and §4).
- 17% of switch failures occur due to software bugs in vendor switch OSes. To mitigate their impact, the cloud provider has developed the SONiC [9] switch OS. We find that using the *same underlying hardware*, SONiC switches have a higher survival likelihood compared to vendor switch OSes. Moreover, this difference in survival likelihood widens over time – at the end of 3 months, SONiC switches have 1% higher survival likelihood than vendor switch OSes (§5), highlighting the efficacy of deploying SONiC in datacenters.

Implications. Datacenter reliability is well-studied in research and practice (§6). Our work complements previous findings by performing a *longitudinal* study of switch failures in an *evolving network* – with new hardware being provisioned daily. Additionally, our focus is to identify characteristics of reliable switches, *e.g.,* hardware vendors and switch operating systems. To this end, our root cause analysis shows that some switch vendors are more likely to run into hardware bugs compared to others. Hardware bugs cannot

be resolved from OS upgrades alone, making them harder and time consuming to mitigate. Second, we estimate the instantaneous rate of failure, or hazard rate, of switches and find that switches from certain vendors are more likely to fail compared to others. Thus, our analysis *informs the purchase of new vendor switches* and for the ones already deployed, it can *inform the placement of data and services* [3] to minimize the impact of inevitable switch failures. We end with two case studies of switch failures in cloud datacenters highlighting the process of isolating, root-causing and mitigating switch failures in production networks (§7).

2 IDENTIFYING SWITCH FAILURES

We define a spontaneous and unplanned interruption in the normal function of network switches as a failure. For this study, we scan all datacenter switches (*i.e.,* top-of-rack, leaf and spine) every 6 hours to collect the output of the *show reload cause* command-line interface (CLI) command. We parse the output to identify the *cause* and *timestamp* of the last reload of the switch.

While there are other ways of inferring switch reboots at scale, we use the output of the reload cause command for two reasons. First, this command is supported by all switch OSes in our datacenters, allowing complete measurement coverage. Second, reboot signals that rely on syslogs and SNMP traps are noisy in nature. For instance, coldStart traps [10] are raised both when the switch reboots and when the SNMP process restarts. Thus, we found that periodically querying the switches to infer potential reloads provides the ground truth for failure events.

In all, there are over 180,000 datacenter switches that were periodically scanned. Due to the large number of switches, each scan can take several hours to complete. Thus, we query each switch once in 6 hours and log a reboot that occurred in the time since the previous scan. A limitation of periodic scanning is that our scans consider multiple reboots of a device within 6 hours as a single reboot. The timestamped collection of all switch reboots inferred from our scans is our primary dataset (**failure-dataset-main**). We note that due to the redundancy in the datacenter topology and failure mitigation mechanisms like live VM migration, most switch failures do not hamper client services.

We manually analyzed a subset of switch failures in-depth using syslogs and signals from monitoring agents. These events were chosen for manual analysis since they caused significant disruption in services, making them higher priority than the others. We annotated failures in this subset with their root causes and durations. This forms the supplementary dataset of our analysis (**failure-dataset-supplement**). Overall, we analyze switch failures in the datacenters of a large cloud provider from June 24^{th} to September 24^{th}, 2020.

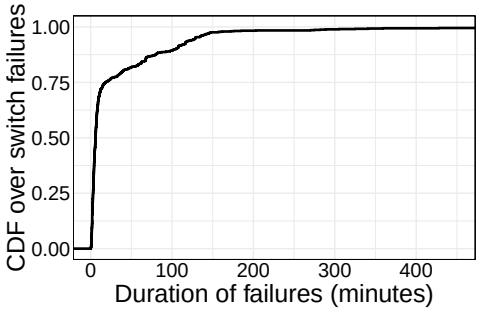

Figure 1: The duration of switch failures.

These switches are located across 130 datacenters globally, consisting of over 180,000 ToR, leaf and spine switches. In the 3-month analysis period, we observed 4,681 switch failures.

2.1 Measuring failure durations

For a subset of switch failures, we estimated the duration of the failure (**failure-dataset-supplement**) using reachability tests to the switch. Figure 1 shows the distribution of durations of switch failures. Nearly 50% of failures lasted for six minutes or lesser. A small number of failures lasted for several hours, accounting for the isolation, removal and replacement of the failed device.

2.2 Identifying failure root causes

Switch interruptions can occur due to planned maintenance activities. For instance, replacement of faulty power supplies and OS upgrades can cause switch outages during the maintenance time window. We differentiate planned maintenance activities from *unplanned failure events* in our analysis.

To differentiate failures from planned maintenance, we analyze the maintenance logs of all switches in the datacenter. These logs contain plans for maintenance activities scheduled for the switch. If a maintenance activity coincides with a failure event, we consider that event as *planned*. We also augment the maintenance logs with authentication, authorization and accounting (AAA) agent logs on switches [1]. The AAA agent keeps a timestamped log of all CLI commands issued on the switch by authorized users. Interruptions that follow within 30 minutes of the execution of a reload command on the switch CLI are considered planned events and are excluded from our analysis. We allow a maximum of 30 minutes to elapse between the AAA log and the planned switch outage events to account for the time lag in the collection of AAA logs to the centralized database. For the rest of this study, we focus only on unplanned switch failures.

In addition to the broad categorization of planned and unplanned events, we conduct a detailed root cause analysis (RCA) for a subset of 331 failures (**failure-dataset-supplement**). We manually analyze the syslogs of the failed

switches in cooperation with the support staff of switch vendors and assign each failure into a root cause bucket.

Hardware vs. software failures. Two of the root cause buckets we define are *hardware* and *software* failures. Both hardware and software failures can manifest in the form of process crash logs and stack traces on the faulty switch, making it hard to differentiate between them in practice. Additionally, even when the failure logs clarify that the cause is a hardware fault, there can be ambiguity about the real source of the error. For example, parity errors encountered on switches [2] can be caused by faulty hardware, bit flips *etc.*, but they can be mitigated in the switch software. Therefore, it is unclear if these faults should be attributed to the hardware or the software. To resolve the categorization ambiguity, we take input from the switch manufacturer's support team – if the manufacturer provides a software fix for the failure, we categorize the fault as a software failure. Categorization of failures software vs. hardware faults helps the cloud provider evaluate which vendors are preferable. Software faults can be resolved by applying patches issued by the vendors but hardware faults necessitate replacement of faulty devices. Thus, devices with fewer hardware faults are preferable for deployment. For fair comparison between vendors, we allow the vendors' resolution of the failure to dictate the categorization of the root-cause.

Figure 2 shows the percentage of failures in each root cause bucket. We find that approximately 32% of switch failures are caused by hardware issues. Unplanned power loss is responsible for 28% of failures. In some cases, it is hard to conduct the post-mortem root cause analysis due to the absence of sufficient clues in the syslogs of the failed switch. In the absense of sufficient evidence, the failed switch is sent to the manufacturer for a detailed RCA, and we classify the failure in the "pending root cause" category. If, despite the efforts of the cloud provider and the switch manufacturer, the root cause cannot be found, we classify the failure category as "unknown". In 5% of the cases, the fault is found in the connection between the switch and servers ("server related"). Switches with unknown failure causes or ones that cannot be mitigated through a software patch undergo a process of return merchandise authorization (RMA). As part of the RMA, the failed switch is returned to the vendor and is replaced with a new one (more in §7).

3 ESTIMATING SWITCH SURVIVAL TIME

Our goal is to estimate the expected time-to-failure, or survival time, of datacenter switches and quantify the factors that affect it. This is hard as datacenter networks continuously evolve: new switches are deployed, switches reboot during maintenance and suffer spontaneous failures. A naive estimate of time-to-failure using switch counts per time window fails to capture two features of the estimation task:

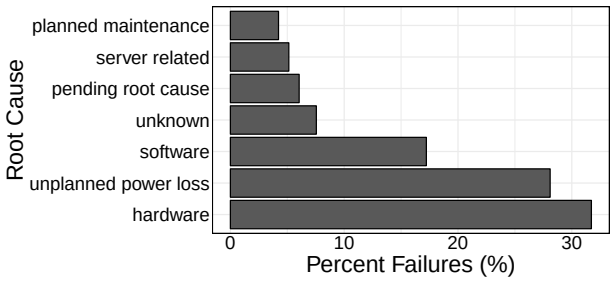

Figure 2: Root causes of switch failures.

1. Not all switch interruptions are failures: Maintenance reboots of switches pre-empt our observation of their time-to-failure. The naive estimator of time-to-failure could handle maintenance reboots in one of three possible ways: (1) In estimating time-to-failure, the naive estimator can ignore all devices that undergo maintenance reboots during the study. This causes several data points to get discarded from the analysis since maintenance reboots are common in datacenters. (2) The naive estimator can assume that there is no interruption in the switch lifetime even if it rebooted due to maintenance. This risks *overestimation* of the time-to-failure of switches. (3) Finally, by considering planned reboots as failures, the naive estimator risks *underestimating* the time-to-failure of switches. Thus, these strategies are inadequate in addressing preempted observation of switch lifetimes.

2. New switches get deployed during the analysis: During the analysis period, new switches get deployed in the datacenters and our observation of their lifetime starts once they get deployed. The naive estimator must account for the limited view into their time-to-failure.

Survival analysis [7] provides a framework for analyzing failure events that incorporates these partial observations of switch survival times. Traditionally used in biostatistics to compute the survival likelihoods of patients in different treatment groups, we use survival analysis to estimate time-to-failure of switches and quantify the impact of switch characteristics on their survival likelihood.

Survival dataset of switches. We use **failure-dataset-main** (Section 2) and convert it to the traditional survival analysis form. Each switch in the study cohort has a unique ID and *status* field. The status field is the key outcome variable of the study, it is 1 if the switch has *failed* in the study period and 0 otherwise. The status is 0 if the switch did not fail *in the study period*. Since, the failure was not observed in the study, this data point is said to be *censored* from the analysis. Censoring is an important concept in survival analysis which addresses cases where we have partial observations of switches' lifetimes. The other important field in the survival dataset is the *observation time* of a switch: the duration for

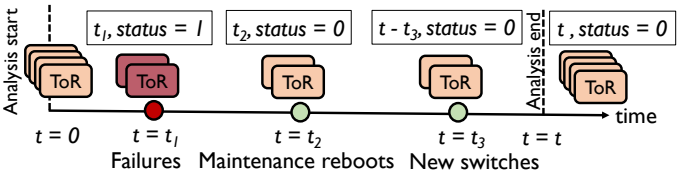

Figure 3: We use survival theory to estimate the time-to-failure of switches. Partial observations of time-to-failure due to maintenance reboots at t_2 are considered *censored*. Censored data points are represented with status = 0 and observation time of t_2 in the survival dataset.

Switch ID	Status (d)	Observed Duration (t)	Vendor Type
Switch-1	1	100 minutes	VENDOR-1
Switch-2	0	200 minutes	VENDOR-2
..	..	..	..

Table 1: Example rows from the survival dataset of switches. Switch-1 fails after 100 minutes of observation and Switch-2 gets rebooted after 200 minutes of observation (censored).

which the switch was observed in the study. The observation time of a failed switch stops when it fails. If the switch gets censored at the end of the study, its observation period stops at the end of the study (Figure 3). Along with switch ID, status and observation time, the survival dataset includes explanatory features of the switch: hardware vendor, OS version *etc.* (shown in Table 1).

Incorporating maintenance outages. We account for maintenance outages using the concept of *right censoring* in survival theory. Switch restarts as part of routine maintenance are considered censored observations ($status = 0$) with the observation period ending at the restart time. When the switch comes back up after the maintenance, it is included in the cohort as a new device with a different switch ID and its observation period starts when it is up and running again. In this manner, we do not discard censored data points in our analysis.

Incorporating repeated switch outages. In traditional survival analysis, the event of interest can occur to the subjects once during the study (*e.g.,* death, remission from cancer). However, switches recover from failures either on their own or following a fix – upgrade of the power supply or fan, installation of an OS patch *etc.*. Recovered switches can fail again. To incorporate repeated failure events, once a switch recovers, we assign it a unique ID, different from the original switch, and begin its observation period when it has recovered from the failure. Of the switches that failed at least once in our analysis, the 98^{th} percentile failure count per switch during the study duration is 34. We found that 8 switches were outliers in terms of their failure counts and excluded the outliers from the analysis to not bias the overall survival rates towards these devices.

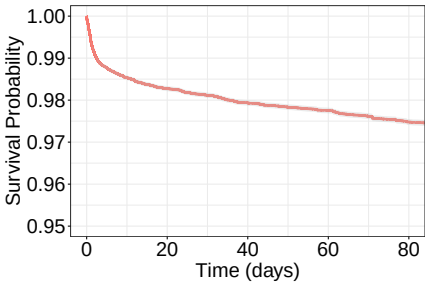

Figure 4: Kaplan Meier survival estimate of datacenter switches, shaded region shows 95% confidence intervals.

3.1 Kaplan Meier survival estimator

In this section we use the survival dataset of switch failures to estimate switch survival time. Let T be the time for which a switch is uninterrupted in its operation, or the survival time. Survival time is a random variable and the survival function, $S(t)$, of a cohort of switches measures the probability that $T > t$. The Kaplan-Meier estimator [6] is a non-parametric statistic used to estimate the survival function, $\hat{S(t)}$:

$$\hat{S(t)} = \prod_{i:t_i \leq t} \left(1 - \frac{d_i}{n_i}\right) \quad (1)$$

where d_i is the number of switch failures that occurred at time t_i and n_i is the size of the risk set at t_i. The risk set consists of switches in the study cohort that have not failed up to time t_i. Censored observations change the size of the risk set, n_i, avoiding the pitfalls of the naive estimator. Since the number of failures (d_i) increase with time, $\hat{S(t)}$ is a non-decreasing function. Figure 4 shows the Kaplan Meier survival curve of switches in our datacenter. At the start of the study, the survival probability is 100% since no failures have occurred. As switch failures are observed, the survival probability drops to 98.5%. This shows that datacenter switches have a 98.5% chance of staying uninterrupted for at least 3 months since deployment in production.

4 PROPORTIONAL HAZARD MODEL

To decompose the contributions of known switch characteristics on the survival time of switches, we use the Cox Proportional Hazards (PH) model [4]. The Cox PH model extends the Kaplan-Meier estimator by quantifying the impact of switch characteristics on survival time. The model estimates the hazard rate – the instantaneous failure rate of switches, as a function of observation time and explanatory variables of the switch. The hazard rate is modeled as:

$$h(t) = h_0(t)e^{(b_1x_1+b_2x_2+..)} \quad (2)$$

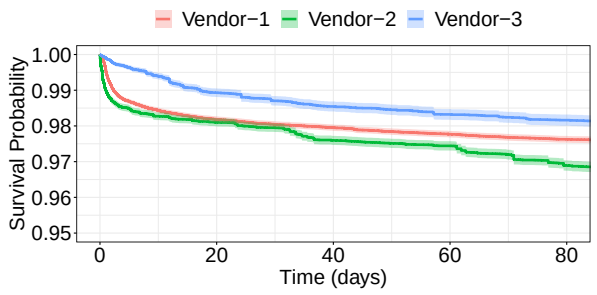

Figure 5: Kaplan Meier survival curves for switches belonging to three main switch vendors. VENDOR-3 has a lower survival likelihood than VENDOR-1 and VENDOR-2.

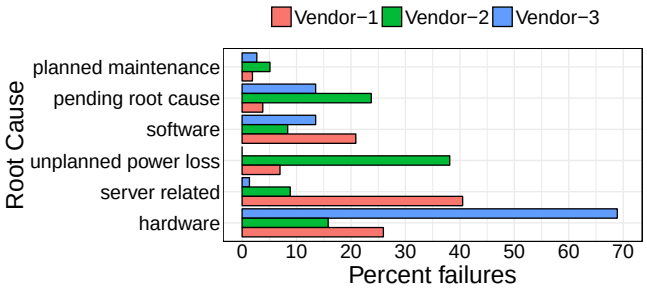

Figure 6: Root causes of failures for different hardware vendors. 70% of all failures in the switches manufactured by VENDOR-3 were hardware issues.

where $h_0(t)$ is the base hazard rate, x_i are the values of the switch characteristics (or covariates) and b_is are the coefficients of the covariates. The base hazard rate is the part of the estimator that varies with time. The exponential only depends on switch characteristics. These coefficients measure the impact of a covariate on the hazard rate of the population. We use a categorical covariate to represent the switch vendor and its corresponding coefficient in the Cox model will describe the impact of switch vendor on failure rate.

The quantity e^{b_i}, called the *hazard ratio* of covariate x_i, compares the increased hazard to switches due to x_i. $b_i > 0$, equivalently the hazard ratio $e^{b_i} > 1$, implies that an increase in the value of x_i leads to an increased hazard to the population. For categorical variables, the hazard ratio is computed in reference to a *base* population. For instance, the hazard ratio of a switch vendor is computed using another vendor's switches as the base population.

4.1 Impact of hardware on resilience

The datacenters we analyze have switches from 3 major switch manufacturers. We anonymize their names to VENDOR-1, VENDOR-2 and VENDOR-3. In this section we compare the reliability of switches of the three vendors using **failure-dataset-main** (§2). We compute the Kaplan Meier estimate for the three switch vendors (Figure 5) and find that while two vendors have similar survival likelihoods, VENDOR-2 is more likely to fail than the others. The trained PH hazard model quantifies this using the hazard ratio and finds that the hazard rate of VENDOR-2's switches is 2X the hazard rate of VENDOR-3 with a highly significant p-value $< 1e-10$.

Since we conducted manual root cause analysis for switch failures in **failure-dataset-supplement** (§2), we categorize root causes of failures for each of the three vendors separately. Figure 6 shows the percentage of failures in each root cause category for the 3 vendors. We find that nearly 70% of VENDOR-3's failures are hardware issues. Hardware issues are harder to resolve since software upgrades and fast reloads do not fix the problem.

5 REDUCING THE HAZARD

Across all vendors, 17% of failures were caused by software bugs (Figure 2). To reduce these failures, the cloud provider has been actively developing an in-house switch operating system, SONiC [9]. During our study period, nearly 75% of the aggregation switches in the cloud datacenters were running SONiC on existing VENDOR-1 hardware. This gave us a unique opportunity to compare the resilience of SONiC and vendor switch OSes. These switches use the same underlying hardware manufactured by VENDOR-1 with the only difference being the switch OS (SONiC vs. VENDOR-1's OS). We show the Kaplan Meier survival curves of the two types of switches (Figure 7) and find that SONiC switches are significantly more reliable despite having the same underlying hardware. With time, the gap in reliability widens and at the end of 3 months, the survival likelihood of SONiC switches is 1% higher than that of non-SONiC switches.

To quantify the difference in survival time and hazard rate of switches running SONiC vs. the vendor OS, we train a Cox PH model on aggregation switches manufactured by VENDOR-1. The sub-grouping helps us focus on the key question by removing other characteristics that could interact with the effect of switch OS on hazard rate. The trained Cox PH model has a coefficient of −7.2 for SONiC aggregation switches in comparison with non-SONiC aggregation switches. Consequently, the hazard ratio is 0.0007 with a highly significant p-value $< 2e-16$. The Concordance index (CI) of the Cox PH model is a generalization of the Receiver operating characteristic (ROC) curve that takes into account censored data points [7]. The CI of the Cox PH model we learn is 0.9 implying that with high confidence, SONiC switches have a significantly lower likelihood to fail compared to non-SONiC switches in our datacenters.

Our analysis shows that replacing vendor switch OSes with SONiC has been beneficial in improving the resilience of datacenter switches. We attribute the resilience of SONiC to the rapid develop-test-deploy cycle made possible by

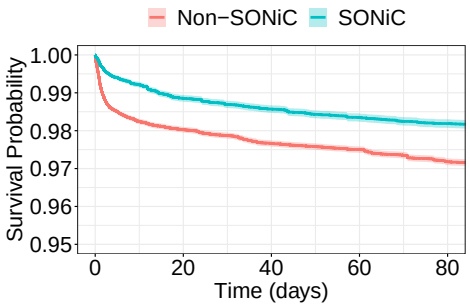

Figure 7: KM survival curves for SONiC and vendor switch OS running on the same switch hardware.

the in-house development of the software. Indeed, vendor software updates and patches are rolled out over longer timescales (*e.g.,* several months). This leads to known issues re-occurring on devices that have not yet been patched with the vendor-supplied fixes. In contrast, SONiC failures are root-caused and fixed over short timescales due to in-house expertise and development teams.

6 RELATED WORK

The work of Gill et. al. is closely related to our study. In their work, authors studied failure of devices and links in datacenters [5]. However, their analysis does not consider the evolution of failure rate with time and the impact of censored data points when new devices get added or maintenance reboots occur. As in [5], we also find that datacenter switches are highly reliable, but, we perform a *longitudinal* study of failure rates to establish the relationship between failure and device lifetime in production. While [5] focusses on a broader variety of device failures (*e.g.,* load balancers), they do not root cause the failures and compare the impact of switch OS and hardware on failure probabilities. Recently, researchers have measured the impact of different components (*e.g.,* devices, configurations *etc.*) on the overall datacenter network reliability at Facebook [8]. In their work, authors have discussed mean time to failure (MTTF) for edge nodes and links in the WAN backbone. Our focus is on identifying characteristics that impact the survivability of *datacenter switches* for informing hardware purchase decisions. In their work [11], authors studied the failure of middleboxes in datacenters and analyzed the causes of their low reliability. Our goal is to identify hardware and software configurations that work well and inform future purchase and deployment decisions. Finally, failure likelihoods change with time since deployment in production and our analysis captures these changes using survival likelihood estimates.

7 DISCUSSION

Rapid detection and mitigation of switch failures is crucial to datacenter networks' incident response. When a switch fails in production datacenters, a team of engineers isolate the device and migrate traffic away from it. If the root cause of the failure is a known bug in the switch OS, the switch OS is upgraded and the switch re-joins the network. If the root cause is a known hardware issue, an operations team finds a replacement for the failed switch, either in the datacenter inventory or from the switch vendor. If the root cause is not clear, the switch is sent back to the vendor for in-depth analysis. As the diagnosis can take weeks, the failed switch is replaced. We illustrate this process with two case studies on switch failures that occurred in our datacenters.

The case of uncorrectable error correcting codes (ECC) Switches of VENDOR-1 began exhibiting a problem signature where they spontaneously rebooted in production and the *reload-cause* after reboot was marked *unknown*. The syslogs did not shed light on the root cause and the devices had to be sent to the vendor for analysis. The vendor found that the reloads happened due to an uncorrectable ECC error on the switch. This bug impacted several CPU models and it was found that the reloads were unpredictable. VENDOR-1 issued a patched BIOS image which updated the memory controller settings. The new BIOS was rolled out incrementally on the impacted switches as part of the mitigation process.

The case of CPLD failure. In late 2019, several VENDOR-3 manufactured switches exhibited a problem signature where all switch ports and linecards went down and were not being detected by the switch OS. The in-house analysis of switch logs did not improve our understanding of the problem and the failing devices were returned to the manufacturer for further analysis. After a few weeks, the manufacturer found that a complex programmable logic device (CPLD) was faulty. For a fraction of such devices in the network, an upgrade of the CPLD firmware and fast reboot was sufficient to fix the problem. A fast reboot disrupts the switch data plane for less than 30 seconds, causing minimal impact to client traffic. Remaining VENDOR-3 switches susceptible to the CPLD bug had to undergo return material authorization (RMA).

8 CONCLUSION

In this work we analyzed switch failures in datacenters from the lens of survival theory. We estimate the survival times and hazard rates of switches in the datacenters of a large commercial cloud provider. Our analysis shows that the two leading causes of switch failures in datacenters are hardware faults and unplanned power loss. Most failures are resolved quickly with the median duration of switch failures being six minutes. We find that one of the major vendor's switches are 2X more likely to fail than others. Finally, we show that by developing an in-house switch OS, SONiC, the cloud provider we analyze has significantly improved the resilience to failure of datacenter switches.

REFERENCES

[1] Arista Networks. AAA Configuration. https://www.arista.com/en/um-eos/eos-aaa-configuration. (Accessed on 2020-05-11).

[2] Arista Networks. EOS Central: Does this indicate a possible DRAM issue? https://eos.arista.com/forum/getting-ipt_crcerrpkt-and-jer_int_idr_mmu_ecc_1b_err_int-log-output-does-this-indicate-a-possible-dram-issue-on-bank-b/. (Accessed on 2020-05-11).

[3] P. Bodík, I. Menache, M. Chowdhury, P. Mani, D. A. Maltz, and I. Stoica. Surviving failures in bandwidth-constrained datacenters. *ACM SIGCOMM Computer Communication Review*, 42(4):431–442, 2012.

[4] D. R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–202, 1972.

[5] P. Gill, N. Jain, and N. Nagappan. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 conference*, pages 350–361, 2011.

[6] E. L. Kaplan and P. Meier. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*, 53(282):457–481, 1958.

[7] D. G. Kleinbaum and M. Klein. *Survival analysis*, volume 3. Springer, 2010.

[8] J. Meza, T. Xu, K. Veeraraghavan, and O. Mutlu. A large scale study of data center network reliability. In *Proceedings of the Internet Measurement Conference 2018*, pages 393–407, 2018.

[9] Microsoft Azure. Software for open networking in the cloud. https://azure.github.io/SONiC/. (Accessed on 2020-05-11).

[10] Net-SNMP. SNMP coldStart. http://net-snmp.sourceforge.net/docs/mibs/snmpMIB.html. (Accessed on 2020-05-11).

[11] R. Potharaju and N. Jain. 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*, IMC ?13, 2013.

[12] A. Singh, J. Ong, A. Agarwal, G. Anderson, A. Armistead, R. Bannon, S. Boving, G. Desai, B. Felderman, P. Germano, et al. Jupiter rising: A decade of clos topologies and centralized control in google's datacenter network. *ACM SIGCOMM computer communication review*, 45(4):183–197, 2015.