The Network is Reliable

An informal survey of real-world communications failures

25 March 2024

Dr. Jacob Hochstetler
Distinguished Engineer, Vice President, Fidelity Investments
Clinical Assistant Professor, University of North Texas

Agenda

- Introduction
- Rumblings From Large Deployments
- Datacenter Network Failures
- Cloud Networks
- Hosting Providers
- Wide Area Networks
- Global Routing Failures
- NICs and Drivers
- Application-Level Failures
- Where Do We Go From Here?
- Conclusion

Introduction

- Published in Associated for Computing Machinery's *Queue* (acmqueue)

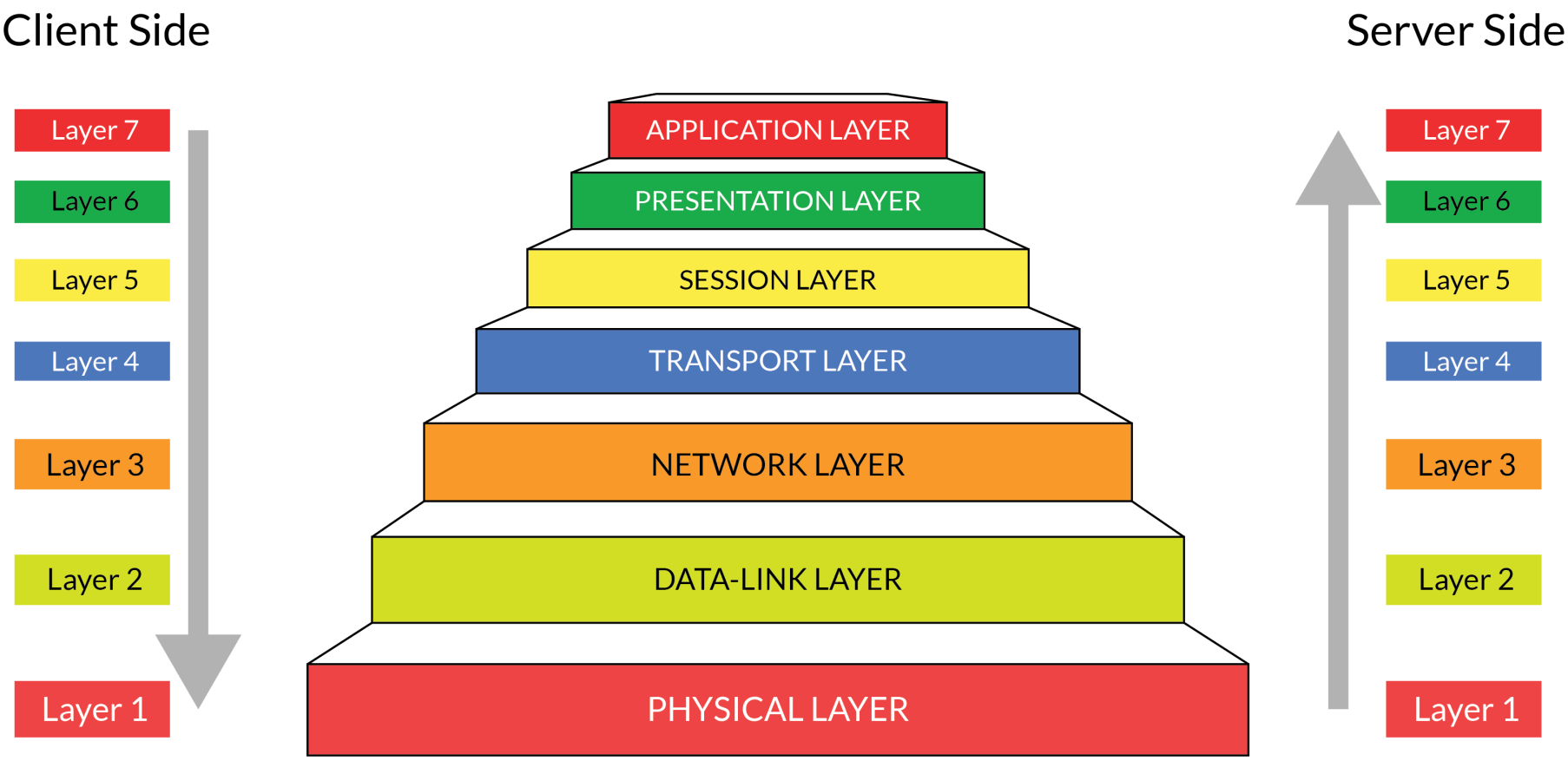
```
"the ACM's magazine for practicing software engineers, written by engineers for engineers"
```

- Queue, Volume 12, Issue 7 (September 2014), pp 20–32
- Authors: Peter Bailis (UC Berkeley) / Kyle Kingsbury (Jepsen Networks)

1. The paper explores the misconceptions of distributed computing, particularly focusing on Peter Deutsch's "Eight fallacies of distributed computing," with an emphasis on the fallacy of network reliability.
2. It emphasizes the need to grasp network behavior and its effect on distributed program design and function, due to the inherent reliance on shared communication channels.

Introduction: Understanding Network Reliability

- *Principle:* Network reliability is crucial for the effectiveness of distributed systems.
- *Common fallacy:* "The network is reliable" leads to design challenges.



Open Systems Interconnection model (OSI model) reference model for how applications communicate over a network. 4

Introduction: What is distributed computing and why does it matter?

> Using *multiple* computers to solve tasks together over a *network*.

These systems are crucial to modern life, because physical limitations prevent a single computer from performing all tasks, and data locality makes many tasks impossible.

Almost all modern applications involve distributed computing in some form, e.g.:

- **Cloud Computing**: Amazon Web Services (AWS), Microsoft Azure
- **Content Delivery Networks (CDNs)**: Akamai, Cloudflare
- **Blockchain and Cryptocurrencies**: Bitcoin, Ethereum
- **Distributed Databases**: Cassandra, MongoDB, CockroachDB
- **Grid Computing**: SETI@home, Folding@home, HPC
- **Internet of Things (IoT)**: Smart home systems, Industrial IoT platforms
- **Peer-to-Peer (P2P) Networks**: BitTorrent, IPFS

Introduction: "The network is reliable"

The "Fallacies of distributed computing" are:

```
1. The network is reliable;
2. Latency is zero;
3. Bandwidth is infinite;
4. The network is secure;
5. Topology doesn't change;
6. There is one administrator;
7. Transport cost is zero;
8. The network is homogeneous.
```

Sun Microsystems' Peter Deutsch (1994).

Six “fallacies” directly pertain to limitations on networked communications.

Introduction: Impact of Network Reliability on Distributed Systems

Challenge: Lack of evidence for comparing network and application reliability.

- **Limited data**: Difficult to track the end-to-end effect on applications.
- **Evidence is deployment-specific**: Often tied to specific vendors, topologies/designs.
- **Lack of transparency**: Organizations rarely share specifics about network behavior.
- **Complex failure modes**: Distributed systems are designed to resist failure:
  - Leading to silent degradation.
  - Delayed problem recognition.
  - Gray failures (partial failures).

Introduction: Survey of Cases

- Rumblings From Large Deployments
- Datacenter Network Failures
- Cloud Networks
- Hosting Providers
- Wide Area Networks
- Global Routing Failures
- NICs and Drivers
- Application-Level Failures

Rumblings from Large Deployments: Background

- **Key Players**: Insights from companies with large-scale distributed systems.
- **Challenges**: Operational experiences of managing massive infrastructures.
- **Network Partitions**: Highlighting the concern of "blast radius" in these deployments.

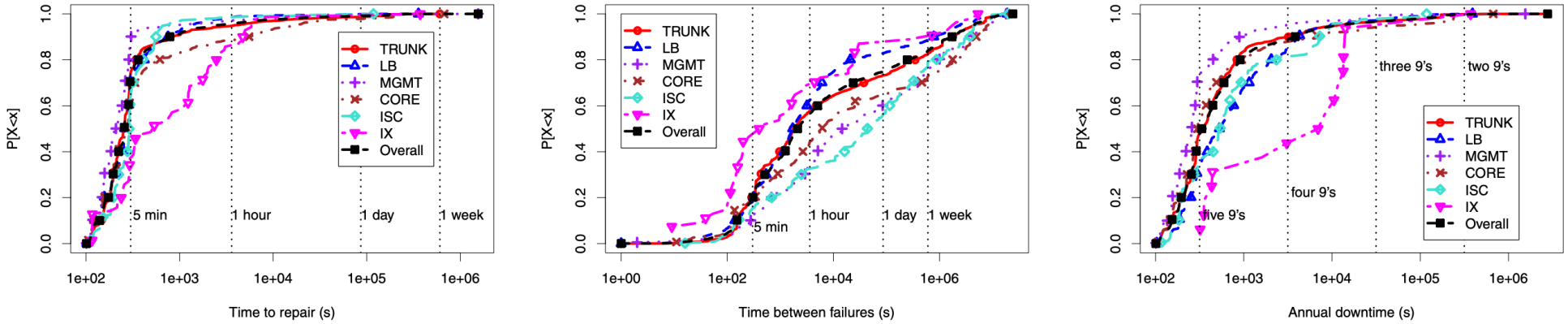
Image generated using Adobe Photoshop.

Rumblings From Large Deployments: Links

- Microsoft Data-Center Study: Highlighted average failure rates of devices and links per day, and the limited impact of network redundancy on traffic improvement.
- HP Labs Managed Networks: Revealed the percentage of support tickets related to connectivity issues and the median incident duration for different priority levels.
- Google Chubby: Identified the causes of outages, including network maintenance and connectivity problems, and their duration.
- Google's Design Lessons from Distributed Systems: Outlined the typical first-year challenges for a new Google cluster, including rack issues, network maintenance events, and router failures.
- Amazon Dynamo: Emphasized the importance of designing for network partitions and the rejection of traditional replicated relational database systems.
- Yahoo! PNUTS/Sherpa: Discussed the limitations of strict adherence to timeline consistency and the shift towards weaker consistency alternatives for better availability during partitions.

Rumblings from Large Deployments: Microsoft Data-Center Study

- Studied network failures in Microsoft's datacenters.
- Found average failure rates of 5.2 devices and 40.8 links per day.
- Median time to repair: approximately five minutes (max one week).
- Estimated median packet loss of 59,000 packets per failure.
- Network redundancy improves median traffic by only 43%.



(a) Time to repair for links **(b) Time between failures for links** **(c) Annualized downtime for links**

Figure 9: Properties of link failures that impacted network traffic.

Rumblings from Large Deployments: HP Labs Managed Networks

- Examined network failures in HP's managed networks.
- "Connectivity"-related tickets accounted for 11.4% of support tickets.
- Median incident duration: 2 hours and 45 minutes for highest-priority tickets.
- Median duration for all tickets: 4 hours and 18 minutes.

Table 4: Product type for the 14% of incidents that matched a Netcool record.

Product type	% of tickets	Median duration (HH:MM)	% of URP minutes
router	3.751	10:10	4.110
switch	2.658	18:38	2.738
router/switch	1.399	04:00	0.179
firewall	1.264	07:57	0.764
Not specified	0.531	07:42	0.234
Loadbalancer	0.529	1 day	1.101

Table 5: Classes of incidents (by severity) in customer networks managed by HP.

	Severity			
Class	1	2	3+	All
WAN	18.5%	2.9%	4.7%	4.8%
LAN	17.5%	3%	11.5%	11.9%
Hardware	31%	62%	37.3%	39%
Software	4.5%	3.7%	16.9%	15%
Config	4.3%	3.1%	6.1%	6%
Connectivity	19%	14.9%	11.2%	11.4%

Rumblings from Large Deployments: Google Chubby

- Described the design and operation of Google's distributed lock manager, *Chubby*.
- Outlined the root causes of 61 outages over a few weeks (700 cell-days) of data:
  - Outages due to maintenance that shut down the datacenter excluded.
  - Causes: network congestion, maintenance, overload, and errors due to operators, software, and hardware.
  - Nine outages lasted more than 30 seconds, caused by network maintenance and connectivity problems.

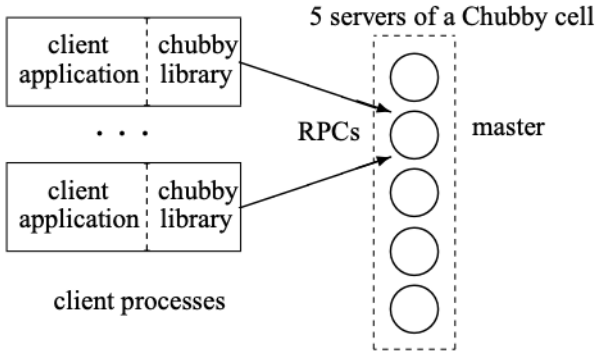


Figure 1: System structure

Rumblings from Large Deployments: Google's Design Lessons

- Insights from Google Fellow Jeff Dean on building large-scale distributed systems.
- Typical first-year challenges: bad racks, network maintenance events, router failures.
- Highlights challenges of split-brain issues surfacing after network partitions fail.

~0.5 overheating (power down most machines in <5 mins, ~1-2 days to recover)
~1 PDU failure (~500-1000 machines suddenly disappear, ~6 hours to come back)
~1 rack-move (plenty of warning, ~500-1000 machines powered down, ~6 hours)
~1 network rewiring (rolling ~5% of machines down over 2-day span)
~20 rack failures (40-80 machines instantly disappear, 1-6 hours to get back)
~5 racks go wonky (40-80 machines see 50% packetloss)
~8 network maintenances (4 might cause ~30-minute random connectivity losses)
~12 router reloads (takes out DNS and external vips for a couple minutes)
~3 router failures (have to immediately pull traffic for an hour)
~dozens of minor 30-second blips for dns
~1000 individual machine failures
~thousands of hard drive failures
slow disks, bad memory, misconfigured machines, flaky machines, etc.

Long distance links: wild dogs, sharks, dead horses, drunken hunters, etc.

"Typical first year for a new cluster", page 10, network failures boxed in blue.

Rumblings from Large Deployments: Amazon Dynamo

- Frequently cites the incidence of network partitions as a key design consideration.
- Rejected designs from traditional replicated relational database systems.
- *"Dynamo is designed to tradeoff consistency for availability."*

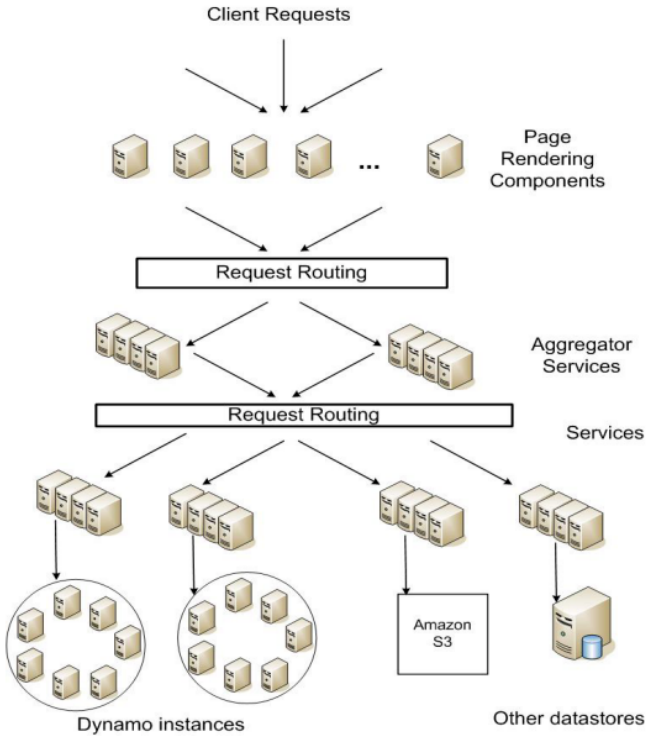


Figure 1: Service-oriented architecture of Amazon's platform

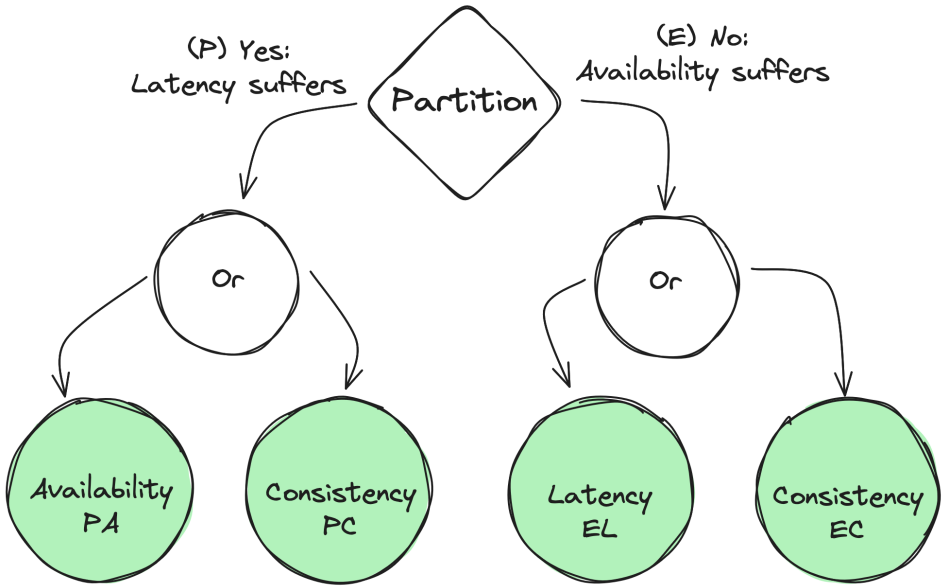
Table 1: Summary of techniques used in *Dynamo* and their advantages.

Problem	Technique	Advantage
Partitioning	Consistent Hashing	Incremental Scalability
High Availability for writes	Vector clocks with reconciliation during reads	Version size is decoupled from update rates.
Handling temporary failures	Sloppy Quorum and hinted handoff	Provides high availability and durability guarantee when some of the replicas are not available.
Recovering from permanent failures	Anti-entropy using Merkle trees	Synchronizes divergent replicas in the background.
Membership and failure detection	Gossip-based membership protocol and failure detection.	Preserves symmetry and avoids having a centralized registry for storing membership and node liveness information.

Table 1 presents a summary of the list of techniques Dynamo uses and their respective advantages.

Rumblings from Large Deployments: Yahoo! PNUTS/Sherpa

- Designed as a distributed database operating in geographically distinct datacenters.
- Originally supported strongly consistent "timeline consistency" operation.
- Shifted to weaker consistency due to network partitioning/server failures.



PACELC: If there is a partition (P) how does the system tradeoff between availability and consistency (A and C); else (E) when the system is running as normal in the absence of partitions, how does the system tradeoff between latency (L) and consistency (C)?

Datacenter Network Failures: Background

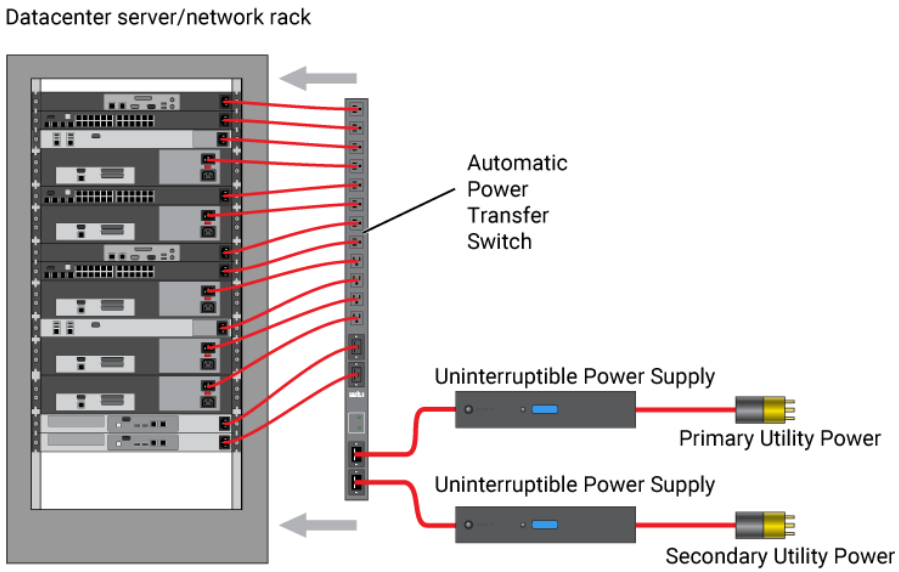
- Datacenters are prone to diverse network failures:
  - power failures
  - misconfiguration
  - firmware bugs
  - topology changes
  - cable damage
  - malicious traffic
  - cows
  - shark attacks
- Redundancy does not always prevent network failures.

Datacenter Network Failures: Links

- Fog Creek Software (n/a): Power failure on both redundant switches.
- Switch split-brain caused by BPDU flood (n/a): A loop formed between switches during a network reconfiguration.
- Bridge loops, misconfiguration, broken MAC caches.: GitHub's network topology change.
- MLAG, Spanning Tree and STONITH: MLAG failure, HA failover, and STONITH.

Datacenter Network Failures: Power Failure on Both Redundant Switches

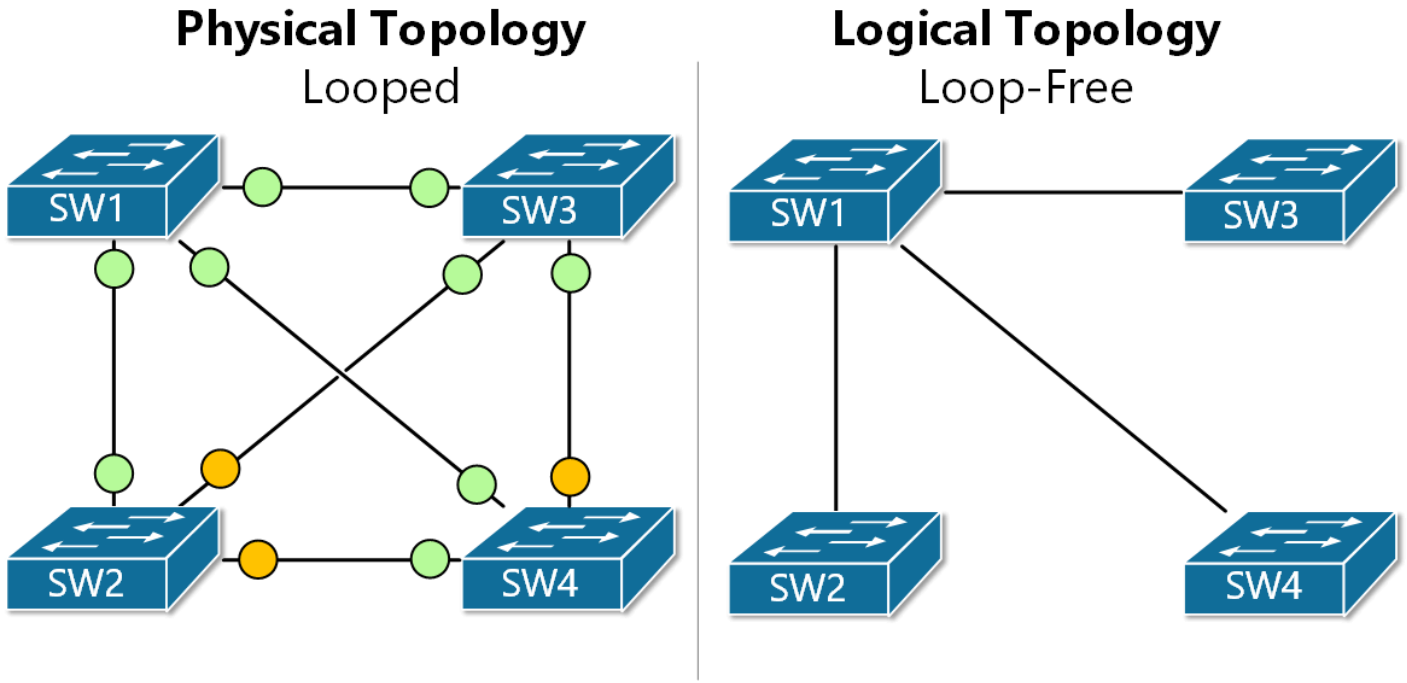
- **Redundancy Limitation**: Even with redundant top-of-rack switches, a power distribution unit failure led to service loss for Fog Creek customers.
- **Unexpected Power Loss**: A second switch also lost power for undetermined reasons, isolating neighboring racks and taking down On Demand services.



A typical datacenter's server / network rack setup with dual uninterruptible power supplies (UPS), with redundancy (hopefully) through automatic power transfer switches connected to primary and secondary utility power sources for fail-safe operation.

Datacenter Network Failures: Switch Split-brain by BPDU Flood

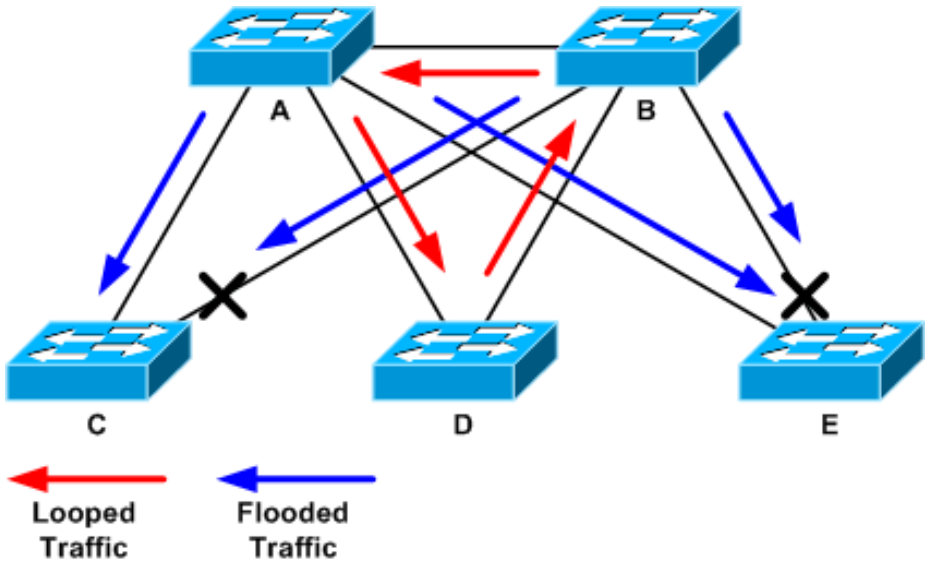
- **Network Reconfiguration**: A loop formed between switches during an attempt to improve reliability, leading to a split-brain scenario.
- **Spanning-tree Flap**: A multi-switch BPDU flood, contrary to the BPDU standard, resulted in two hours of total service unavailability for Fog Creek Software.



Redundant LAN with Spanning-Tree with blocked ports in orange and open ports in green.

Datacenter Network Failures: Bridge Loops, Misconfiguration, Broken MAC Caches

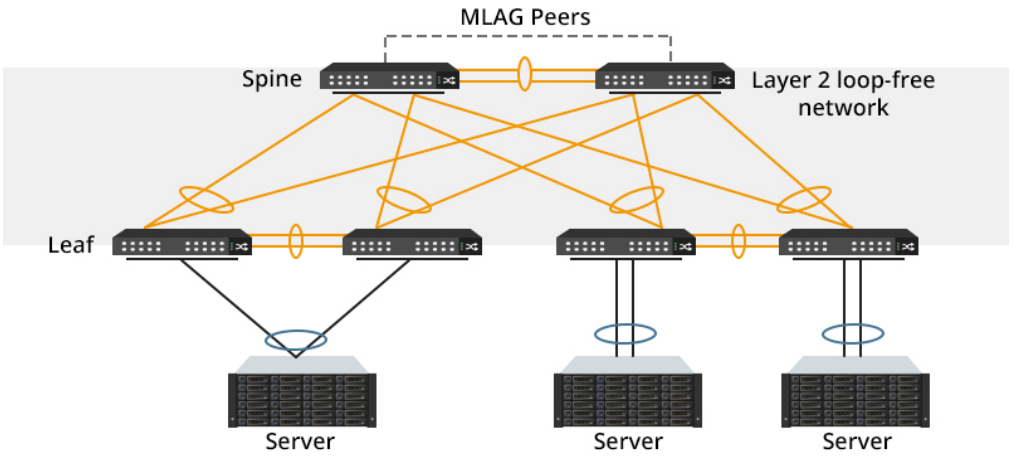
- **Network Topology Change**: GitHub's installation of aggregation switches to reduce latency caused bridge loops and link disabling.
- **Firmware Bug**: A misconfigured switch and a firmware bug in MAC address cache updating led to 18 minutes of downtime.



A network bridge loop (switch loop) scenario: Traffic between Switches A and B becomes looped (red arrows), leading to flooded traffic (blue arrows) that overwhelms Switches C, D, and E.

Datacenter Network Failures: MLAG, Spanning Tree, and STONITH

- **MLAG failure**: An update on an aggregation switch led to a 90-sec network partition.
- **HA failover**: The partition caused HA file servers to issue STONITH messages, resulting in both nodes being "shot" and files being unavailable.
- **Recovery**: File-server pair recovering took five hours, significantly degrading service.



A 2-tier spine-leaf multi-chassis link aggregation group (MLAG) topology with two switches acting as one. NIC bonding or Link Aggregation connects servers with multiple interfaces to active-active redundant leaf switches in an MLAG pair. At the spine layer, two FS datacenter switches form another MLAG pair, aggregating all uplinks and eliminating blocked ports for full interconnect bandwidth.

Cloud Networks: Background

- Cloud environments face challenges with transient latency, dropped packets, and network partitions.
- Network issues can have cascading effects on cloud services.

Image generated using Adobe Photoshop.

Cloud Networks: Links

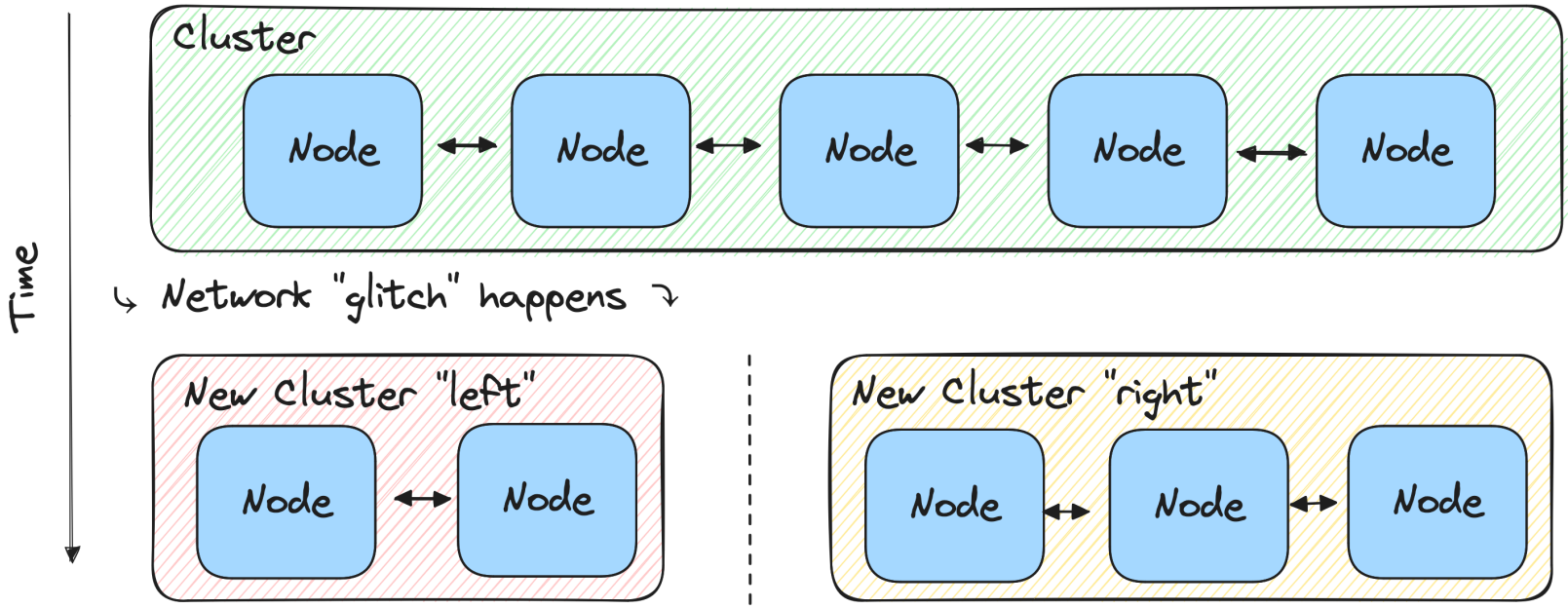
- Isolated MongoDB Primary on EC2: A network partition isolated a MongoDB primary, leading to data loss and rollback.
- Mnesia Split-brain on EC2: A network partition caused a split-brain scenario in a Mnesia cluster.
- EC2 Instability Affecting MongoDB and ElasticSearch (n/a): Network issues on EC2 caused partitions between front-end and back-end servers, leading to short but impactful outages.
- AWS EBS Outage: Incorrect routing policies during scaling operations led to a 12-hour EC2 outage and over 80-hour EBS disruption.
- Isolated Redis Primary on EC2: A network partition isolated Twilio's Redis primary, causing secondary nodes to overload the primary during reconnection.

Cloud Networks: An Isolated MongoDB Primary on EC2

- **Network Partition**: EC2 West's network issues caused a partition, isolating PRIMARY from its SECONDARIES in a 3-node `replSet`.
- **Data Loss**: When the old primary rejoined after 2 hours, it caused a rollback and write loss on the new primary.
- **Common Failures**: Large-scale MongoDB users frequently experience failover on EC2 due to network events.

Cloud Networks: Mnesia Split-brain on EC2

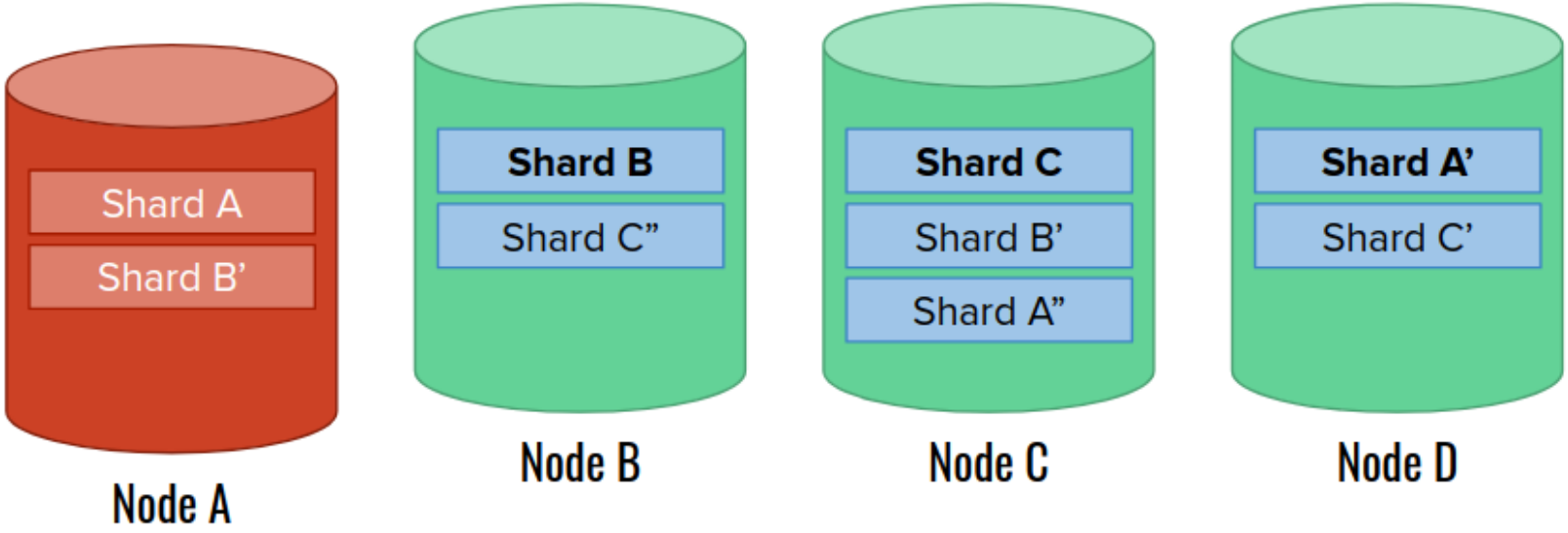
- **Outage Consequence**: Two Mnesia nodes stayed online but disconnected from each other, leading to data inconsistency and loss.
- **Operational Decision**: The team nuked one side of the cluster, underlining the need for better network partition recovery strategies.



How a split-brain scenario in an Mnesia cluster develops: A network 'glitch' results in a partitioned cluster, creating two separate 'left' and 'right' clusters, each with their own set of nodes that can no longer communicate with each other. 26

Cloud Networks: EC2 Instability Affecting MongoDB and ElasticSearch

- **Selective Disruptions**: Network issues on EC2 caused partitions between front-end and back-end servers, leading to short but impactful outages.
- **Service Impact**: Despite brief network glitches, resulting outages lasted 30 to 45 minutes with a corrupted index for ElasticSearch (OpenSearch).



Elasticsearch cluster mishap: Node A's missed heartbeat prompts shard redistribution to Nodes B, C, and D. Resharding typically starts after one minute, but Node A's quick (< 10s) reconnection causes index corruption due to interim writes27

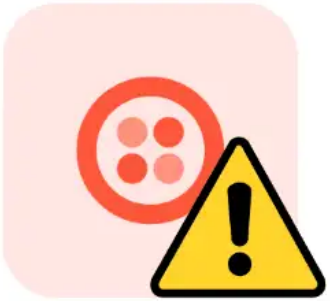
Cloud Networks: AWS EBS Outage

- **Massive Unavailability**: Incorrect routing policies during scaling operations led to a 12-hour EC2 outage and over 80-hour EBS disruption.
- **Compounded Failures**: Aggressive recovery efforts and a race condition in EBS mirrored the network congestion, furthering the service degradation.
- **Relational Database Service**: A failover protocol bug in Amazon's RDS prevented 2.5% of multi-AZ databases from switching AZs correctly during the outage.

Reddit's downtime mascot: An image displayed during the April 21, 2011 AWS EBS outage, informing users of service degradation with a promise of resolution and an apology for the inconvenience.

Cloud Networks: Isolated Redis Primary on EC2

- **Billing System Failure**: A network partition isolated Twilio's Redis primary, causing secondary nodes to overload the primary during reconnection.
- **Operational Missteps**: Restarting the Redis primary with an incorrect configuration file led it to read-only mode, triggering automatic overbilling.
- **Recovery Efforts**: Twilio restored proper service and user credits from an independent billing system after the Redis issue caused widespread overbilling.



1.1% of Twilio customers were overbilled for 40 minutes. E.g., Appointment Reminder reported that every SMS message and phone call it issued resulted in a $500 charge to its credit card, which stopped accepting charges after $3,500.

Hosting Providers: Background

- Hosting providers play a critical role in network reliability but can also experience failures.
- Network issues in hosted environments can lead to service disruptions and data inconsistencies.

Image generated using Adobe Photoshop.

Hosting Providers: Links

- An Undetected GlusterFS Split-Brain at Freistil IT: Localized packet loss and an undetected split-brain in a GlusterFS file system.
- **Network Failures at an Anonymous Hosting Provider (n/a)**: Frequent network partitions affecting 100-200 nodes.
- Pacemaker/Heartbeat Split-Brain Scenario (n/a): Two Linode VMs in a Heartbeat pair declared each other dead and contested for a shared IP.

Hosting Providers: An Undetected GlusterFS Split-Brain at Freistil IT

- **Localized Packet Loss**: Freistil IT experienced 50-100% packet loss in a data center due to a router firmware bug.
- **Undetected Split-Brain**: GlusterFS file system entered a split-brain state undetected, causing issues with website image delivery.
- **Self-Heal Challenges**: The GlusterFS self-heal algorithm failed to resolve inconsistencies, leading to a brief surge in network traffic and node overload during repair.

Hosting Providers: Network Failures at an Anonymous Hosting Provider

- **Frequent Partitions**: A major hosting provider experienced five network partitions over 90 days, affecting 100-200 nodes.
- **Connectivity Disruption**: Some partitions disrupted connectivity between the cloud network and the public Internet, while others isolated the cloud from the internal network.

Hosting Providers: Pacemaker/Heartbeat Split-Brain Scenario

- **Persistent Heartbeat Issue**: Two Linode VMs in a Heartbeat pair declared each other dead and contested for a shared IP, resulting in a long-running partition.
- **Network Unreachability**: Further network problems included DNS resolution failures and nodes reporting network unreachability, though impact was minimal as the application was only a proxy.

Wide Area Networks (WAN): Background

- WANs face unique challenges in ensuring network reliability, especially for geographically widespread services.
- Graceful degradation under partitions or increased latency is crucial for high availability.

Image generated using Adobe Photoshop.

WAN: Links

- CENIC WAN Study by UCSD Researchers: Five years of data from the CENIC WAN analyzed, involving over 200 routers across California.
- PagerDuty's System Availability Amidst Connectivity Issues: An AWS peering point issue in Northern California led to increased latencies and quorum loss for a PagerDuty EC2 node.

WAN: CENIC WAN Study by UCSD Researchers

- **WAN Analysis**: Five years of data from the CENIC WAN analyzed, involving over 200 routers across California.
- **Network Partitions**: Over 500 isolating network partitions identified, disrupting host connectivity.
- **Partition Duration**: Average durations from 6 minutes for software failures to over 8.2 hours for hardware issues, with a 95th percentile reaching up to 3.7 days.

WAN: PagerDuty's Availability During EC2 Quorum Loss

- **Resilience Design**: PagerDuty's infrastructure replicated across two EC2 regions and a Linode data center for high availability.
- **Peering Point Degradation**: An AWS peering point issue in Northern California led to increased latencies and quorum loss for a PagerDuty EC2 node.
- **Impact of Correlated Failures**: Shared infrastructure vulnerabilities resulted in 18 minutes of service downtime, affecting API requests and message dispatching.

Global Routing Failures: Background

- Global routing failures can occur despite high levels of redundancy in Internet systems.
- Such failures can have a widespread impact on Internet services.

Image generated using Adobe Photoshop.

Global Routing Failures: Links

- CloudFlare's DDoS Firewall Rule Incident: Deployed new firewall rule against a DDoS attack across 23 data centers.
- Juniper Routing Bug Affecting Level 3 Communications: Juniper Networks' routers experienced a bug leading to Level 3 Communications backbone outages.
- Global BGP Outages and Misconfigurations: Past BGP misconfigurations led to significant outages, affecting major sites and even attempting to redirect all Internet traffic.

Global Routing Failures: CloudFlare's DDoS Firewall Rule Incident

- **Network Defense**: Deployed new firewall rule against a DDoS attack across 23 data centers.
- **Unexpected RAM Consumption**: Rule propagated by FlowSpec protocol caused routers to crash due to RAM overload.
- **Recovery Complications**: Automatic reboots failed and management ports were inaccessible, causing prolonged outages and manual reboots.

Global Routing Failures: Juniper Bug Becomes Backbone Outages

- **Firmware Upgrade Bug**: Juniper Networks' routers experienced a bug leading to Level 3 Communications backbone outages.
- **Service Impact**: Major services like Time Warner Cable, RIM BlackBerry, and UK ISPs went offline due to the bug.

Global Routing Failures: Global BGP Outages and Misconfigurations

- **Pakistan Telecom YouTube Hijack**: Incorrect BGP advertisement led to global traffic hijack and YouTube becoming unreachable in 2008.
- **Duke University BGP Experiment**: An experimental BGP flag tested by researchers caused a global internet outage in 2010.
- **Notable Misconfigurations**: Past BGP misconfigurations led to significant outages:
  - 1997: AS7007 caused a global outage.
  - 2005: Turkey attempted a redirect for the entire Internet.
  - 2006: sites including *Martha Stewart Living* & *The New York Times* knocked offline.
  - 2014: Turkey hijacks BGP and impersonates Google's public DNS servers.

NICs and Drivers: Background

- NICs and drivers play a crucial role in network reliability, but hardware and driver issues can lead to network partitions.
- Firmware bugs and driver issues can cause network failures and service outages.

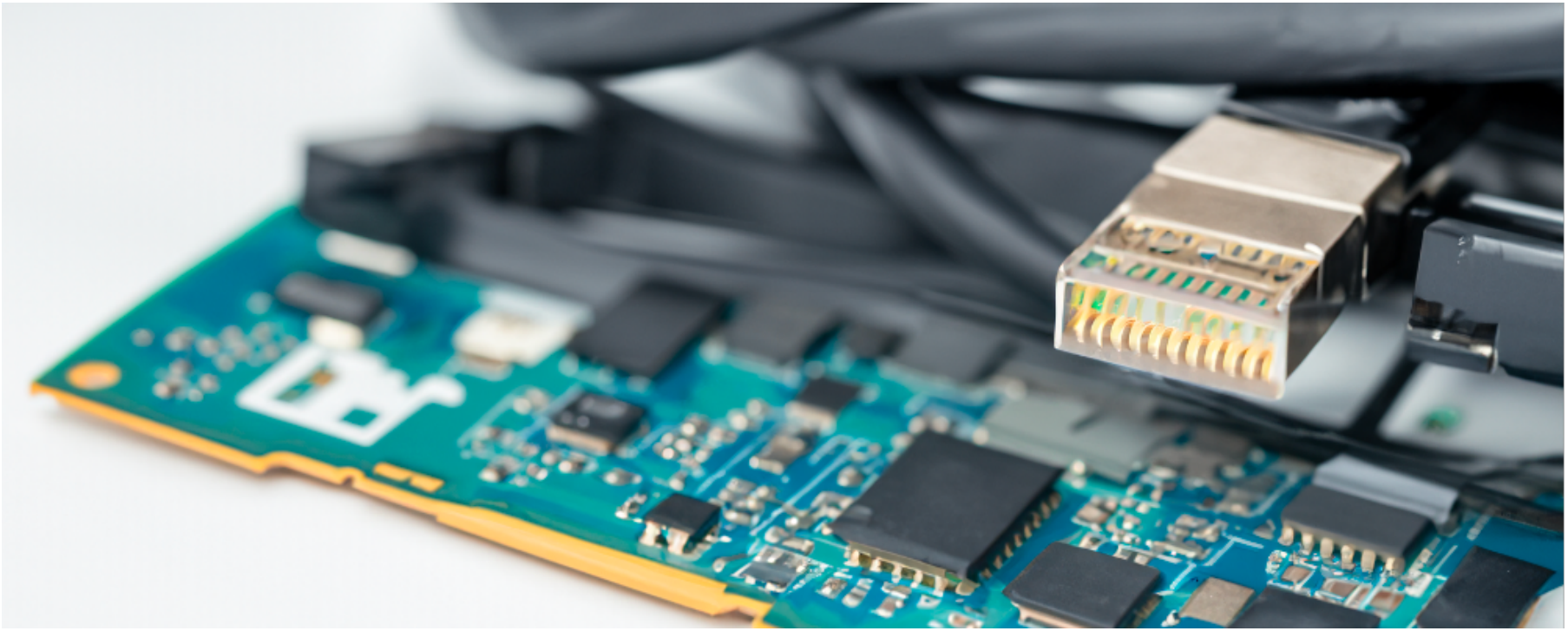


Image generated using Adobe Photoshop.

NICs and Drivers: Links

- BCM5709 Chip Issues and Impact: Broadcom BCM5709 chip failed to process inbound packets while still sending heartbeats, causing service unavailability for five hours.
- Intel 82574 "Packet of Death": Incorrect flashing of the EEPROM on Intel 82574-based systems caused a hard-to-diagnose error, disabling the NIC upon receiving a specific SIP packet.
- GlusterFS Partition Due to Driver Bug: CityCloud encountered network issues in GlusterFS pairs post-upgrade, initially suspected to be linked to link aggregation.

NICs and Drivers: BCM5709 Chip Issues and Impact

- **Inbound Packet Drop**: Broadcom BCM5709 chip failed to process inbound packets while still sending heartbeats, causing service unavailability for five hours.
- **Widespread Firmware Bugs**: The BCM5709's bugs, particularly in flow control, led to cascading failures across servers and network switches.
- **Driver Challenges**: The bnx2 driver and Broadcom 57711 chipset were associated with network instability and high latencies, respectively.

NICs and Drivers: Intel 82574 "Packet of Death"

- **EEPROM Flashing Failure**: Incorrect flashing of the EEPROM on Intel 82574-based systems caused a hard-to-diagnose error, disabling the NIC upon receiving a specific SIP packet.
- **Cold Restart Required**: Only a complete system restart could restore normal functionality.

NICs and Drivers: GlusterFS Partition Due to Driver Bug

- **Unexpected Network Failures**: CityCloud encountered network issues in GlusterFS pairs post-upgrade, initially suspected to be linked to link aggregation.
- **Driver Issue Identification**: The problem was traced back to a driver bug, which, once resolved, restored service but left data inconsistencies and corruption.

Application-Level Failures: Background

- Application-level issues can result in communication failures and network partitions.
- Designs that expect synchronous communication may behave unexpectedly during periods of asynchrony.

Image generated using Adobe Photoshop.

Application-Level Failures: Links

- Excessive CPU and Memory Use at Bonsai.io (n/a): High CPU/memory demand on an ElasticSearch node led to connectivity issues and request bottlenecks.
- Long GC Pauses and I/O in ElasticSearch: Long garbage collection pauses caused ElasticSearch nodes to lose connection, triggering unnecessary leader elections.
- MySQL Overload and Pacemaker Segfault on GitHub: Migration-induced overload resulted in the primary MySQL node's demotion and promotion of a bad secondary.
- VoltDB Regular Network Failures on EC2: Frequent network issues caused replica divergence, with both nodes operating as primaries and leading to data loss.
- Mystery RabbitMQ Partitions: Partitions with no clear connectivity loss, with increased timeout settings only reducing the frequency, not preventing them.
- ElasticSearch Discovery Failure on EC2: Two-node ElasticSearch clusters on EC2 failed to form correctly resulting in split-brain issues.
- RabbitMQ and ElasticSearch on Windows Azure: Periodic network partitions on Azure affected RabbitMQ and ElasticSearch.

Application-Level Failures: Excessive CPU and Memory Use at Bonsai.io

- **Cluster Overload**: High CPU and memory demand on an ElasticSearch node led to connectivity issues and expensive request bottlenecks.
- **Split-Brain Consequences**: Server restarts caused the cluster to divide, impeding index operations and leading to service outage and delays.

Application-Level Failures: Long GC Pauses and I/O in ElasticSearch

- **GC Induced Failures**: Long garbage collection pauses caused ElasticSearch nodes to lose connection, triggering unnecessary leader elections.
- **Quorum Issues**: Faulty quorum configurations resulted in the election of multiple primaries, causing data inconsistencies and system downtime.

Application-Level Failures: MySQL Overload and Pacemaker Segfault on GitHub

- **MySQL Migration Load**: A migration-induced load led to the primary MySQL node being erroneously demoted and the promotion of a less efficient secondary.
- **Coordinator Crash**: Following a configuration conflict and a segfault in the replication coordinator, GitHub faced a significant service disruption.

Application-Level Failures: VoltDB Regular Network Failures on EC2

- **Replica Divergence**: Frequent network issues caused replica divergence, with both nodes operating as primaries and leading to data loss.

Application-Level Failures: Mystery RabbitMQ Partitions

- **Unexplained Partitions**: RabbitMQ experienced partitions with no clear connectivity loss, with increased timeout settings only reducing the frequency, not preventing them.

Application-Level Failures: ElasticSearch Discovery Failure on EC2

- **Cluster Formation Delays**: Two-node ElasticSearch clusters on EC2 failed to form correctly due to extended discovery message exchanges, resulting in split-brain issues.

Application-Level Failures: RabbitMQ and ElasticSearch on Windows Azure

- **Azure Network Stability**: Reports of periodic network partitions on Azure affected RabbitMQ and ElasticSearch, although limited data is available due to Azure's newer market presence.

Where Do We Go From Here?

- A more open discussion on real-world network behavior is needed for robust distributed systems design.
- Sharing experiences and data can improve approaches to handling network partitions and failures.
- Balancing design for partition tolerance with the economic costs of highly reliable networks.
- Benefits of partition-aware designs in terms of latency and coordination advantages.

Conclusion

Thank you

Dr. Jacob Hochstetler
Distinguished Engineer, Vice President, Fidelity Investments
Clinical Assistant Professor, University of North Texas
Jacob.Hochstetler@UNT.edu
Jacob.Hochstetler@Fidelity.com
https://github.com/jh125486