Reliable Software Systems

Week 6: Data Integrity

February 14, 2018
Alyssa Pittman
University of Washington Allen School

Motivating Example: Gitlab data loss

Gitlab had a single primary and single secondary database.

Due to increased load, replication lagged and started failing.

While responding, the oncall engineer accidentally deleted the live database.

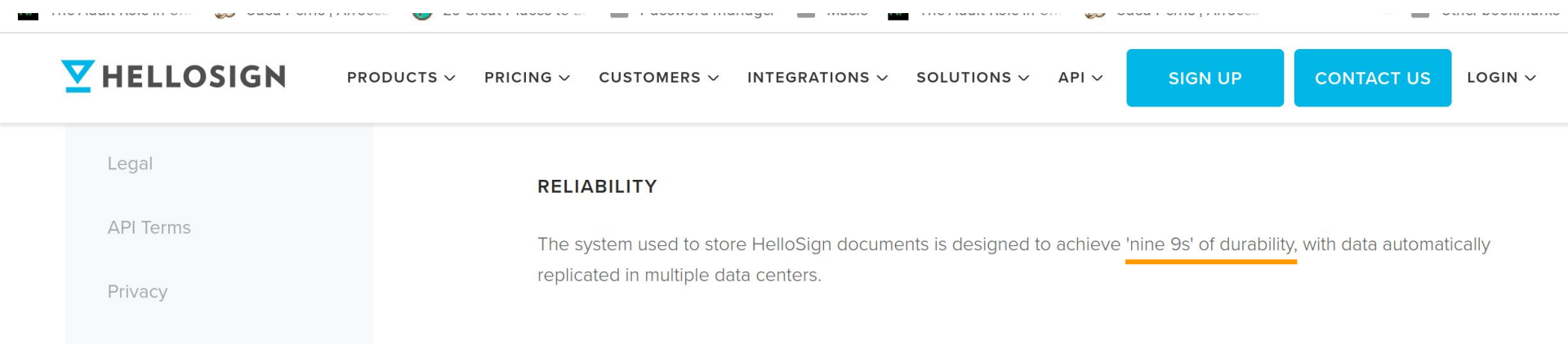
The nightly backups had been failing.

It took over 18 hours just to copy data from the latest snapshot.

Gitlab lost 6 hours of data (estimated at 5000 projects, 5000 comments, and roughly 700 users).

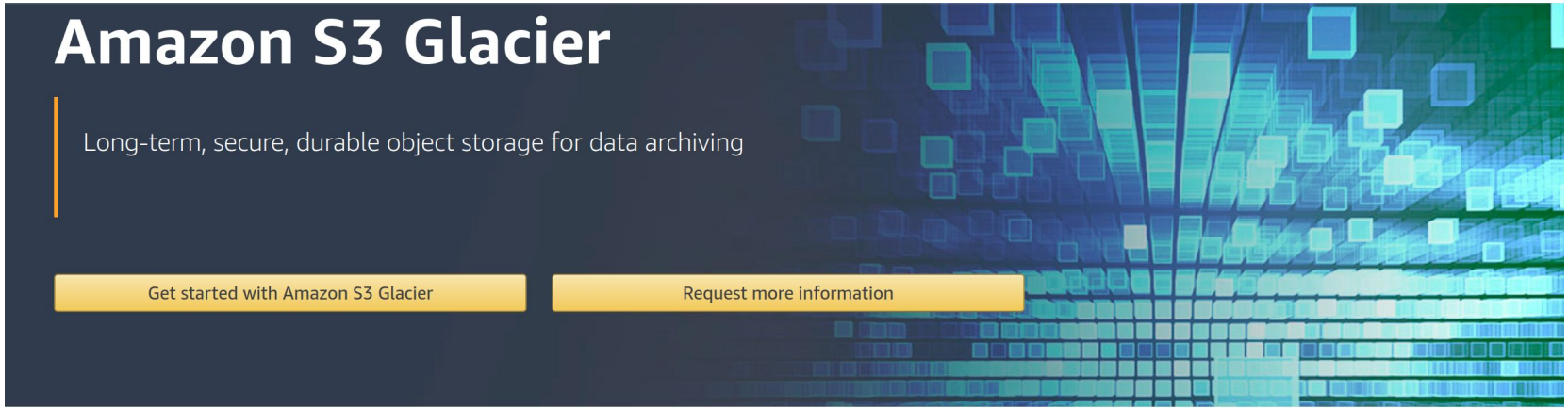
https://gitlab.com/gitlab-com/www-gitlab-com/blob/master/source/posts/2017-02-10-postmortem-of-database-outage-of-january-31.html.md

Example SLO: HelloSign

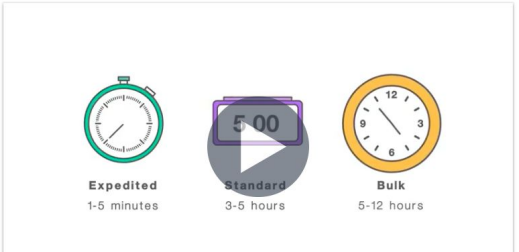


https://www.hellosign.com/legal/security

Example SLO: AWS Glacier



Amazon S3 Glacier is a secure, durable, and extremely low-cost cloud storage service for data archiving and long-term backup. It is designed to deliver 99.999999999% durability, and provides comprehensive security and compliance capabilities that can help meet even the most stringent regulatory requirements. Amazon S3 Glacier provides query-in-place functionality, allowing you to run powerful analytics directly on your archive data at rest. Customers can store data for as little as $0.004 per gigabyte per month, a significant savings compared to on-premises solutions. To keep costs low yet suitable for varying retrieval needs, Amazon S3 Glacier provides three options for access to archives, from a few minutes to several hours.



https://www.hellosign.com/legal/security

Service levels for durability

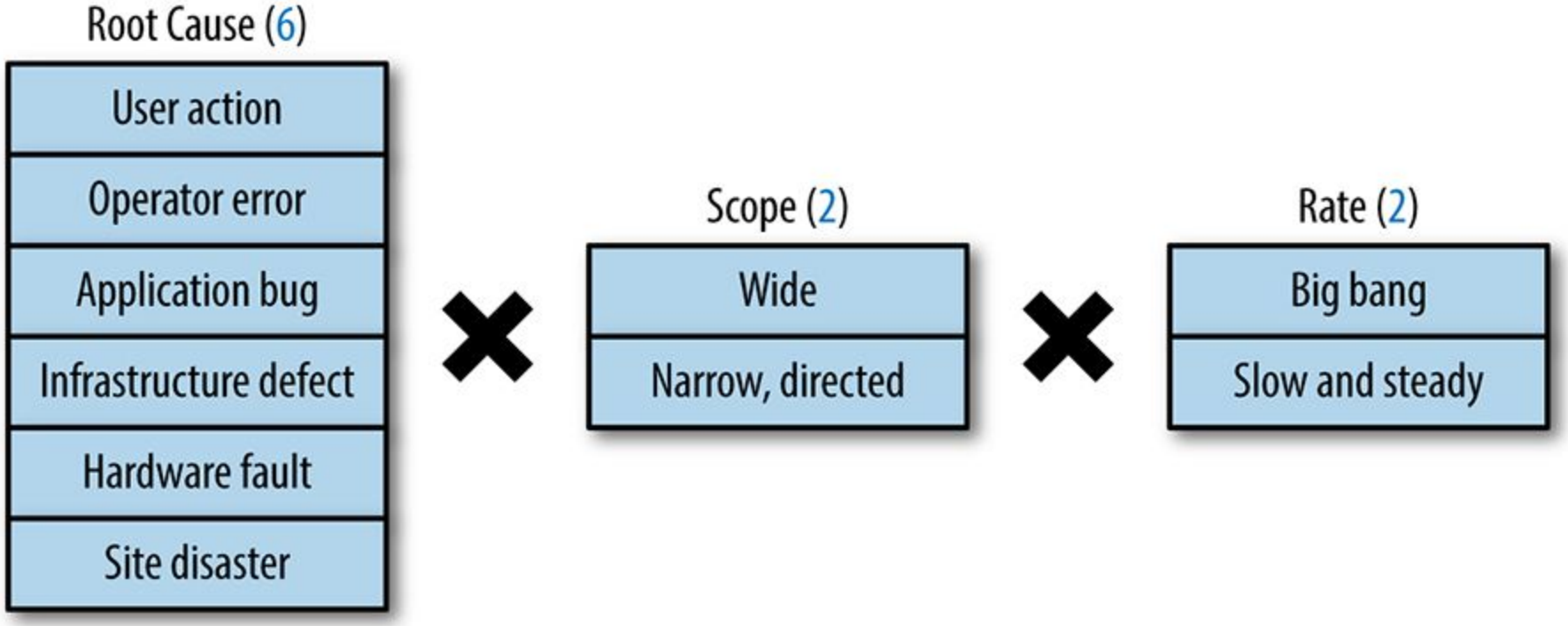
RPO: Recovery Point Objective

How much data can you lose during a failure?

RTO: Recovery Time Objective

How long does it take you to get data back online after a failure?

Types of failures



Replication

Stream changes to extra databases

Solves:

- Data locality
- Failover

But:

- Corruption is quickly replicated too.

Redundancy

Think of it as distributed RAID

9x3 data storage

All customer data is stored up to nine times across three geographically disparate locations, providing a superior level of data integrity and protection against data corruption.

From https://www.docusign.com/how-it-works/availability

Solves:

- Failover
- Restore if some copies are corrupted

But:

- Corruption/deletion can be propagated on write

Backups

Periodically backup all data, often in multiple locations/mediums

Solves:

- Recent deletions/corruptions

But:

- Stale backups lose data
- Can be expensive (space, computation)

Backups

Incremental backups: only save data that’s changed since the previous backup

On-demand backups: create a backup when you’re about to do a risky update

Transaction logs: log mutations that occurred since the last backup

Storage mediums: tape, off-site, ...

Backups are useless...

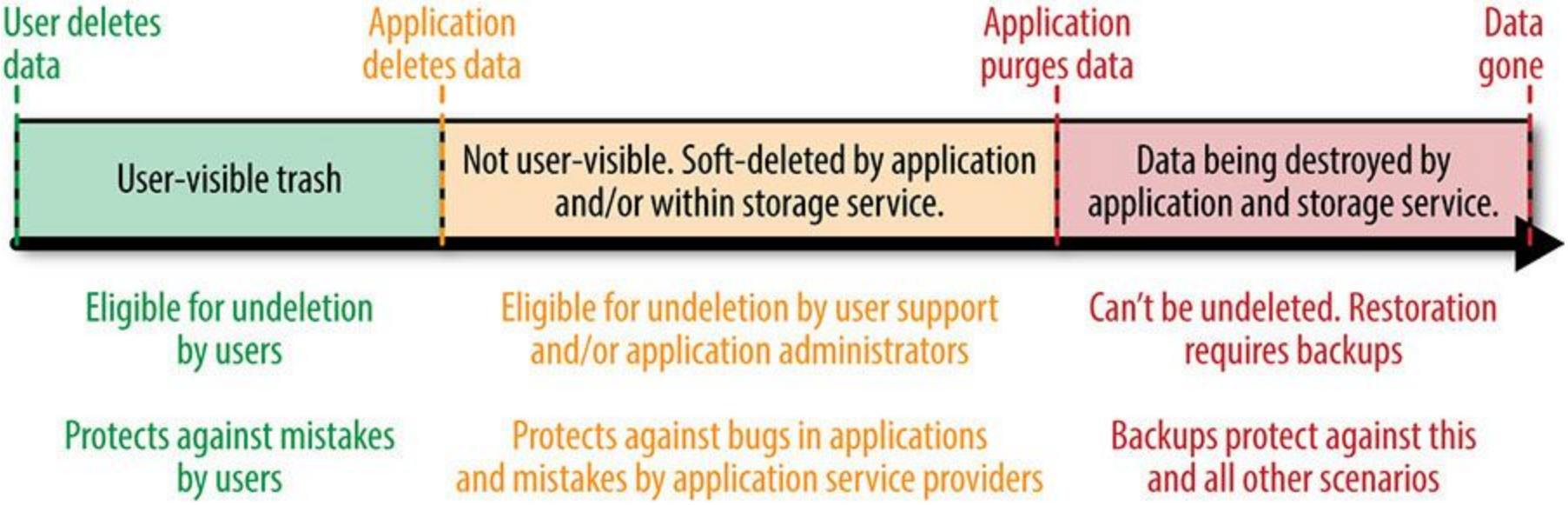
...What people care about is restores.

Monitor backups.

Test your restore process.

- Schema changes?
- Data migrations?
- Point-in-time restore?

Data retention



Data validation

Can you find (and solve) problems before users notice?

Automated validation pipelines

Checksums

Don’t underestimate the human element

Can you prevent people from easily running commands that destroy data?

…or not have those commands at all?

Can you limit access to the people who really need it?

Can you grant access to enough people that catastrophes (or vacations!) don’t cause problems?

Defense in depth

TL;DR:

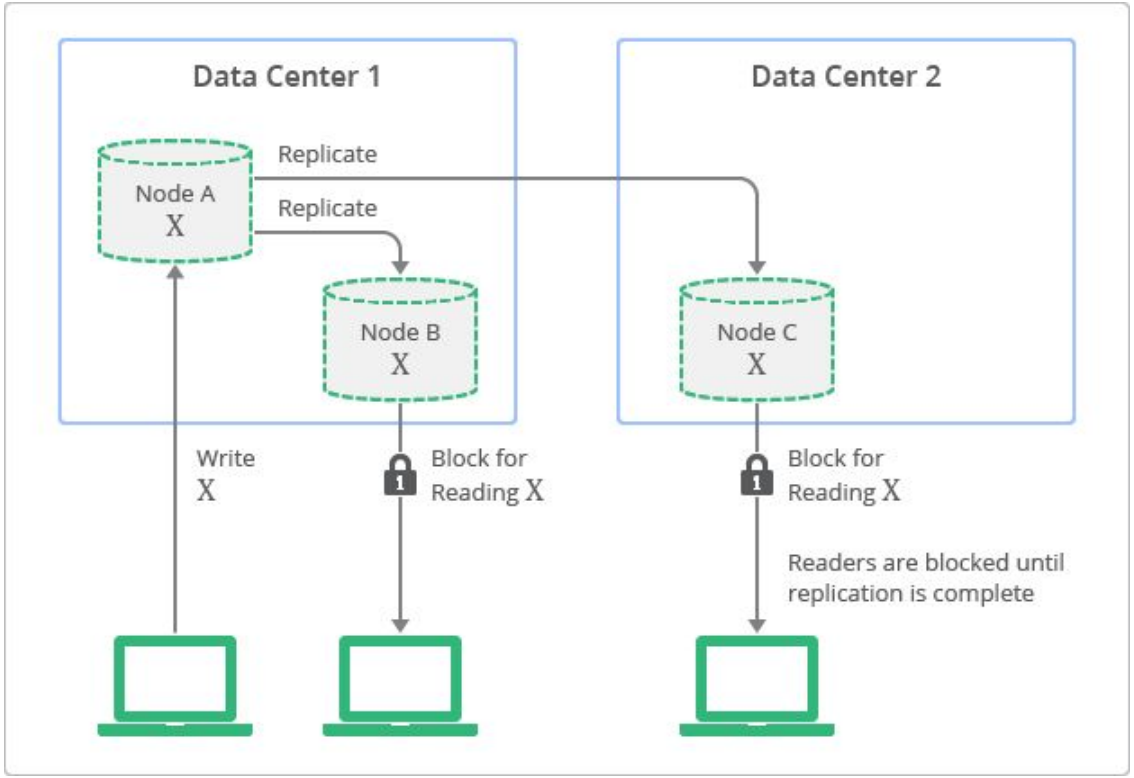
Have multiple approaches

And make sure they work!

Switching gears

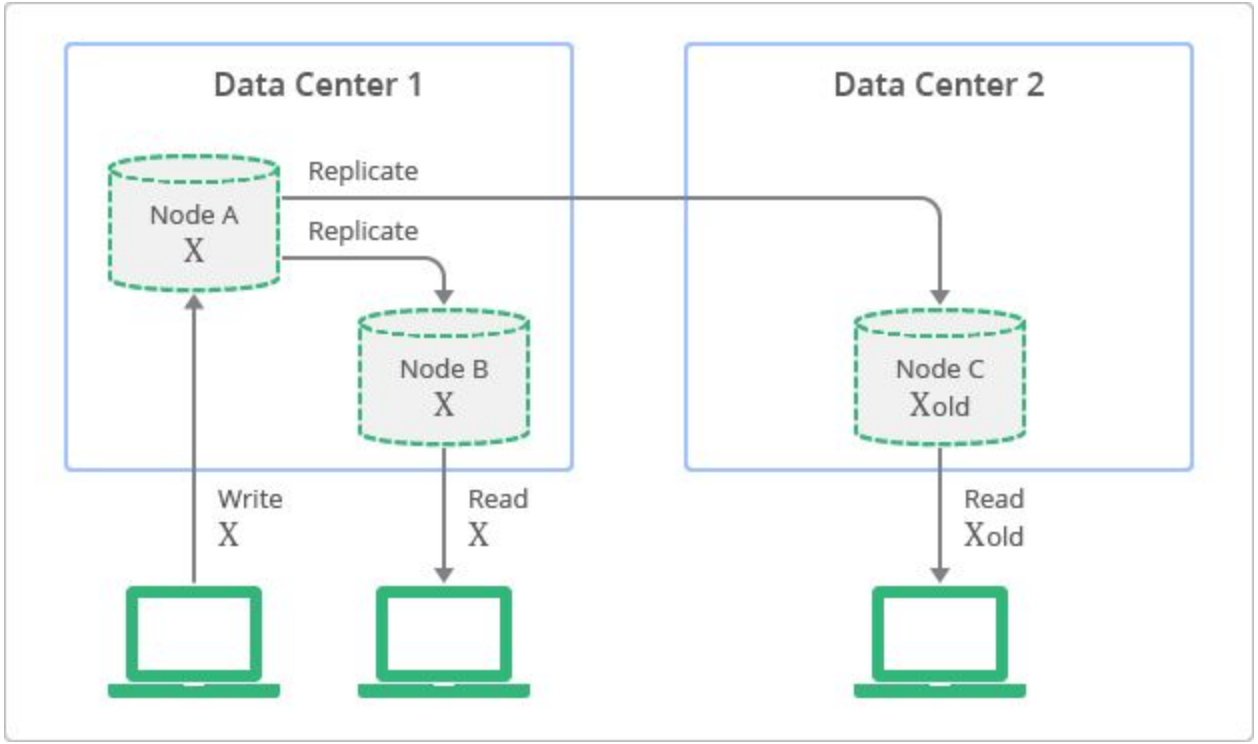
...from operational concerns to how we as developers expect our data to work

Strong consistency

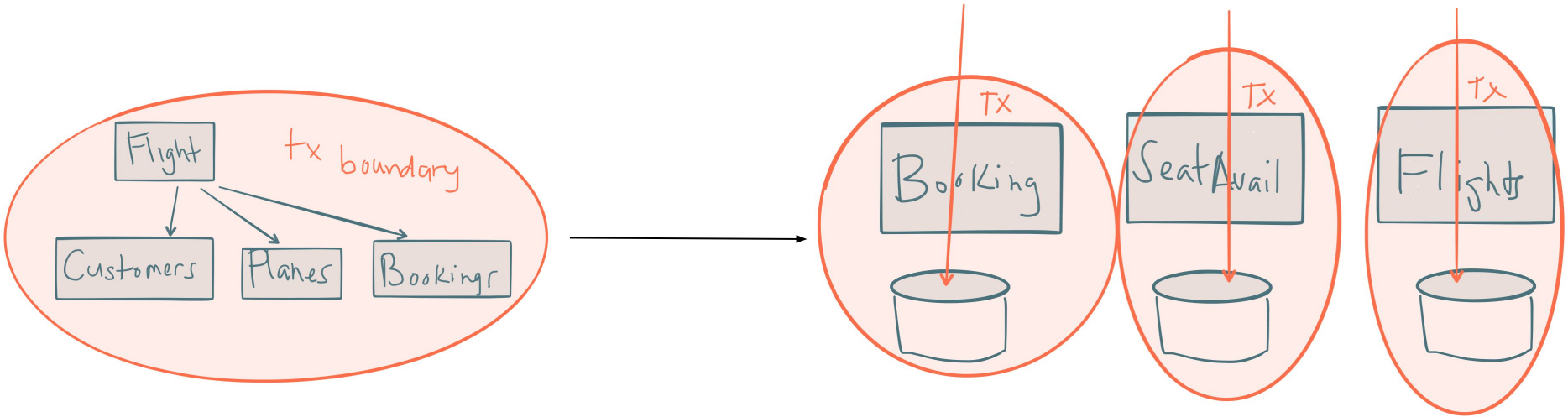


From Cloud Datastore documentation, copyright Google Inc , used under CC BY 3.0

Eventual consistency



Microservices and eventual consistency



https://blog.christianposta.com/microservices/the-hardest-part-about-microservices-data/

END

Next week - voting on topic for week 10

Think of any topics you want to suggest!