zalando

Типичные ошибки при построении высокодоступных кластеров и как их избежать

PGConf.Russia 2019,
Moscow

Alexander Kukushkin

06-02-2018

ABOUT ME

Alexander Kukushkin

Database Engineer @ZalandoTech

The Patroni guy

alexander.kukushkin@zalando.de

Twitter: @cyberdemn

WE BRING FASHION TO PEOPLE IN 17 COUNTRIES

17 markets

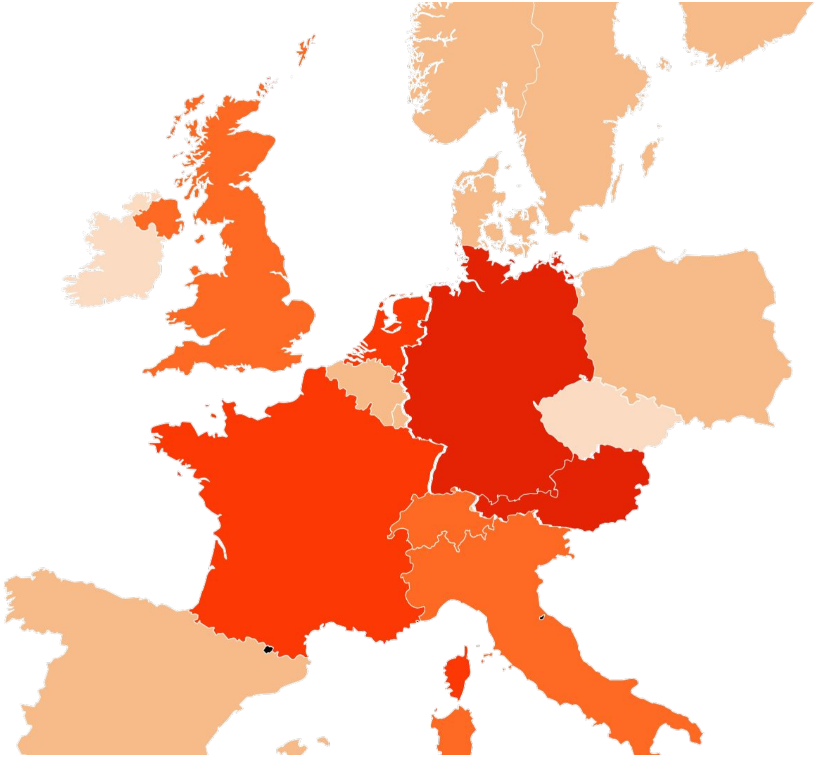
7 fulfillment centers

23 million active customers

4.5 billion € net sales 2017

200 million visits per month

15,000 employees in Europe



FACTS & FIGURES

> 300 databases

on premise

> 650 clusters

in the Cloud (AWS)

AGENDA

What is High Availability?

Disaster recovery

Automatic failover done right

Examples of real incidents

What HA will not solve?

Wrap it up

What is ~~High~~ Availability?

Availability

$$A = \frac{E[\text{uptime}]}{E[\text{uptime}] + E[\text{downtime}]}$$

Causes of Downtime

- Scheduled downtime (often excluded from availability)
  - Hardware/BIOS/Firmware upgrade
  - Software update
- Unscheduled downtime
  - Datacenter failure (natural disasters, fire, power outage)
  - Network splits
  - Hardware failure (CPU, network card, disk controller, disk)
  - Software/Data corruption (Bugs in application/OS code)
  - User error (rm -fr $PGDATA, DROP/TRUNCATE table, UPDATE/DELETE without WHERE clause)

Availability	Downtime			
	Year	Month	Week	Day
99% (“Two nines”)	3.65 d	7.31 h	1.68 h	14.4 m
99.9% (“Three nines”)	8.77 h	43.83 m	10.08 m	1.44 m
99.95% (“Three and a half nines”)	4.38 h	21.92 m	5.04 m	43.2 s
99.99% (“Four nines”)	52.6 m	4.38 m	1.01 m	8.64 s
99.999% (“Five nines”)	5.26 m	26.3 s	6.05 s	864 ms
99.9999% (“Six nines”)	31.56 s	2.63 s	604.8 ms	86.4 ms
99.99999% (“Seven nines”)	3.16 s	262.98 ms	60.48 ms	864 µs

What is HA anyway?

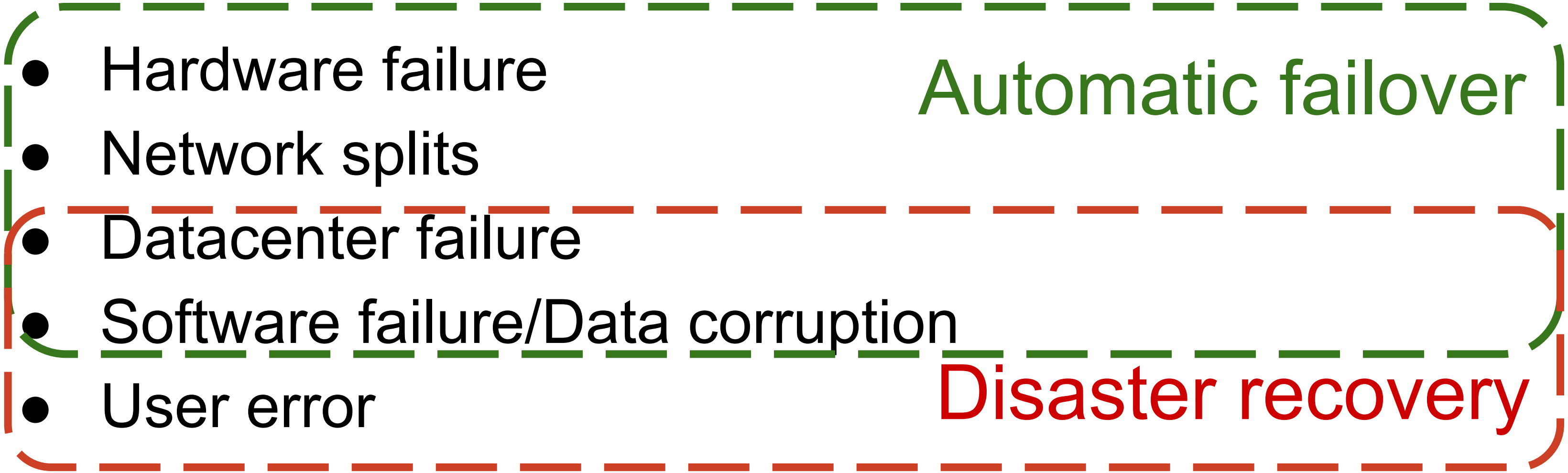
- No Official Definition appears to exist!

- Wikipedia:
  - **High availability** (HA) is a characteristic of a system, which aims to ensure an **agreed** level of operational performance, usually **uptime**, for a higher than normal period.

SLA, SLI, and SLO

- A **Service-Level Agreement** (**SLA**) is an agreement between a service provider and a client.
  - Type of service to be provided
  - Desired performance level (especially availability, reliability and responsiveness)
  - Monitoring process and service level **reporting**
  - Steps for reporting issues
  - Response and issue resolution time-frame
- A **Service-Level Indicator (SLI)** is a measure of the service level provided by a service provider to a customer
  - **Availability**
  - Latency
  - Throughput
- A **Service-Level Objective (SLO)** is a key element of **SLA;** a goal that service provider wants to reach

Causes of Unscheduled Downtime



- Hardware failure
- Network splits
- Datacenter failure
- Software failure/Data corruption
- User error

Automatic failover

Disaster recovery

What is High Availability?

Disaster recovery

Automatic failover done right

Examples of real incidents

What HA will not solve?

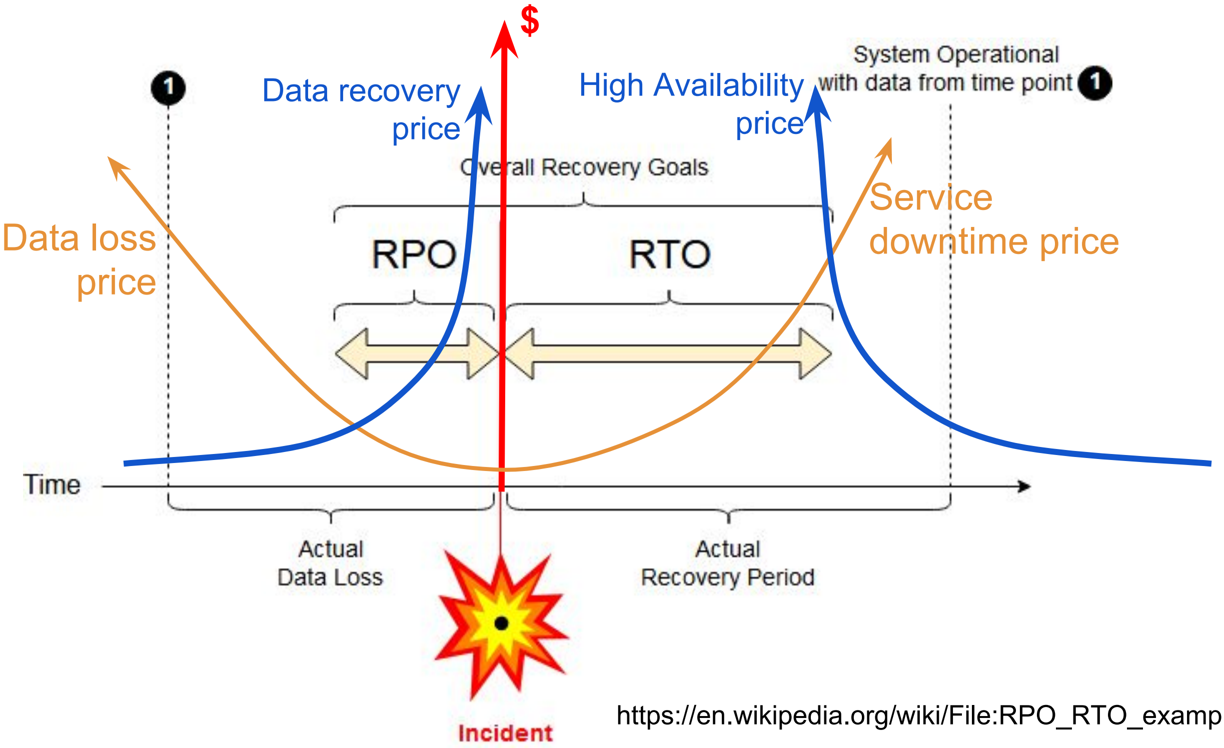
Wrap it up

Disaster recovery

- Involves a set of policies, tools and procedures to enable the recovery or continuation of vital technology infrastructure and systems following a natural or human-induced disaster

- Recovery point objective (**RPO**) and recovery time objective (**RTO**) are two important measurements in disaster recovery and downtime
    - A **recovery point objective (RPO)** is defined by business continuity planning. It is the maximum targeted period in which data (transactions) might be lost from an IT service due to a major incident
    - The **recovery time objective (RTO)** is the targeted duration of time and a service level within which a business process must be restored after a disaster (or disruption) in order to avoid unacceptable consequences associated with a break in business continuity

Disaster recovery



https://en.wikipedia.org/wiki/File:RPO_RTO_example_converted.png

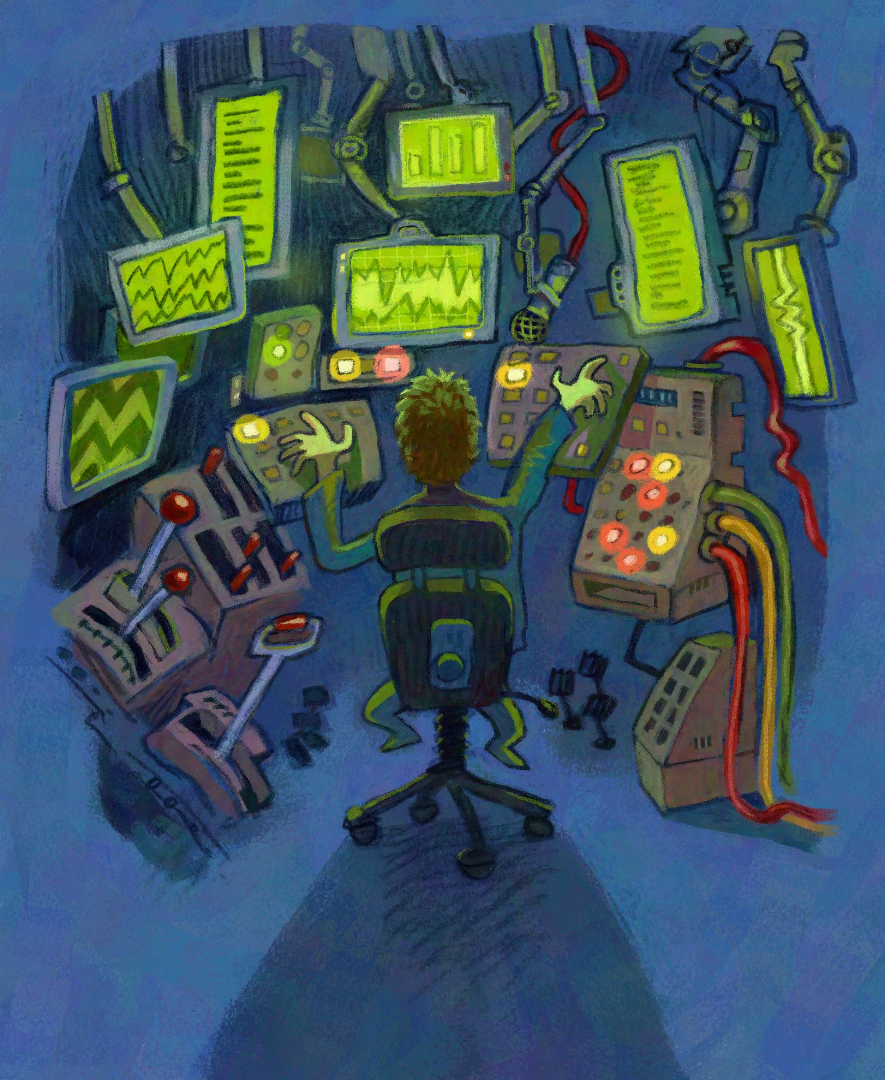
RPO, RTO & PostgreSQL

- Automatic failover won’t help to backup and restore data
  - Enable backups and log archiving
    - archive_timeout - how often postgres should archive WALs
    - pg_receivewal
  - Recovery from the backup might take hours
    - Consider having a delayed replica (recovery_min_apply_delay)
- if RTO is higher than 15 minutes, you don’t need automatic failover!
  - Unless you are running hundreds of clusters
- synchronous replication - to prevent data loss during failover

Sub-second Automatic Failover

- In general it is possible, but VERY expensive
- This is a price for complexity of such system
    - Complexity is often decreasing availability
    - The more elements a system has, the more reliable each element has to be
- Trade-off between the speed of failure detection and false positives

High Availability and Disaster Recovery Need Each Other!



What is High Availability?

Disaster recovery

Automatic failover done right

Examples of real incidents

What HA will not solve?

Wrap it up

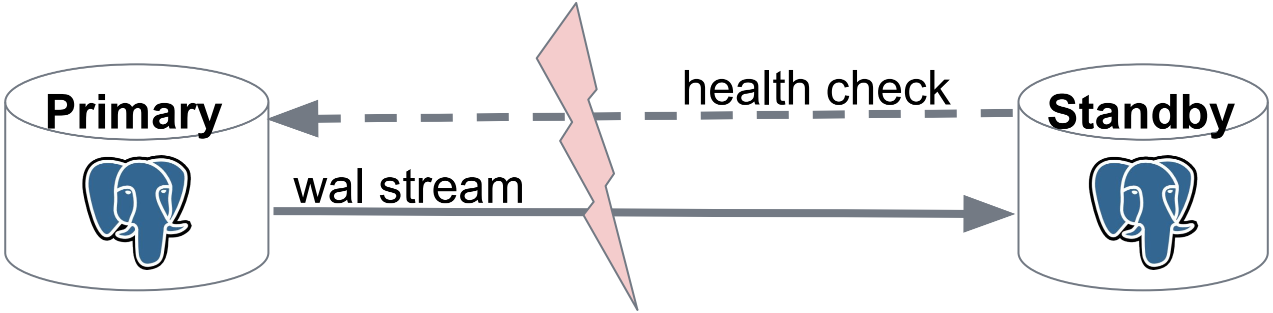
Multimaster?

- PostgreSQL XC/XL
  - Data nodes + Coordinators + 2PC + GTM(SPOF)
- BDR
  - logical replication + conflict resolution
    - eventual consistency
- Postgres Pro Enterprise (proprietary)
  - logical replication + E3PC

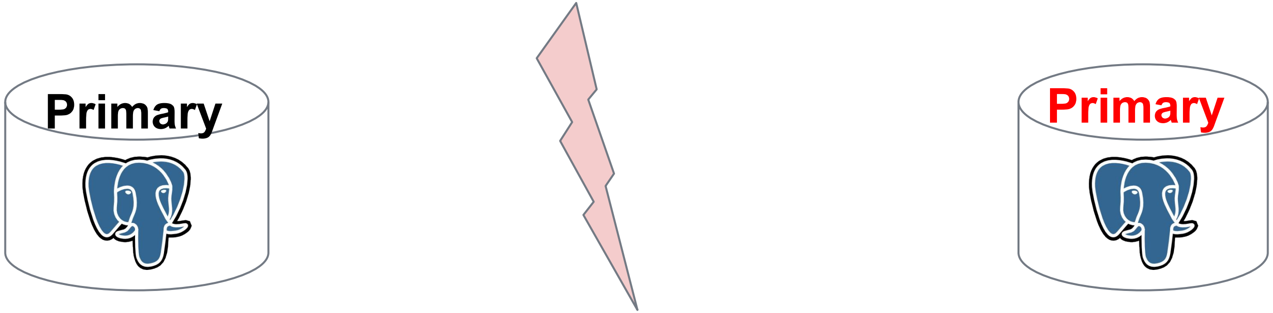
A good HA system

- Quorum
  - Helps to deal with network splits
  - Requires at least 3 nodes
- Fencing
  - Make sure the old primary is unaccessible. STONITH!
- Watchdog
  - Primary should not run if supervising HA process failed

No Quorum and no Fencing

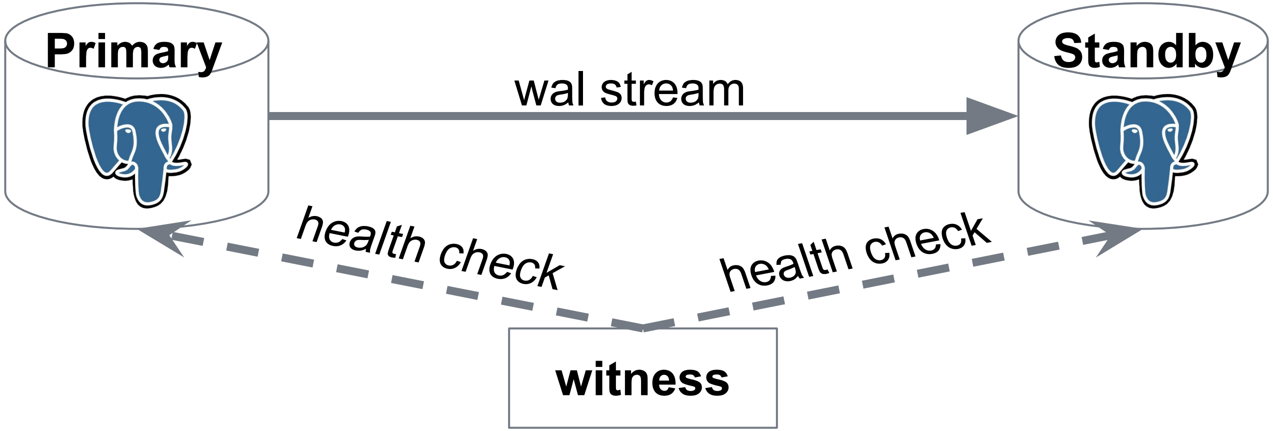


No Quorum and no Fencing

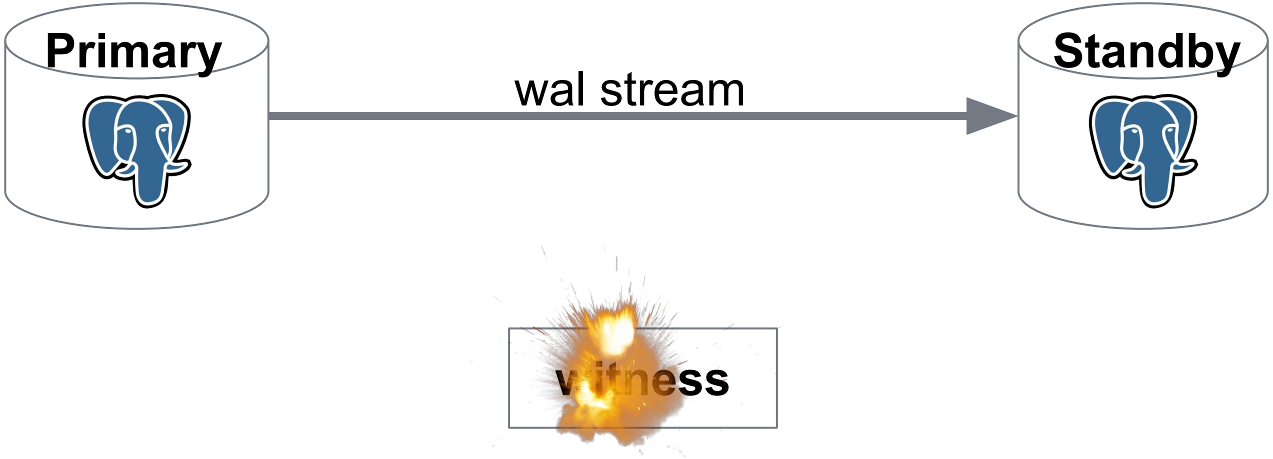


https://github.com/MasahikoSawada/pg_keeper

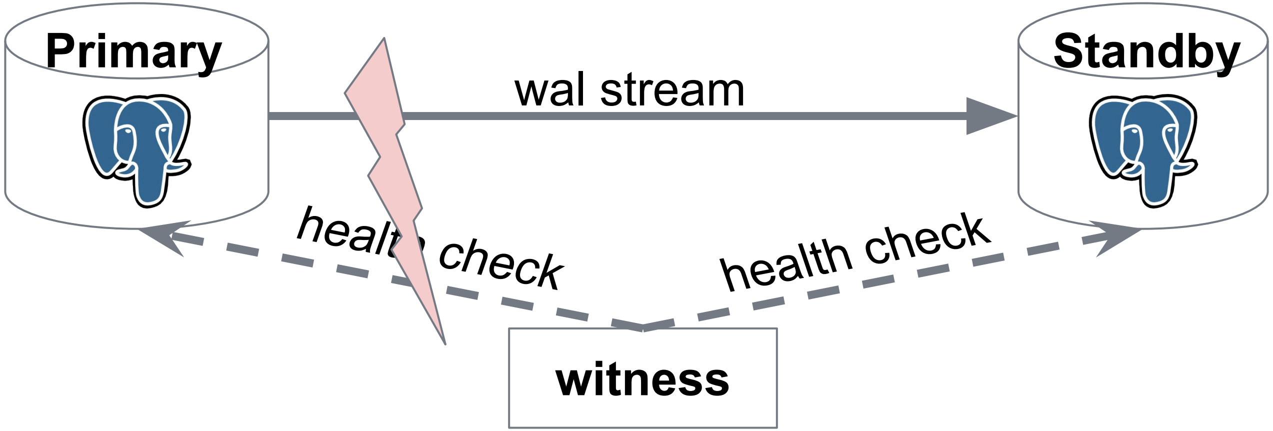
Witness node is making decisions



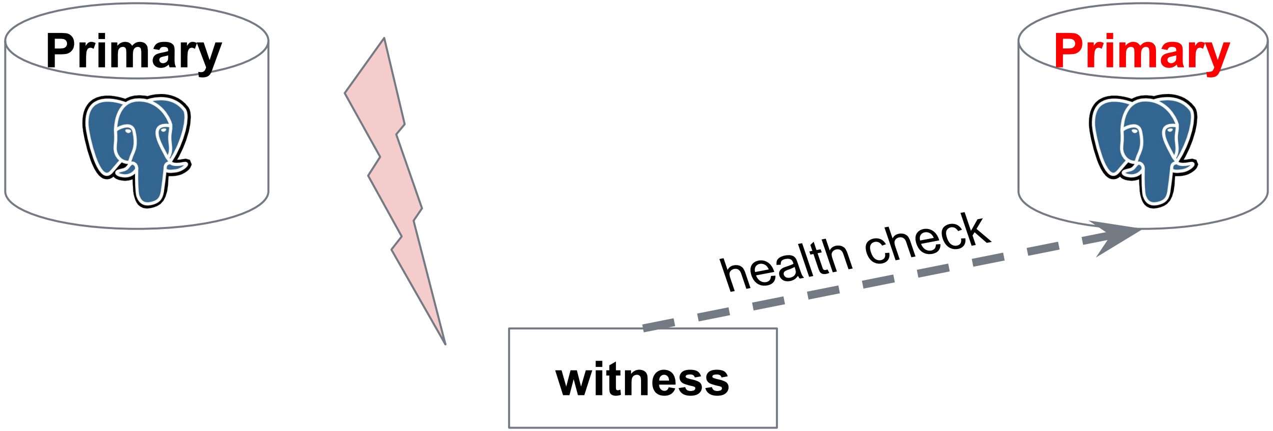
Witness node dies



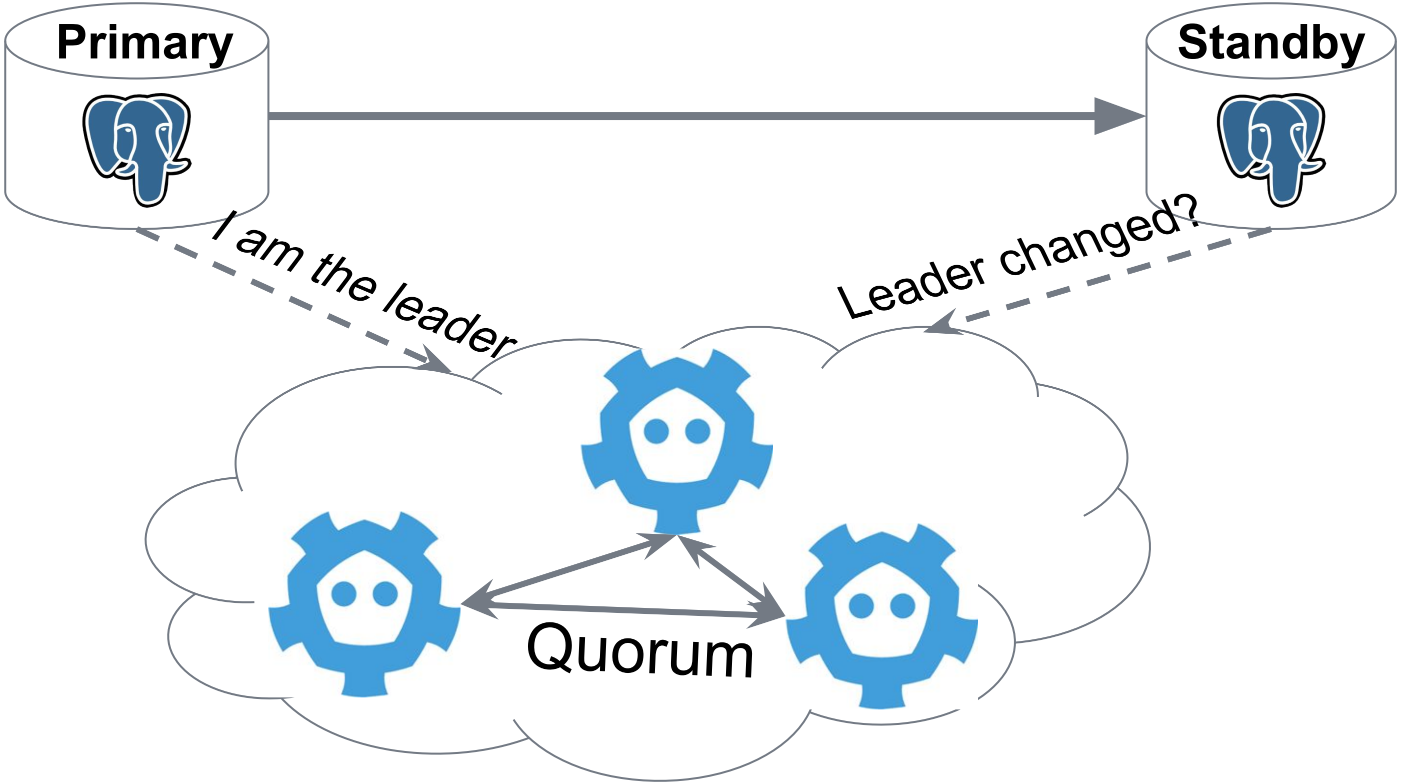
Witness and no Fencing



Witness and no Fencing



Automatic failover done right



What is High Availability?

Disaster recovery

Automatic failover done right

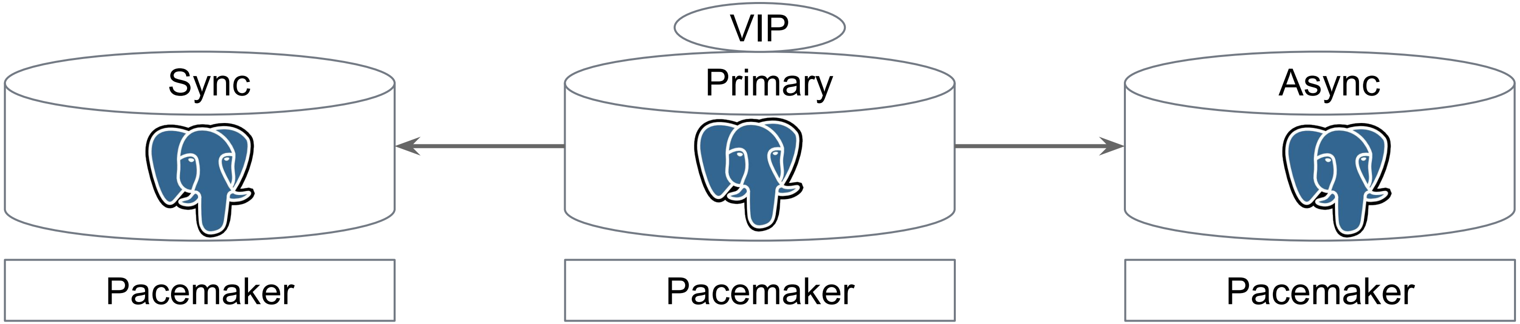
Examples of real incidents

What HA will not solve?

Wrap it up

Learn your HA system

GoCardless Incident



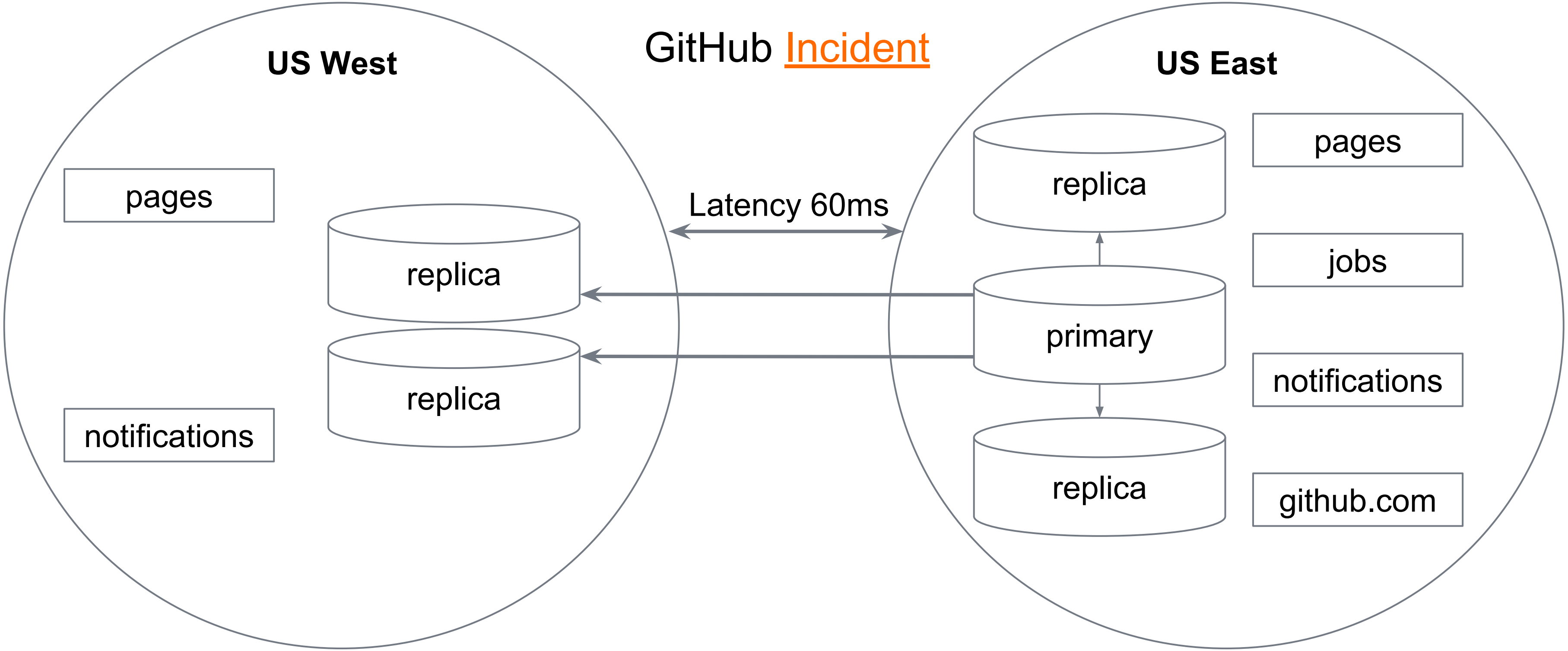
GoCardless Incident

- Failed raid controller on the primary
- Primary was manually terminating (hope on auto failover)
- Auto Failover didn’t work due to coincident crash of postgres on sync replica
- Spend 1h30m trying to trigger a failover using Pacemaker!
- Manually promoted sync replica
- Total outage 1h50m

GoCardless Lessons

- HA systems usually are quite complex
- Running them is similar to flying modern airplane
    - Mostly autopilot
    - But sometimes it fails
    - You need to know how to “fly” manually
- Learn your HA system
    - Try to break it and fix afterwards

Resource Planning



GitHub Incident

- Network split due to network maintenance
- Automatic failover from East to West Coast datacenter
- Applications from East are slow due to latency between East and West
- Switchback to East wasn’t possible due to a few seconds of writes which were not replicated
- Rebuild of all replicas in the East took nearly 16 hours
- Total time of incident 24h11m

GitHub Lessons

- Avoid doing cross-region failover if you don’t have 100% resources symmetry
- MySQL can’t do **pg_rewind** :)

Broken Disaster Recovery procedures

GitLab Incident

- Primary-Replica setup (no automatic failover)
- Increased database load on the primary resulted in **replica falling behind**
  - **WAL segment** needed for replica was **recycled** by primary
- A few attempts to rebuild replica with pg_basebackup (--checkpoint=spread)
- rm -fr $PGDATA on the primary! (**human error**)
- Three different backups were done only **once a day** (**no WAL archiving**)
  - pg_dump was **always failing** due to major version mismatch!
  - Azure disk snapshots were **disabled** for database servers!
  - LVM snapshots were **working** and periodically tested by restoring them to staging
    - Incident happened nearly 24 hours after the last snapshot was taken!
    - “Luckily”, someone manually created the snapshot 6 hours before the incident
- Recovery from LVM snapshot took longer than 18 hours

GitLab Lessons

- **RPO** and **RTO** were not set or not adequate to their business needs
  - Daily snapshots only and no WAL archiving (**RPO** = 24 hours)
  - Streaming replication can’t be used for Disaster Recovery
    - Unless it is a “delayed” replica
- Runbooks can’t replace fire-drills
- Backups must be monitored and tested

What is High Availability?

Disaster recovery

Automatic failover done right

Examples of real incidents

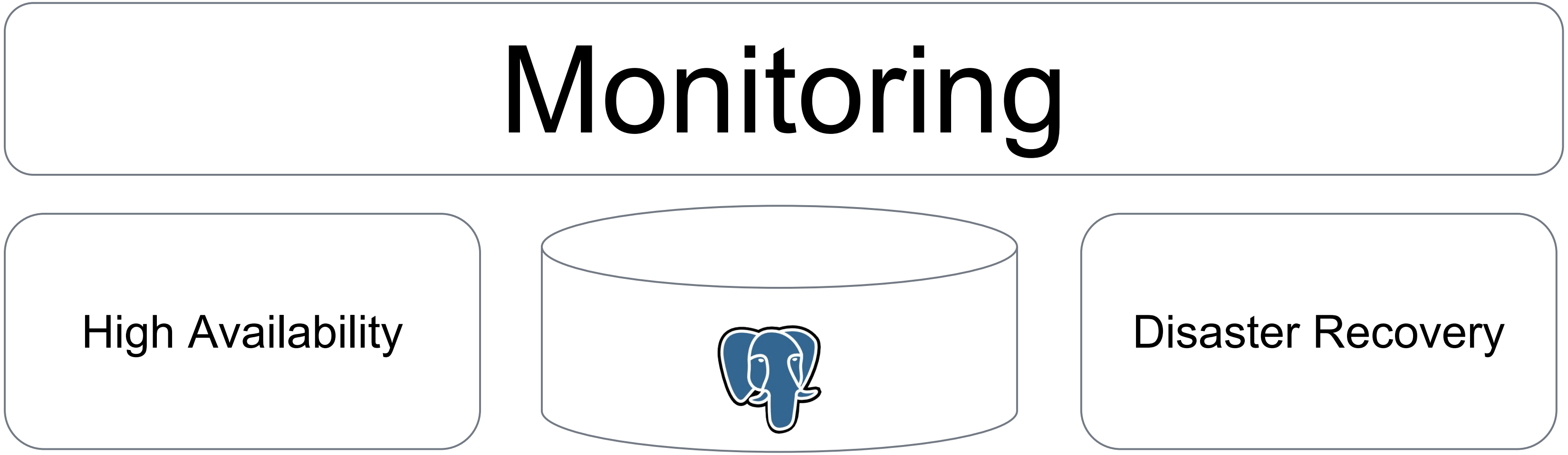
What HA will not solve?

Wrap it up

Monitoring

- HA doesn’t solve all problems with postgres, it won't cover:
  - Hardware errors, CPU load, Memory, etc...
  - Disk space for $PGDATA, tablespaces and pg_wal
  - autovacuums, checkpoints
  - Tables and indexes bloat
  - Queries performance
  - etc...
- Depending on **RPO** you maybe don’t need HA at all, but monitoring is a must
  - Don’t forget to monitor your HA system!

Everything must be monitored



Monitoring

High Availability

Disaster Recovery

Configuration tuning

- OS configuration tuning
  - Huge pages, shared memory, semaphores, overcommit, etc...
- PostgreSQL configuration tuning
  - shared_buffers, max_wal_size, checkpoint completion_target, random_page_cost, etc...
- HA won’t do it for you!

What is High Availability?

Disaster recovery

Automatic failover done right

Examples of real incidents

What HA will not solve?

Wrap it up

Wrap it up

- Always start with Disaster Recovery planning
  - Define **RPO** and **RTO**
    - Depending on **RTO** you maybe don’t need **HA**
  - Build the Availability you need, not the Availability you want
- Test everything
  - High Availability system
  - Backups!
- Do regular fire-drills

Thank you!

Questions?