PostgreSQL Data Recovery on a Damaged Filesystem

Chris Travers

Adjust GmbH

2018-10-12

Outline

About The Author

Chris Travers heads the Adjust database team, helps out with DBA tasks, and has contributed patches to PostgreSQL. He has been working with PostgreSQL since 1999.

About Adjust

Adjust provides mobile advertisement attribution and analytics on mobile application events. On average we track advertisement performance for 10 applications for each smartphone in use. We focus on fairness and transparency.

What we will cover

- Why we did data recovery
- Non-Technical Aspects of How we Approach Data Recovery
- Technical Aspects of How we Approach Data Recovery

The Disaster

- Catastrophic power event
- Massive file system damage
- Affected systems were all RAID on spinning disk
- Flash storage not affected and good
- Blocks appear to have been written to wrong locations

About the System Involved

- Two of our point of entry databases for analytics
- Not customer facing
- 1-2TB of new/hot data on SSD
- 6-10TB if older/colder data on RAID with spinning disk
- Catalogs on slower storage

Initial Symptoms

Shortly after the systems were restarted, we began to see errors:

Errors

ERROR: could not open file "base/142578/2605": No such file or directory

These errors were found on two different systems. The error indicates pg_cast does not exist.

fsck results

- Over a thousand orphaned files found
- pg_filedump proves to be only marginally useful for matching
- We eventually determine that the missing file is not found by fsck
- Flash storage where tablespace for recent tables stored is fine

Assessment of Backups

- Backups are nightly
- Could behind by up to 100M records on recent tables
- Recent tables on flash storage more behind than tables on damaged filesystem

Existing Backup Design

- Don't want to have 2PB of space allocated for backups
- Nightly
- Tables with lots of changes
- Business continuity impact of some data loss is not severe

Next backup solution

Immediately we started working on a new backup solution:

- Logical streaming backups
- Still per table
- Base plus deltas
- Much less space than binary backups
- No relational integrity guarantees (warehouse environment)
- `https://github.com/ikitiki/logical_backup`

Other Countermeasures We Could Have Used

- Enable page checksums
- Streaming backups of some kind

However in the case of these two systems, the damage was massive ans severe. Also many other things don't make sense in this case (replicas etc).

Impact of Taking These Systems Offline

- Impact is not customer-visible
- Reduces inbound capacity by a bit under 10%
- Systems already 25% over normal load with 2 systems down
- Major impact is it reduces our safety margin and ability to run maintenance
- For comparison this means 400k requests per second sustained with 2 servers down.

Determining Scope of Recovery

Goal is to recover data from intact file system to reduce data loss due to recovery from backup. The goal is to ensure that we have more accurate metrics if we ever have to backfill something.

General Assessment

- Not practical to work on copy (we break that rule)
- Would be really nice to have data recovered
- Data loss implications occur only for new metrics

Game Plan

- Take systems down
- Best effort at recovery
- Restore backups to new systems, along with recovered data
- Bring extra capacity online to cover down systems

The First Actual Recovery

5 Basic Ground Rules

1. Always Remember State of System is Uncertain
2. Deliberate before you Act
3. Understand Internals/Use the Source
4. Reason from Internals to Effects
5. If something is not expected, circle back and understand it before proceeding

General Strategy

1. Recover Catalogs Enough to Reindex Them
2. Dump tablespace of Recent Tables

Easy, right? Of course there are cases we could have to give up (like missing pg_class or similar)

Recovering Catalog Files

- Structure of base directory
- How Database Creation Happens in PostgreSQL
- Copying files from template0 when missing
- Checking what files were missing and impact of restoring to pristine state

PG Data Directory Structure

```
Chriss-MBP:test-10 christravers$ ls
PG_VERSION          pg_dynshmem         pg_multixact        pg_snapshots
pg_tblspc           postgresql.auto.conf base               pg_hba.conf
pg_notify           pg_stat             pg_twophase         global
pg_replslot         pg_stat_tmp         pg_wal              postmaster.opts
pg_commit_ts        pg_logical          pg_serial           pg_subtrans
pg_xact             postmaster.pid
```

Structure Of ‘base’ Directory

```
Chriss-MBP:test-10 christravers$ ls base
```

1 12557 12558 16390 24582 pgsql_tmp The entry ‘1’ is template1, and ‘12557’ is template0

Problem 1: Missing files in pg_xact

Reindexing would fail due to missing files in pg_xact (formerly pg_clog). In our workload rollbacks are rare.

- Structure of clog/pg_xact files
- Using hex editor to locate file with all commits
- Approach taken: copy known good file everywhere it is missing

After 3 days, success!

pg_xact directory structure

```
Chriss-MBP:test-10 christravers$ ls pg_xact
0000    0001    0002    0003    0004    0005
```

Files range from 0000 through FFFF. Each file is a series of codes for transaction states.

hexdump of file with multiple codes

hexdump 03C4

```
0000000 5555 5555 5555 5555 5555 5555 5555 5555
*
002d660 5555 5555 5555 5555 5555 5555 9555 5555
002d670 5555 5555 5555 5555 5555 5555 5555 5555
*
002dbe0 5555 5555 5555 5555 5555 5655 5555 5555
002dbf0 5555 5555 5555 5555 5555 5555 5555 5555
*
002e4d0 5555 5555 5555 5555 5555 5555 5555 5556
002e4e0 5555 5555 5555 5555 5555 5555 5555 5555
*
0040000
```

The Second Actual Recovery

State of Second Machine

- Corrupt postgresql.conf
- Missing pg_xact directory
- Missing template0 database
- Missing tablespace directories
- Significantly More Filesystem Damage

Approach Taken

- Create A full pg_xact with all possible files of all commits
- Copy from postgres db instead
- Re-initialize tablespaces directories manually
- Due to learning in previous one, this took only 3 hrs

Note near misses with tablespaces directories!

Data Recovery Is Not Just About Technique

- We are working on systems with known levels of breakage.
- Downtime can have costs and it is important to assess and reassess
- Work closely with stakeholders in terms of effects of downtime.
- Understand you might not succeed.
- **Do Not Panic**

Technical Takeaways

- Do not be afraid to read the source
- Make sure you know how things work
- The goal is to fake a working system enough to get data off.
- If in doubt, call experts.

Thank you for coming!