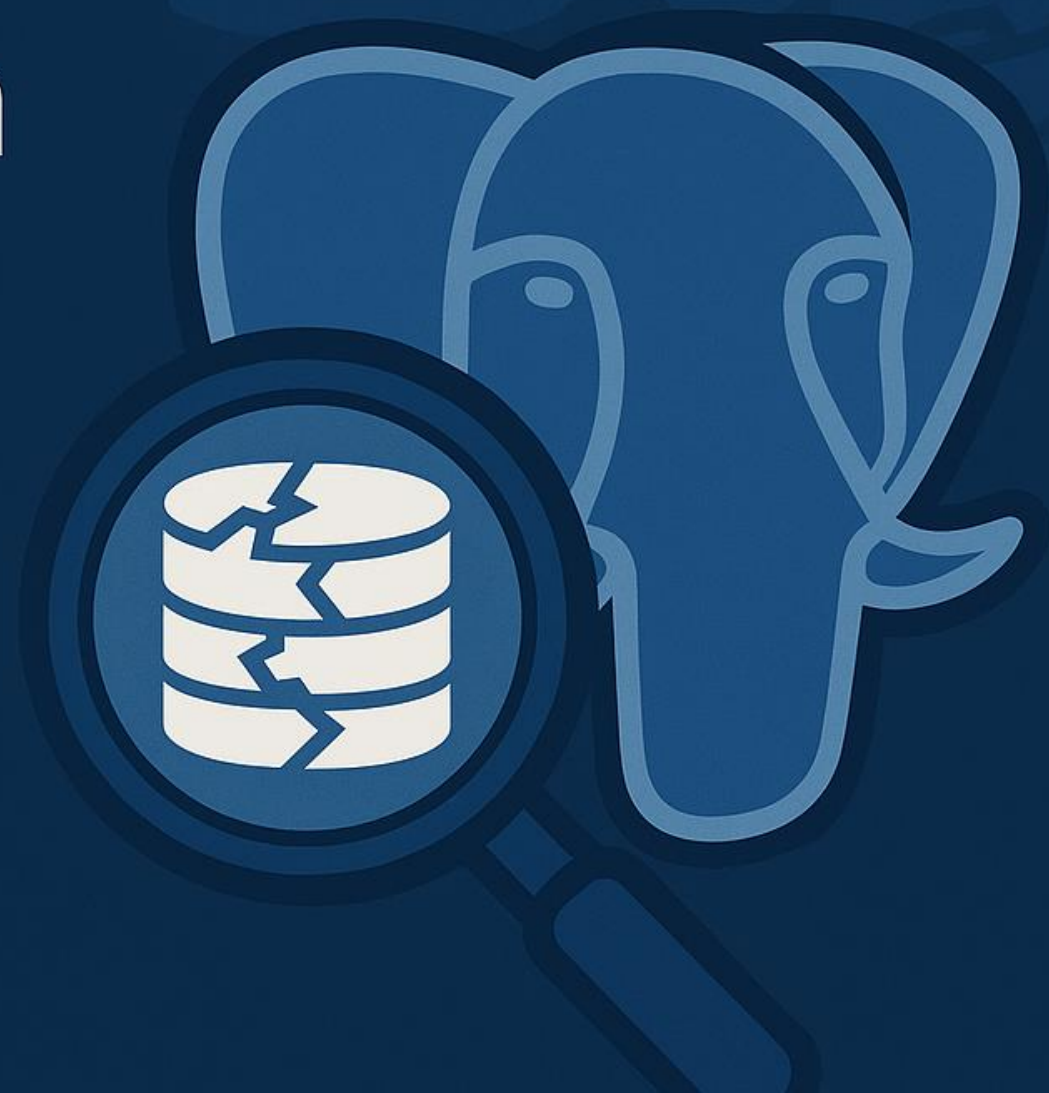


PGCONF.DEV
2025



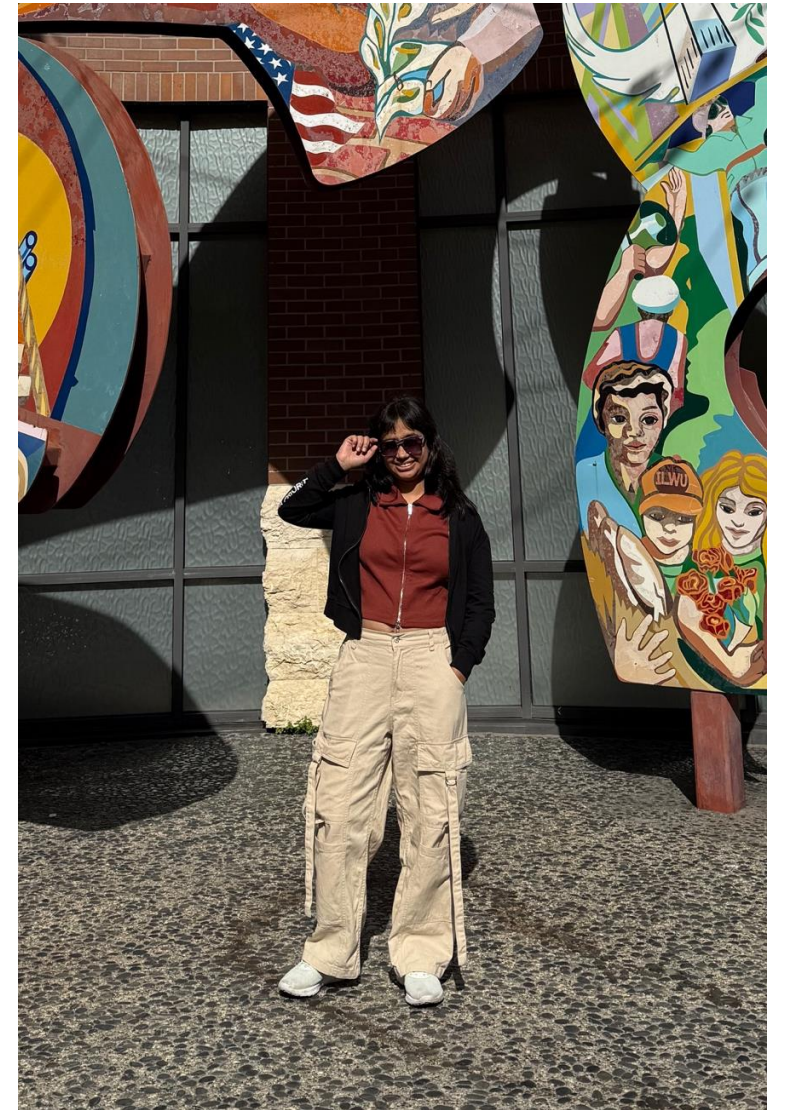
Debugging Data Corruption in PostgreSQL

A Systematic Approach



Introduction

- Palak
 - Software Engineer @ Microsoft
 - Member of the Azure DatabaseFor PostgreSQL team
 - Experience on onboarding new extensions to FSPG, handling core dumps and corruption issues and making core changes to FSPG.
 - Contributing to PG community.





Agenda

- What is Data Corruption
 - Causes of Corruption
 - Analysis of Few Corruption Cases
 - "Could not read block"
 - "Could not locate a valid checkpoint"
 - General Approach to Handling Corruption Cases
 - Best Practices
 - How to Detect Corruption
 - Corruption Recovery
-

What is Data Corruption?

- Data corruption is like amnesia for data
- It's like opening a spreadsheet and finding gibberish instead of numbers.
- Unintended changes in the database



Causes of Corruption



Bad Hardware

– For example, bad disk or bad memory.

Causes of Corruption



Bad Hardware

– For example, bad disk or bad memory.



Bad Software

– For example, bugs in PostgreSQL, kernel, filesystem, backup tool, etc.

Causes of Corruption



Bad Hardware

– For example, bad disk or bad memory.



Bad Software

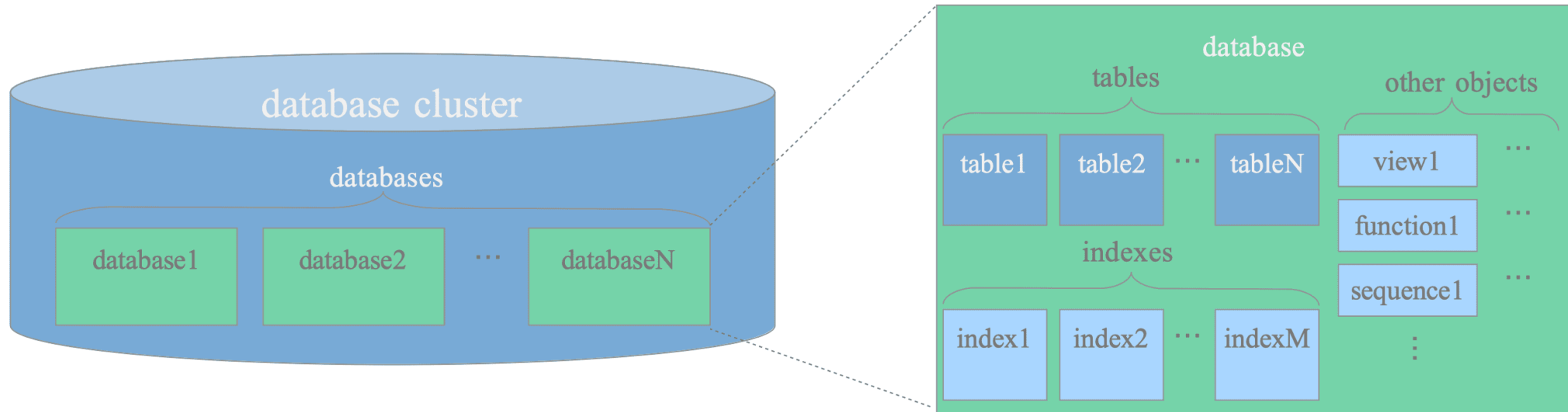
– For example, bugs in PostgreSQL, kernel, filesystem, backup tool, etc.



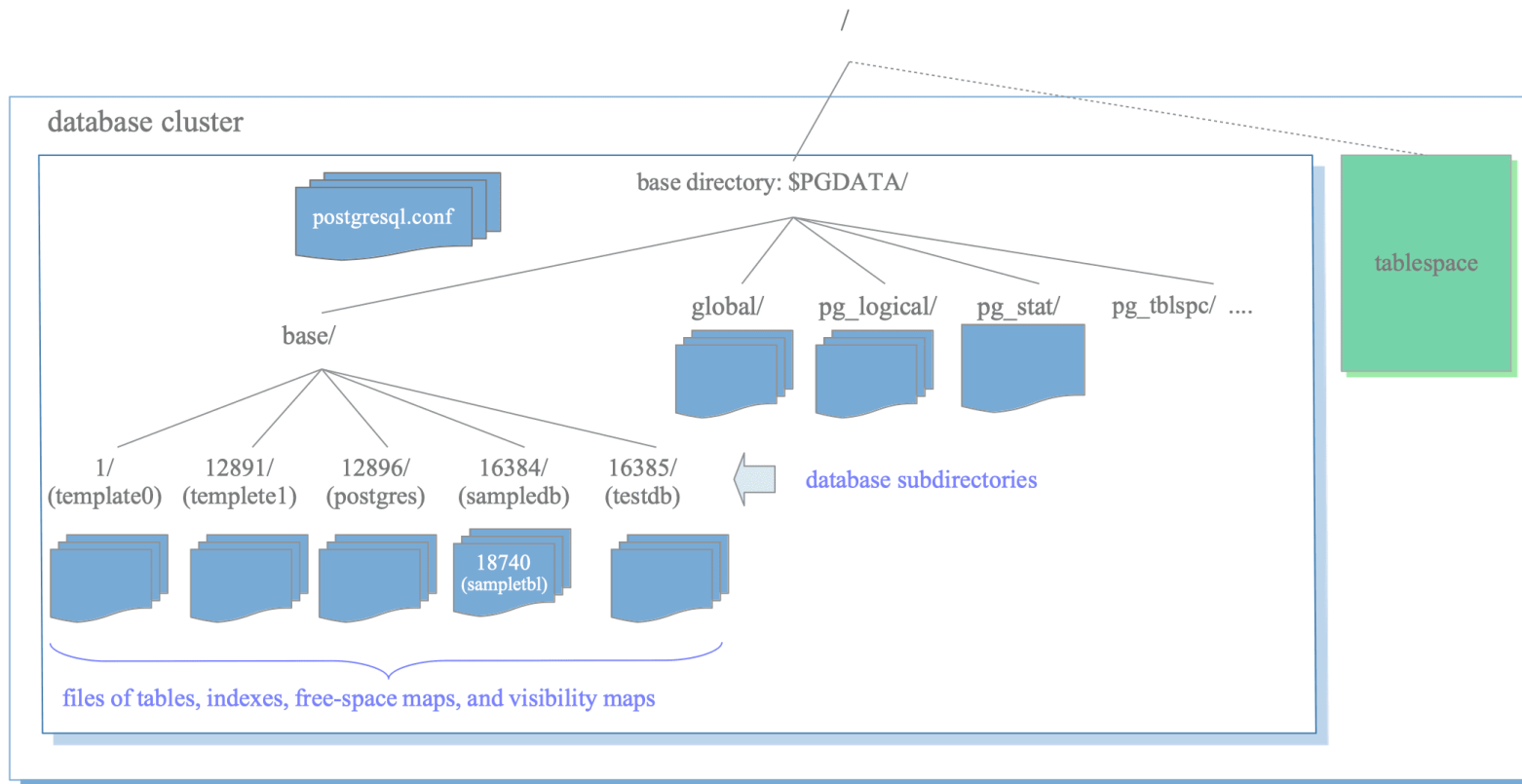
User Error

– For example, faulty backup and recovery procedures.

Logical layout of DB cluster



Physical Layout of DB cluster



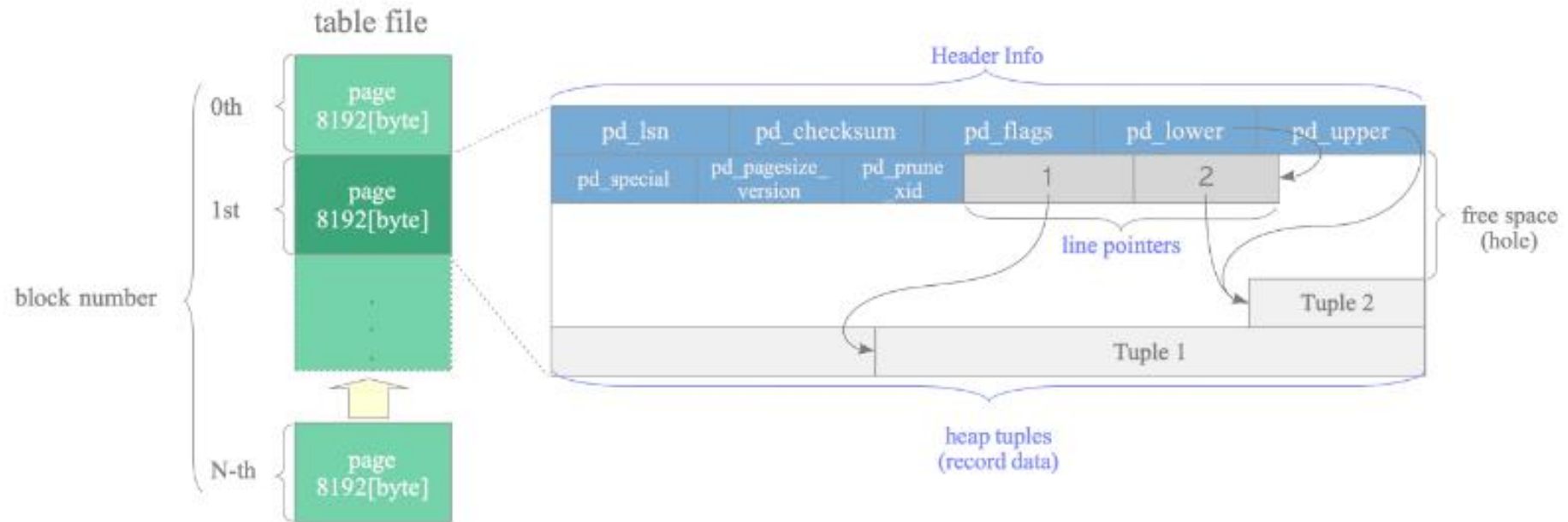
Physical Layout of DB cluster

```
~/orcasql-postgresql/test$ ls -lh global
total 596K
-rw----- 1 palak palak 8.0K Apr 30 19:40 1213
-rw----- 1 palak palak 24K Apr 30 19:40 1213_fsm
-rw----- 1 palak palak 8.0K Apr 30 19:40 1213_vm
-rw----- 1 palak palak 8.0K May  2 07:14 1214
-rw----- 1 palak palak 16K May  2 07:14 1232
-rw----- 1 palak palak 8.0K May  2 07:49 pg_control
-rw----- 1 palak palak 524 Apr 30 19:40 pg_filenode.map
-rw----- 1 palak palak 29K May  2 07:44 pg_internal.init
```

Physical Layout of DB cluster

```
~/orcasql-postgresql/test$ ls -lh base
total 40K
drwx----- 2 palak palak 4.0K May  2 07:44 1
drwx----- 2 palak palak 12K May  2 07:44 16427
drwx----- 2 palak palak 12K May  2 07:44 16498
drwx----- 2 palak palak 4.0K Apr 30 19:40 4
drwx----- 2 palak palak 4.0K May  2 07:44 5
drwx----- 2 palak palak 4.0K May  2 07:23 pgsql_tmp
```

Internal Layout of Heap Table File





Analysis of Few Corruption Cases

Case 1: Could not read block



When this occurs

This error appears when PostgreSQL tries to read a specific block from a table or index file and fails—often during query execution or vacuuming.

Case 1: Could not read block



When this occurs

This error appears when PostgreSQL tries to read a specific block from a table or index file and fails—often during query execution or vacuuming.



What are the causes

- Disk corruption
- File system issues
- Accidental tampering

Corrupted Index

```
# dd if=/dev/urandom of=base/16498/16546 bs=8192 count=1 skip=2 conv=notrunc
```

```
test=# select count(*) from pgbench_branches;
```

```
ERROR:  invalid page in block 0 of relation base/16498/16546
```

```
test=# select oid,relname,relkind, relfilenode from pg_class where  
relfilenode = 16546;
```

```
oid | relname | relkind | relfilenode
```

```
-----+-----+-----+-----  
16546 | pgbench_branches_pkey | i | 16546 (1 row)
```

Corrupted Index (Easy Recovery)

```
test=# reindex index concurrently pgbench_branches_pkey  
REINDEX
```

```
test=# select count(*) from pgbench_branches;  
count  
-----  
      1  
(1 row)
```

The Case of the Missing Block

- `$ echo -n "Corrupted Data" | dd conv=notrunc oflag=seek_bytes seek=4000 bs=9 count=1 of=base/16498/16543`
 - WARNING: page verification failed, calculated checksum 9478 but expected 26228
 - ERROR: invalid page in block 0 of relation base/16498/16543
- `Truncate -s 8192 base/16498/16542`
 - ERROR: could not read block 1 of base/16498/16542 : read 0 of 8192 bytes
- Corrupt the metadata for number of blocks
 - Wrong file size is present in the metadata of that file

Case 2: “Could not locate a valid checkpoint record”



When this occurs

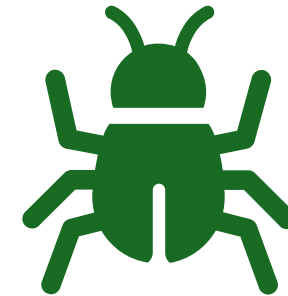
This error happens when the server is unable to read the checkpoint record during recovery upon startup.

Case 2: “Could not locate a valid checkpoint record”



When this occurs

This error happens when the server is unable to read the checkpoint record during recovery upon startup.



What are the causes

Hardware issues
Software bugs

Case of missing WALs

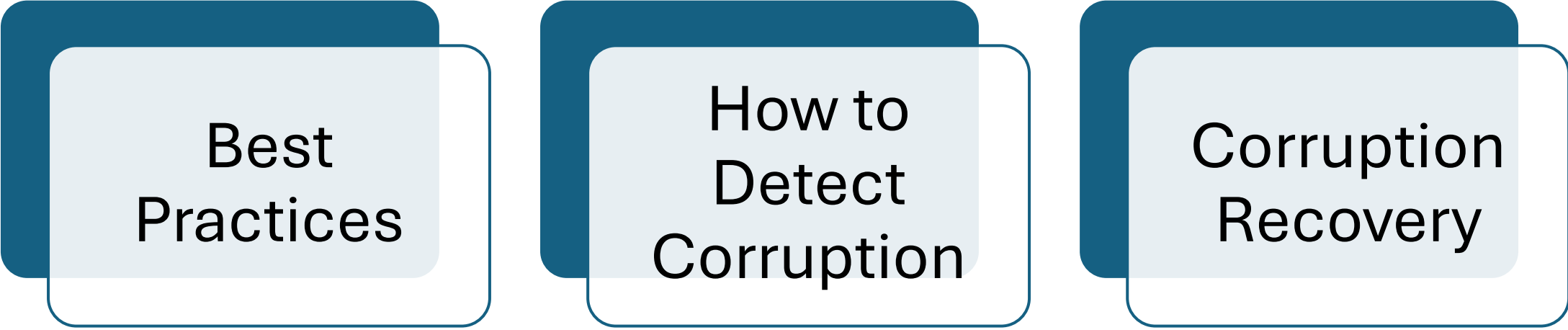
- Retrieve the content of pg control data.
- Latest checkpoint's REDO WAL file is 000000020000000200000054.
- Latest checkpoint's REDO location is 2/54000028.
- pg_waldump
000000020000000200000054--
start=2/54000028
- If the pg_waldump utility cannot read the data, then the WAL data is corrupted.

```
(stdout)
os_hostname,dad0dfe6039a
os_now_utc,2024-12-04T10:03:26Z

pg_control version number:      1300
Catalog version number:        202307071
Database system identifier:     7440566108176953390
Database cluster state:        shut down
pg_control last modified:       Mon 25 Nov 2024 08:39:44 PM UTC
Latest checkpoint location:     2/54000028
Latest checkpoint's REDO location: 2/54000028
Latest checkpoint's REDO WAL file: 000000020000000200000054
Latest checkpoint's TimelineID: 2
Latest checkpoint's PrevTimelineID: 2
Latest checkpoint's full_page_writes: on
Latest checkpoint's NextXID:    0:45774
Latest checkpoint's NextOID:    40961
Latest checkpoint's NextMultiXactId: 1
Latest checkpoint's NextMultiOffset: 0
Latest checkpoint's oldestXID:  722
Latest checkpoint's oldestXID's DB: 1
Latest checkpoint's oldestActiveXID: 0
Latest checkpoint's oldestMultiXid: 1
Latest checkpoint's oldestMulti's DB: 1
Latest checkpoint's oldestCommitTsXid: 0
Latest checkpoint's newestCommitTsXid: 0
Time of latest checkpoint:      Mon 25 Nov 2024 08:39:44 PM UTC
Fake LSN counter for unlogged rels: 0/3e8
Minimum recovery ending location: 0/0
Min recovery ending loc's timeline: 0
Backup start location:           0/0
Backup end location:             0/0
End-of-backup record required:   no
wal_level setting:              logical
wal_log_hints setting:          off
max_connections setting:        859
max_worker_processes setting:   12
max_wal_senders setting:        10
max_prepared_xacts setting:     1718
max_locks_per_xact setting:     64
track_commit_timestamp setting: off
Maximum data alignment:         8
Database block size:            8192
Blocks per segment of large relation: 131072
WAL block size:                 8192
Bytes per WAL segment:          16777216
Maximum length of identifiers:  64
Maximum columns in an index:    32
Maximum size of a TOAST chunk:  1996
Size of a large-object chunk:   2048
Date/time type storage:         64-bit integers
Float8 argument passing:        by value
Data page checksum version:     1
Mock authentication nonce:      2ed7459c6dc4419012c0f5ac89e43856c5cee

(stderr)
```

General Approach to Handling Corruption Cases



Best
Practices

How to
Detect
Corruption

Corruption
Recovery



Best Practices

- Configurations
 - Management
 - Backup and Restore
-

Best Practices: Configurations

1

Don't set fsync=off

Best Practices: Configurations

1

Don't set `fsync=off`

2

Don't set
`full_page_writes=off`

Best Practices: Configurations

1

Don't set `fsync=off`

2

Don't set
`full_page_writes=off`

3

Run with checksums
enabled



Best Practices: Management

1

Do not manually
modify the data
files in any way



Best Practices: Management

1

Do not manually
modify the data
files in any way

2

Don't remove
postmaster.pid

Best Practices: Management

1

Do not manually
modify the data
files in any way

2

Don't remove
postmaster.pid

3

Don't remove WAL
files manually

- [Patch in community](#)

Best Practices: Management

1

Do not manually
modify the data
files in any way

2

Don't remove
postmaster.pid

3

Don't remove WAL
files manually

- [Patch in community](#)

4

Monitor
PostgreSQL and OS
logs for storage-
related errors

Best Practices: Backup and Restore

Take backups regularly.

- If you can take a backup, all of your data is still readable.
- Also, if you have corruption later, you have the option of restoring from your backup

Make sure that you can restore your backups.

- Otherwise, they're not very useful
- Also try to maintain the valid WAL chain irrespective of the retention period. Otherwise, cannot restore from backup.



How to Detect Corruption

- Monitor PostgreSQL Logs
 - Impact of Database Corruption
 - Messages Indicating Corruption:
 - ERROR: could not read block 1064 in file "base/28800/292814141": read only 0 of 8192 bytes
 - PANIC: could not locate a valid checkpoint record.
 - Error Codes:
 - XX001: Indicates data corruption in the database.
 - XX002: Indicates an "index_corrupted" error.
 - pg_amcheck
 - Checksums
-



Corruption Recovery

1. General Approaches
 2. Handling pg_dump Failures
 3. pg_resetwal
-



Corruption Recovery: General Approaches

- **Standby Issue:** Rebuild the standby.
 - **Master Issue:** Restore from backup or failover to an unaffected standby.
 - **Data Reconstruction:** Check if data can be reconstructed from another source.
 - **Last Resort:** Attempt to recover data from the corrupted database only if other options are not feasible.
-



Corruption Recovery: General Approaches

- **Backup First:** Always take a backup before attempting data recovery from a corrupted database.
 - **Recovery Strategy:** Use `pg_dump` to back up the database contents and restore them into a new database created with `initdb`.
 - **Avoid Quick Fixes:** Simply fixing the existing corrupted database can lead to future issues.
-



Corruption Recovery: Handling pg_dump Failures

- Drop problematic objects
 - Use REINDEX
 - Dump selectively
-



Corruption Recovery: pg_resetwal

- pg_resetwal can often enable a corrupted database to start.
 - It is recommended to use pg_resetwal when the database won't start due to lost or corrupted WAL files that cannot be recovered.
 - However, this should be considered a last resort, as it can result in some data loss.
-



Thank you

