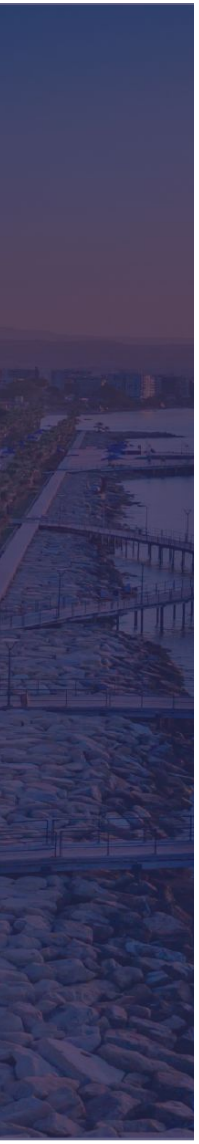
pg_tde

Bringing Transparent Data Encryption to Postgres with Open Source Software

Limassol, Cyprus 2024

PERCONA UNIVERSITY



Transparent Data Encryption (TDE)

TDE offers encryption at file level. It solves the problem of protecting data at rest, encrypting databases both on the hard drive and consequently on backup media. It does not protect data in transit nor data in use. Enterprises typically employ TDE to solve compliance issues such as PCI DSS which require the protection of data at rest.

What's in Postgres

- There are proprietary solutions.
- No partial (selected tables/DBs) or multi-tenancy TDE solution AFAIK.
- Two community patches, one implementing cluster-wide encryption with a single key (2016), and a proposal (2018) and first patch in 2019 implementing table-level encryption and a 2-tier key architecture.

The last message is dated Jan 31, 2020.

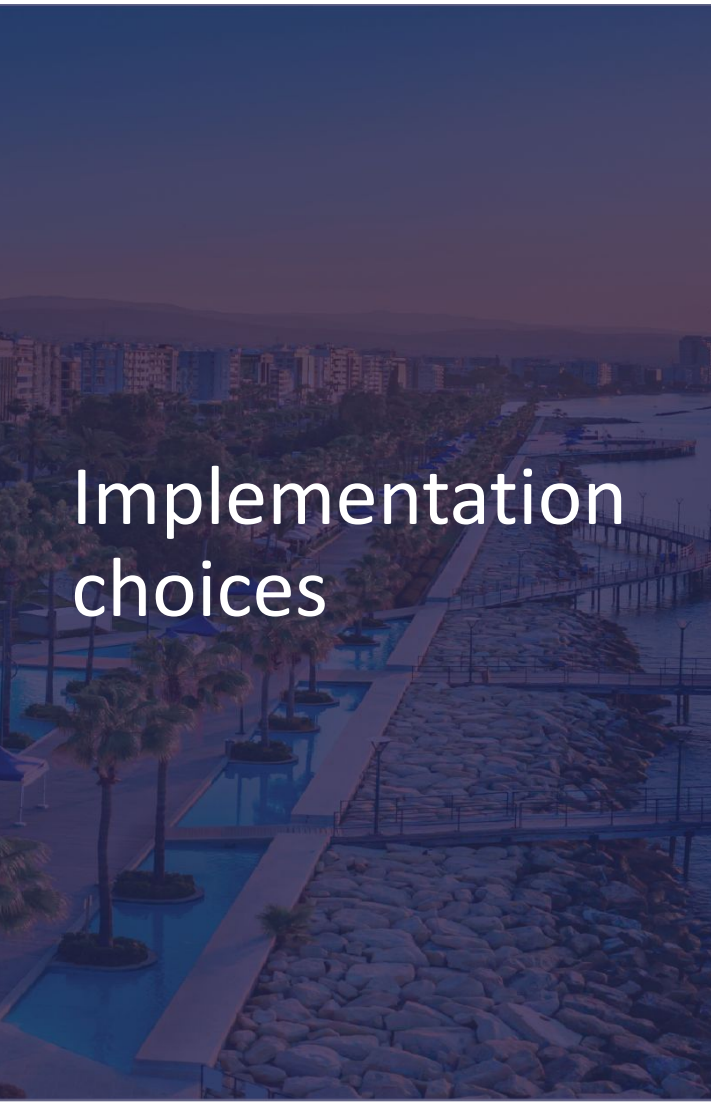
See https://wiki.postgresql.org/wiki/Transparent_Data_Encryption

pg_tde

pg_tde is the extension Encryption (TDE) to PostgreSQL and enables e and secure. It enables users to configure encry ase, encrypting specific tables in some database , while keeping others non encrypted.

currently in v. Alpha1

PERCONA



Implementation choices

Fork vs Extension

Feature	Fork	Extension
Complete cluster encryption	Possible	Not possible
Selective tables encryption	Possible (depending on design)	Possible
Index encryption		
Key rotation		ed syntax
External KMS		
Encrypted metac catalogs		
Existing databas encryption		migration
Table space enc		orkaround
Encrypted backu		o provide a for that
3rd party plugga storage engines encryption		
WAL encryption	Possible (depending on design)	Need further investigation

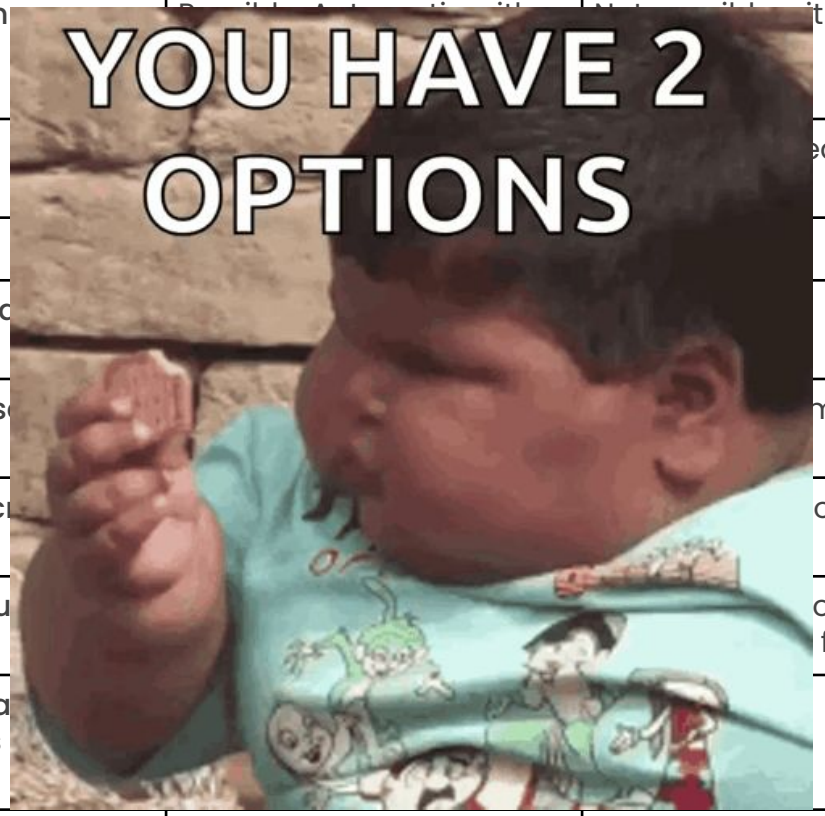
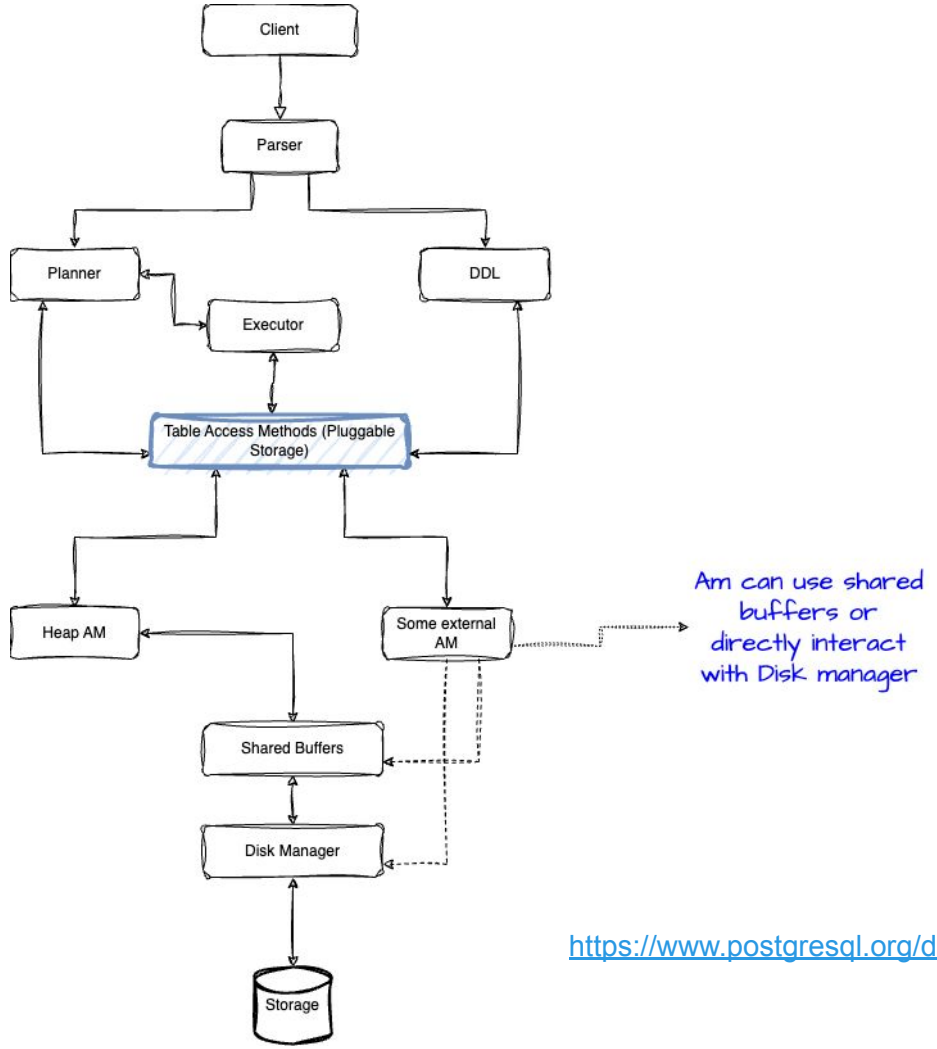


Table Access Methods



https://www.postgresql.org/docs/current/tableam.html

PERCONA

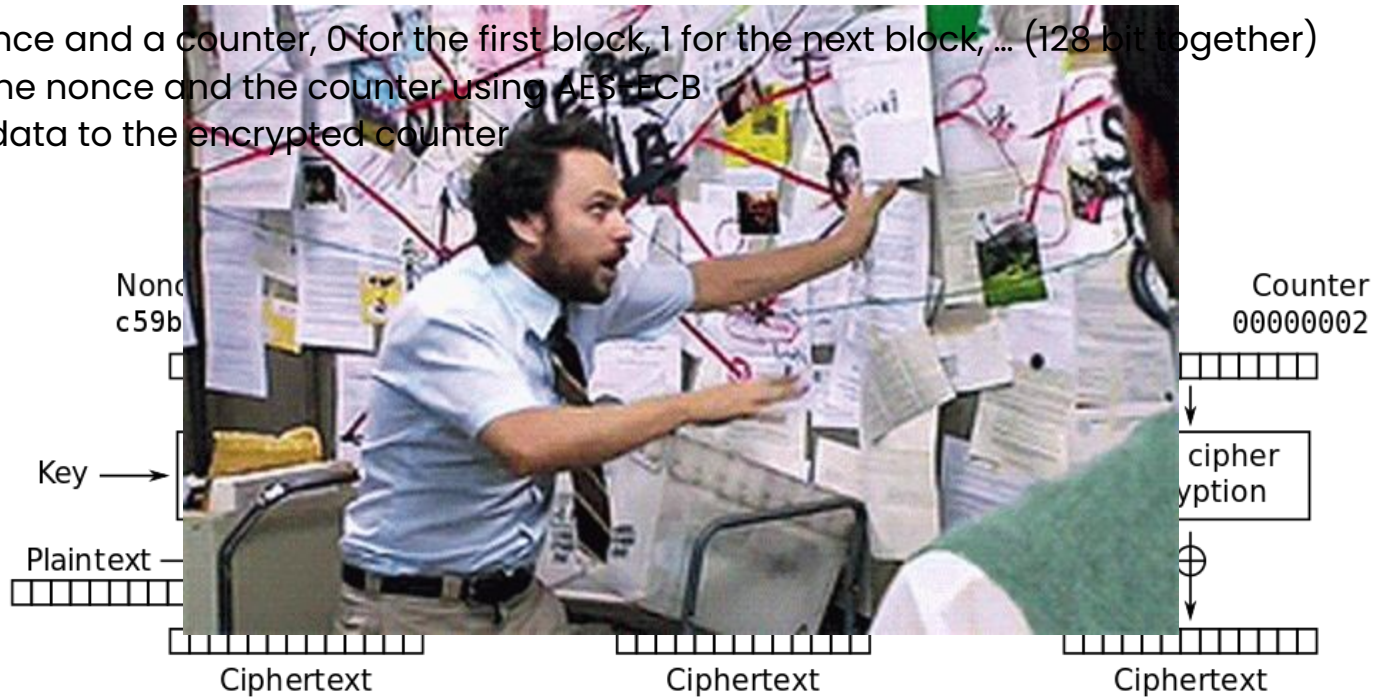
pg_tde extension

```
shared_preload_libraries = 'pg_tde'

-- Access method
CREATE ACCESS METHOD pg_tde TYPE TABLE HANDLER pg_tdeam_handler;

-- User interface
CREATE EXTENSION pg_tde;

SELECT pg_tde_add_key_provider_vault_v2('vault-v2',:'root_token','http://127.0.0.1:8200','secret',NULL);
SELECT pg_tde_set_master_key('vault-v2-master-key','vault-v2');

CREATE TABLE albums (
   id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
   artist VARCHAR(256),
   title TEXT NOT NULL,
   released DATE NOT NULL
) USING pg_tde;
```

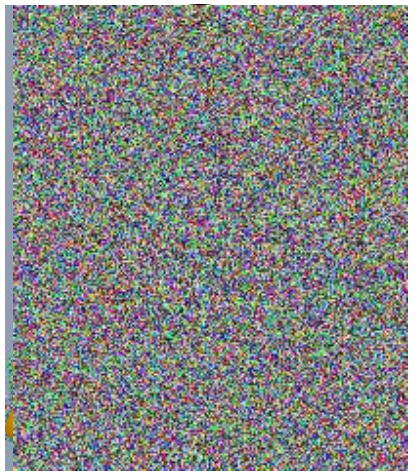
Encryption

AES-CTR:

- Uses a nonce and a counter, 0 for the first block, 1 for the next block, ... (128 bit together)
- Encrypts the nonce and the counter using AES-ECB
- XORs the data to the encrypted counter



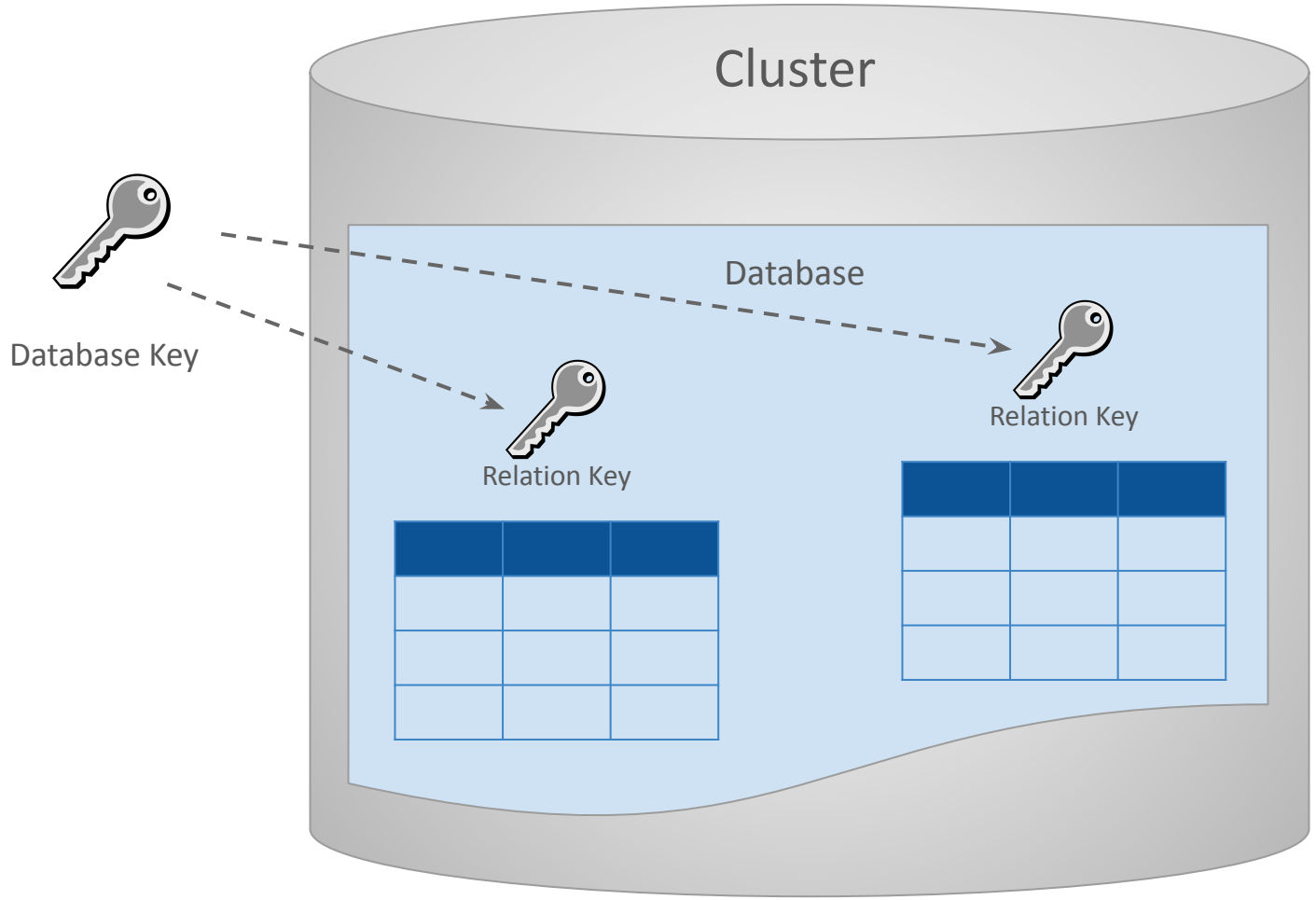
Counter (CTR) mode encryption



https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation#Counter_(CTR)

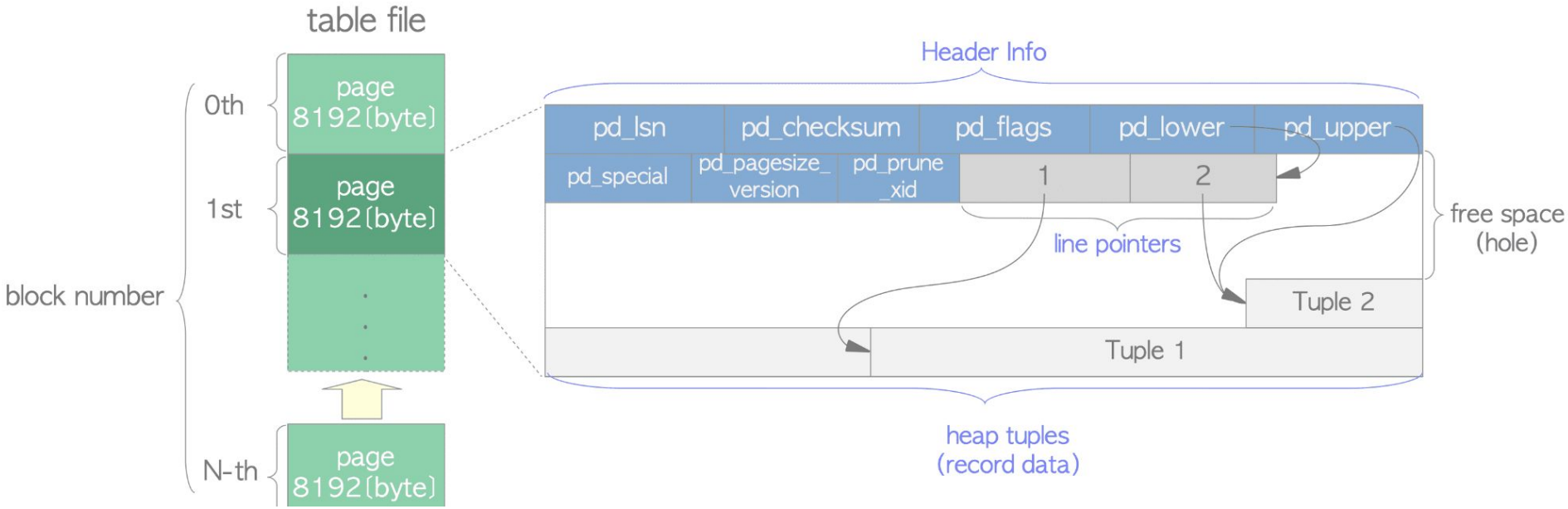
PERCONA

2-tier key architecture



PERCONA

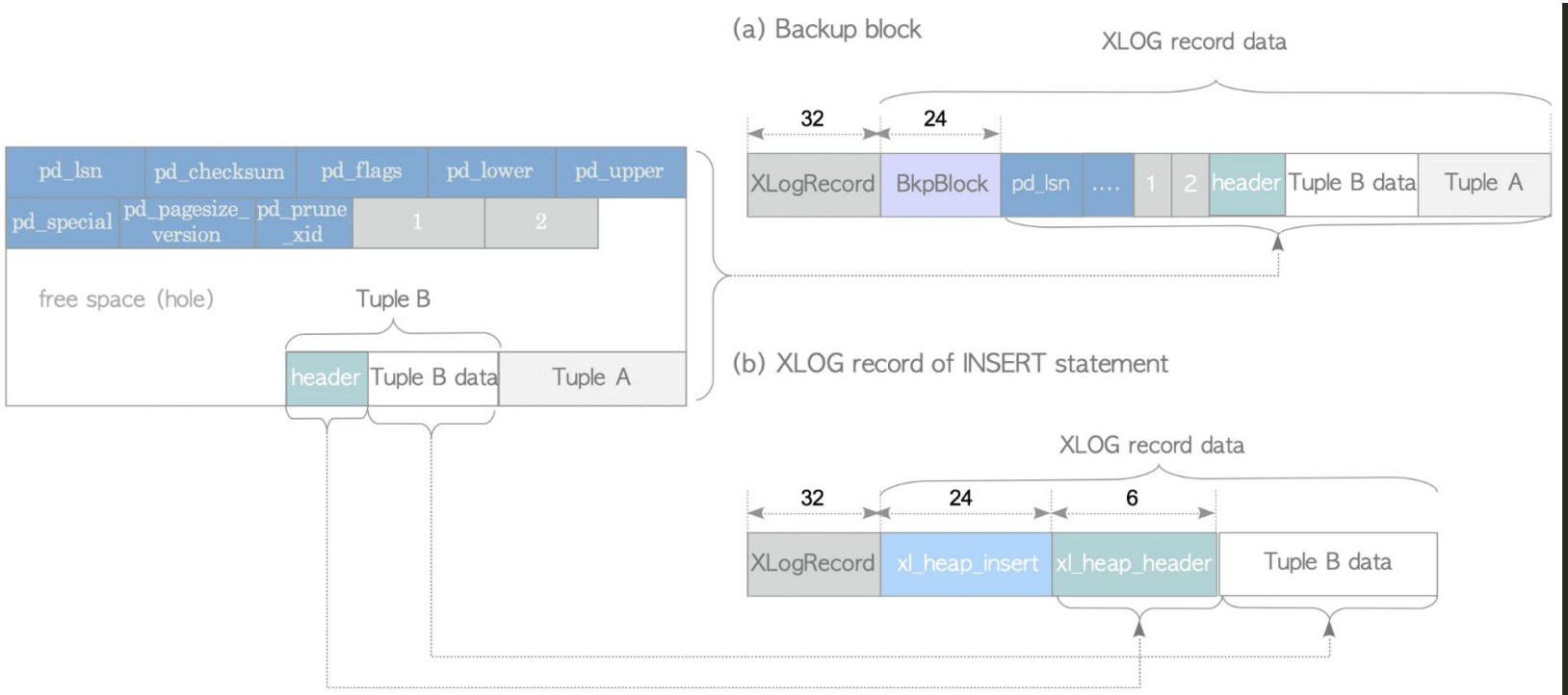
What to encrypt



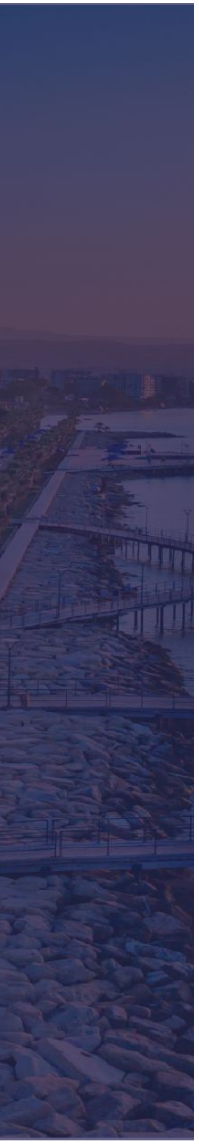
https://www.interdb.jp/pg/pgsql01/03.html, https://www.interdb.jp/pg/pgsql05/02.html

PERCONA

Write Ahead Log & Replication



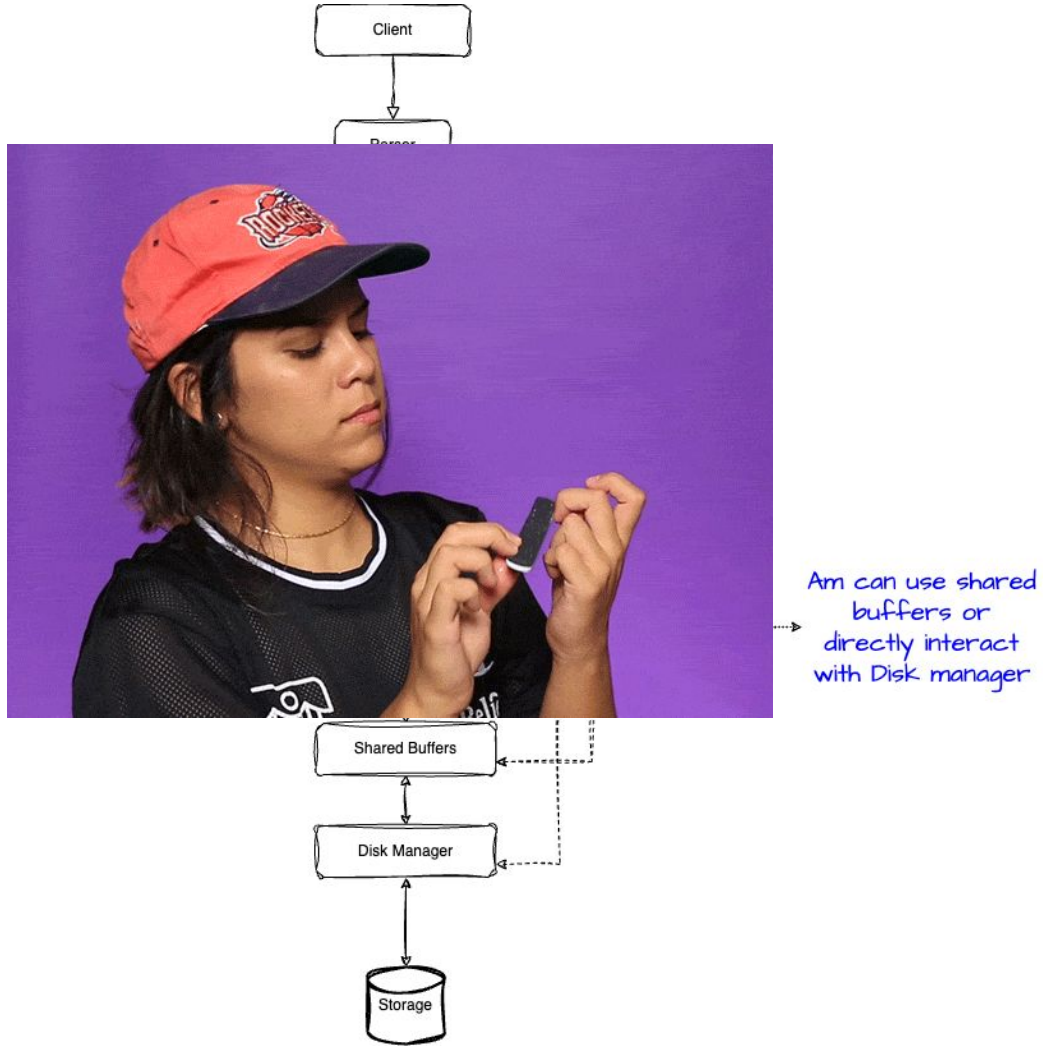

```
typedef struct XLogRecord

{

    uint32 xl_tot_len;  /* total len of entire record */

    TransactionId xl_xid;  /* xact id */

    XLogRecPtr   xl_prev;   /* ptr to previous record in log */

    uint8  xl_info;   /* flag bits, see below */

    RmgrId  xl_rmid; /* resource manager for this record */

    /* 2 bytes of padding here, initialize to zero */

    pg_crc32c  xl_crc;  /* CRC for this record */

    /* XLogRecordBlockHeaders and XLogRecordDataHeader follow, no padding */
} XLogRecord;
```



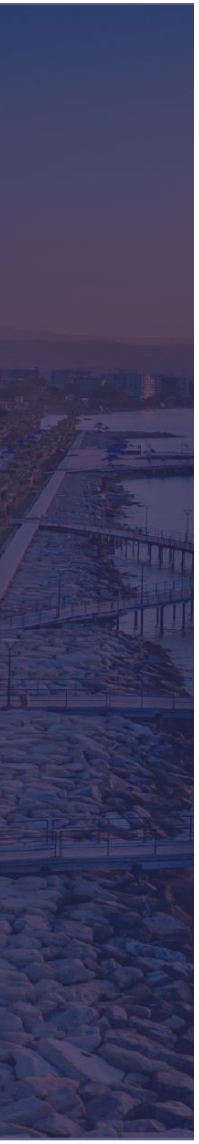
Key Management

- Internal (relation) keys stored in the data catalog
- Master (database) keys either in Hashicorp Vault or FS (not recommended)
- Master key rotation per database
- SQL interface for the key rotation and other management

What about Indexes?



PERCONA



What’s next

Now we’re working on a Postgres fork with extensible storage managers for the relations and WAL.

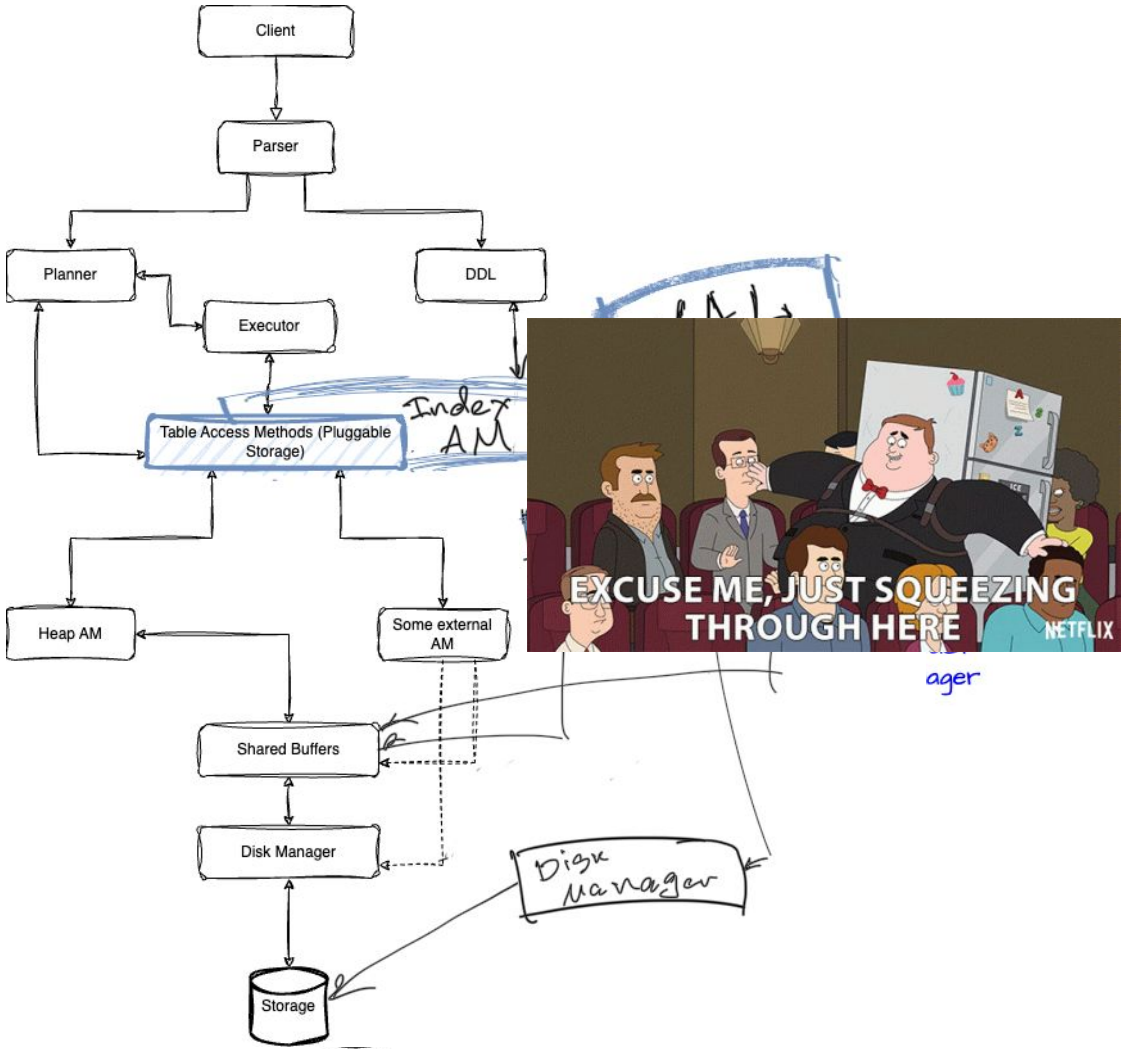
Postgres fork: https://github.com/Percona-Lab/postgres/tree/master_tde

XLog POC draft: https://github.com/Percona-Lab/pg_tde/pull/170

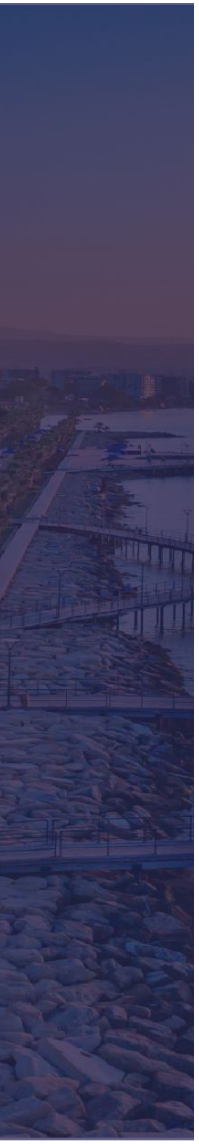
Storage manager: https://github.com/Percona-Lab/pg_tde/pull/178

PERCONA

Disk manager



PERCONA



Links

https://github.com/Percona-Lab/pg_tde

https://percona-lab.github.io/pg_tde/main/index.html

https://www.percona.com/blog/adding-transparent-data-encryption-to-postgresql-with-pg_tde-please-test/

https://www.percona.com/blog/using-the-transparent-data-encryption-extension-pg_tde-with-postgresql/

https://www.interdb.jp/pg/index.html

PERCONA

We’re hiring

[Senior Software Engineer – PostgreSQL]

[PostgreSQL Evangelist]

[Database Performance Consultant – PostgreSQL]

PERCONA

Thank you!