

What's this ruckus about TDE in PG?



Budapest, Hungary 2024

PERCONA
UNIVERSITY

Open Source Databases Meetup

In partnership with





PERCONA
UNIVERSITY

Percona University Budapest

What's all this ruckus about TDE in PostgreSQL?

Monday, July 1 • 2024

Graphisoft Park – Office Park
7 Záhony utca 1031 Budapest Hungary

Open Source Databases
Meetup 2024



Jan Wieremjewicz
Sr. Product Manager,
Percona

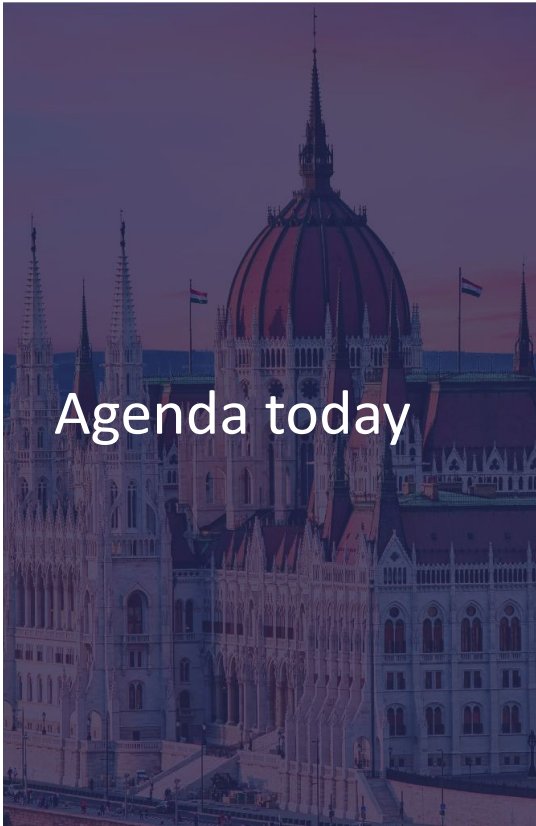
Who am I?



Jan Wieremjewicz

Senior Product Manager, Percona

LinkedIn: <https://www.linkedin.com/in/janwier/>



Agenda today

1. Why encrypt?
2. Storage encryption
3. Transparent data encryption
4. A quick look at PostgreSQL
5. Encryption solutions for PG today
6. Percona contribution 🎉



Budapest, Hungary 2024

PERCONA
UNIVERSITY

Open Source Databases Meetup

Why encrypt?

1. Types of encryption
2. Why encrypt?
3. What to encrypt?



Types of encryption

1 WAY

- Only encrypt and verify
- Never decrypt
- Typical use cases:
 - a. Passwords
 - b. Anonymization

2 WAY

- Encrypt
- Decrypt into meaningful format
- Typical use cases:
 - a. Financial (PAN) data
 - b. PII data

Reasons to protect data



General Data Security

Privacy Protection

Prevention of Data Breaches

Compliance and Trust

States of digital data



Data at rest



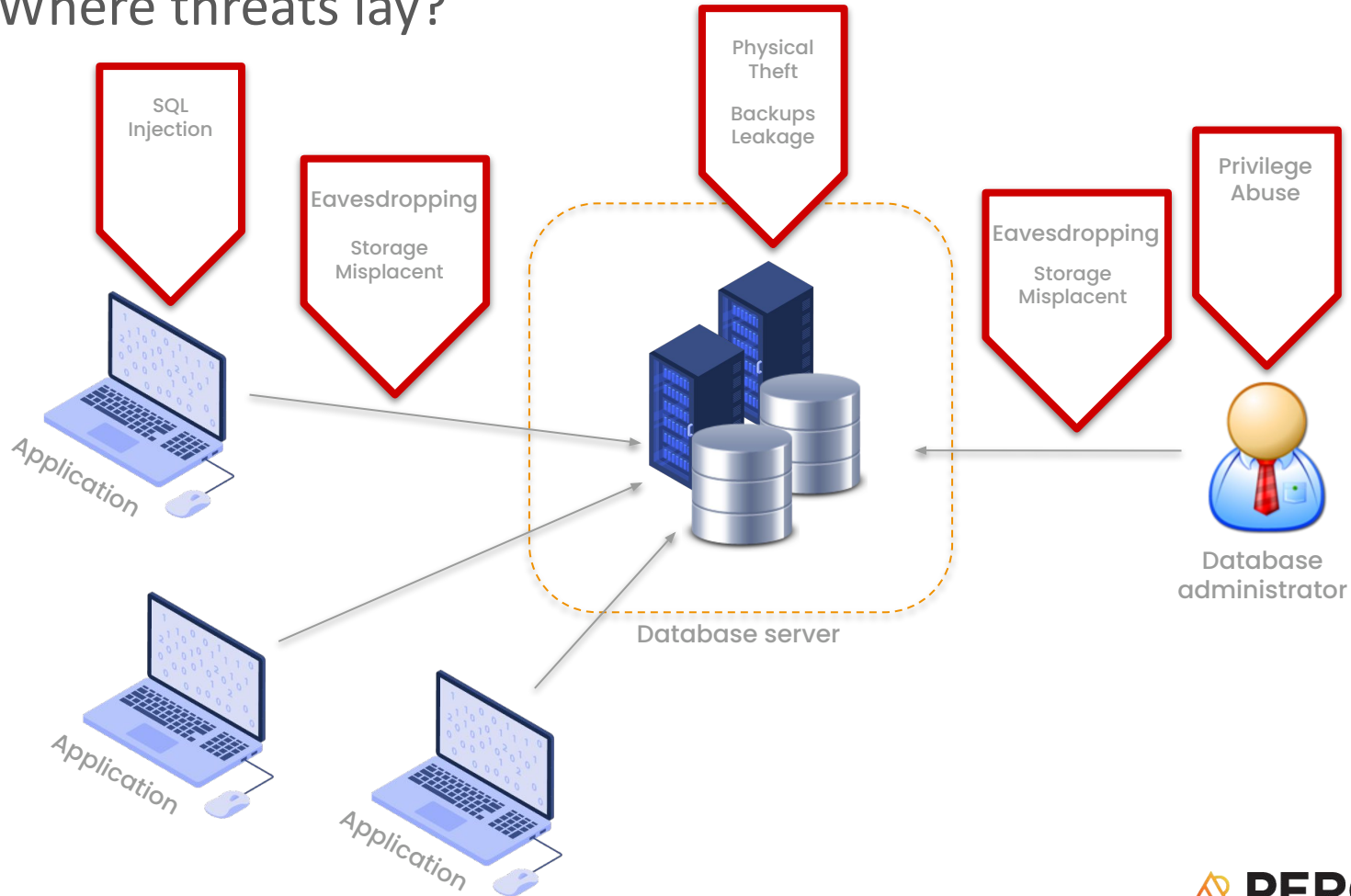
Data in motion



Data in use

- Examples:
- **Static data:**
 - Backups files
 - Archive files
 - Storage devices
 - Database storage engine files
 - **Transmitted data:**
 - Sending emails
 - Sharing files
 - Syncing files
 - Downloads & uploads
 - **Processed data**
 - Office apps
 - Database apps
 - CPU data
 - RAM data
 - Other apps using data

Where threats lay?



Storage Encryption



1. **What?**
2. **Why?**
3. **How?**



Storage Encryption



Storage encryption is the process of encrypting the entire contents of a disk or storage device, including the operating system, files, and data, to protect it from unauthorized access.

11

Why is it Important?

- Data Protection
- Confidentiality
- Compliance Requirements
- Risk Mitigation
- Peace of Mind



Few examples of storage encryption



- **Full Disk Encryption (FDE)**

- Encrypts the entire disk, including the operating system.
- Examples: BitLocker (Windows), FileVault (macOS), dm-crypt/LUKS (Linux).

- **Volume Encryption**

- Encrypts a specific volume or partition within a disk.
- Examples: Veracrypt, BitLocker To Go for external drives, LUKS (Linux).

- **File-Level Encryption**

- Encrypts individual files rather than the entire disk.
- Examples: EFS (Encrypting File System) in Windows, GnuPG (GNU Privacy Guard).

Transparent Data Encryption (TDE)

1. What?
2. Why?
3. How?



Data at rest (DARE): TDE



TDE - Transparent data encryption:

- **Purpose:**
 - Encrypting stored data.
- **How it works:**
 - Storing data in unreadable form, decryptable only with the right keys.
- **Advantages:**
 - Protection against data loss due to theft or unauthorized access.
 - Transparent to application use
 - Ensuring data integrity.
- **Disadvantages:**
 - Performance overhead
 - Potential maintenance overhead
 - Need to secure keys to prevent data loss
 - No restore from backups w/o keys!

Advantages and Challenges of TDE



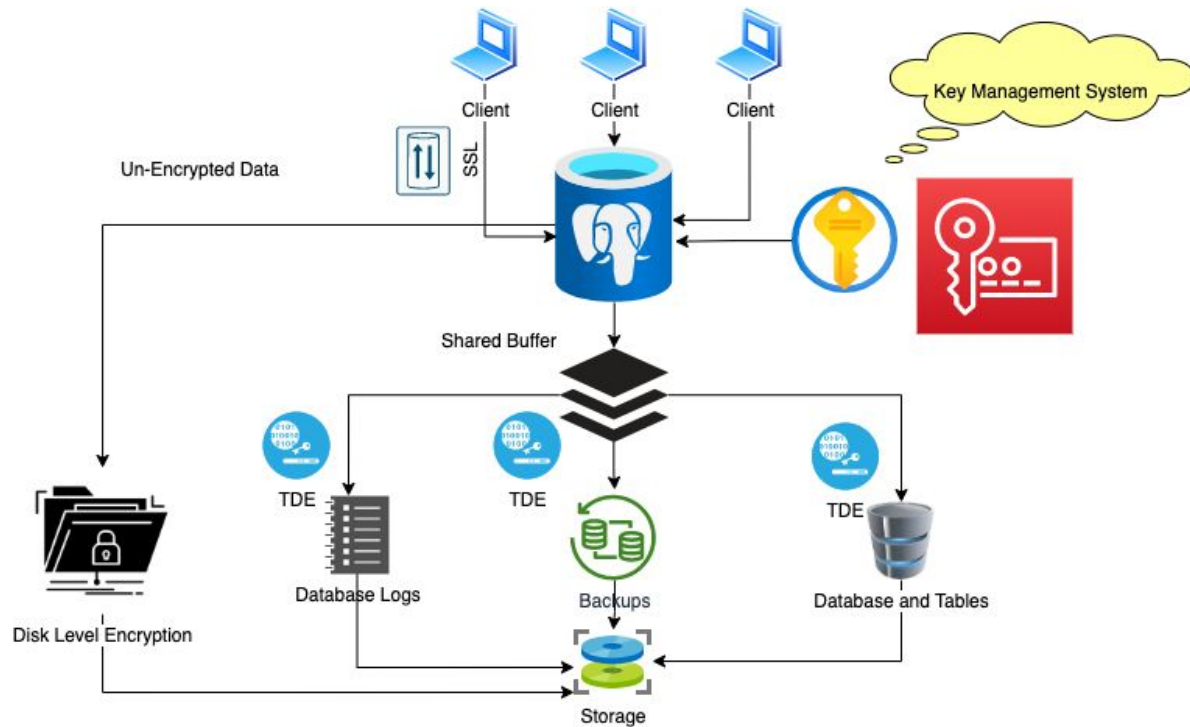
- **Advantages:**

- High security of stored data.
- Easy implementation without changes to applications.

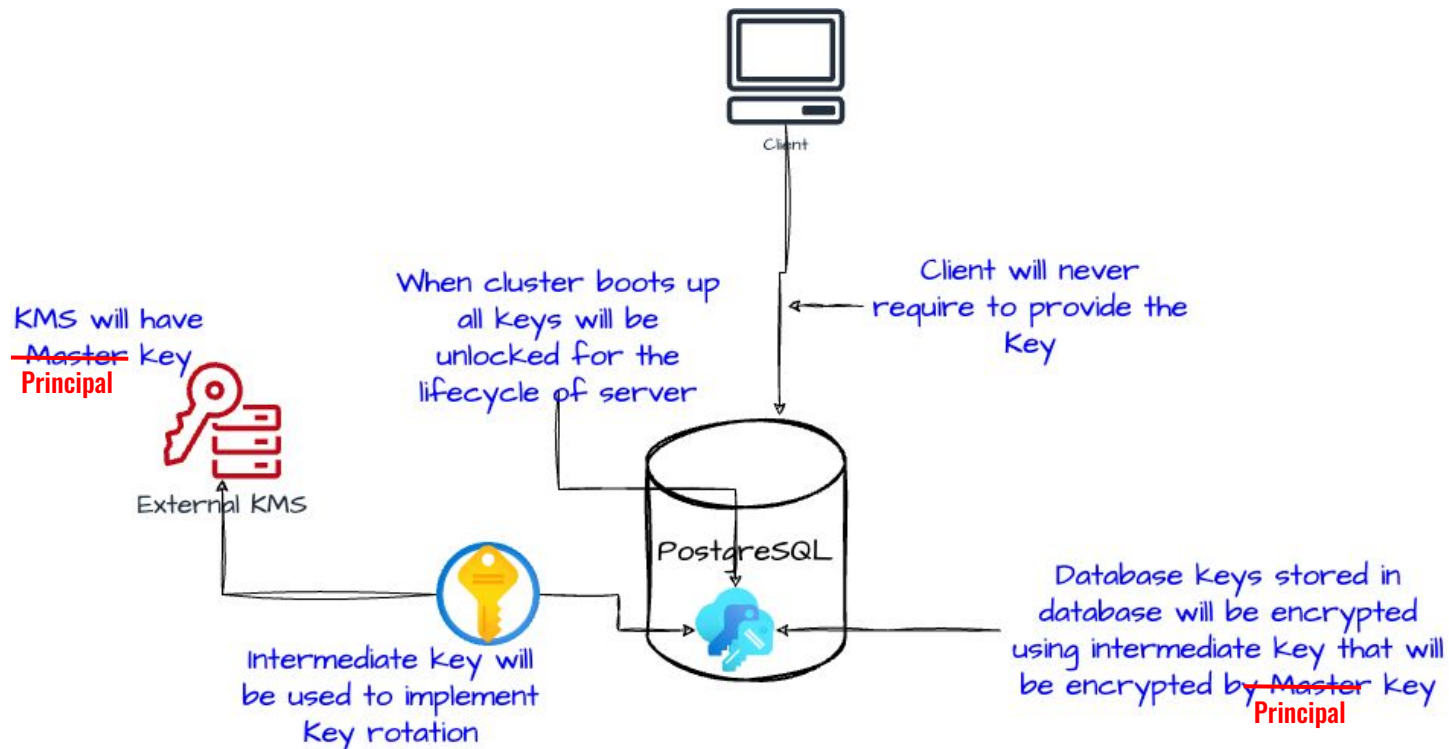
- **Challenges:**

- Complexity of key management.
- Performance overhead due to encryption and decryption.
- Ensuring keys are secure and accessible.

How TDE works?



Encryption Keys



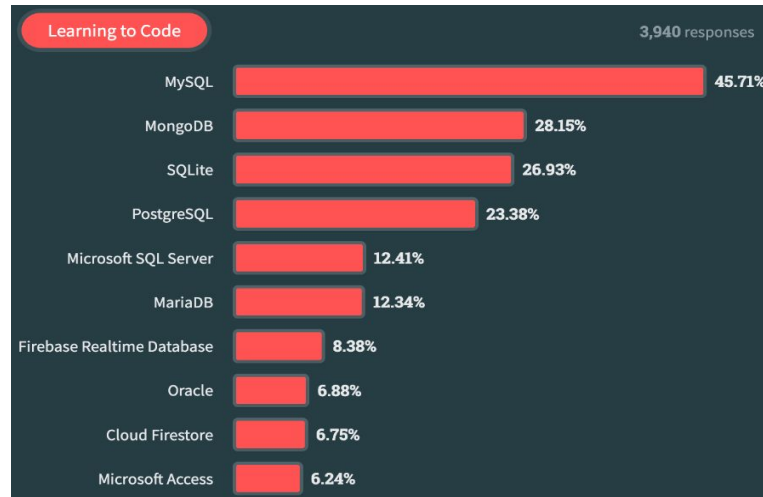
PostgreSQL - quick look



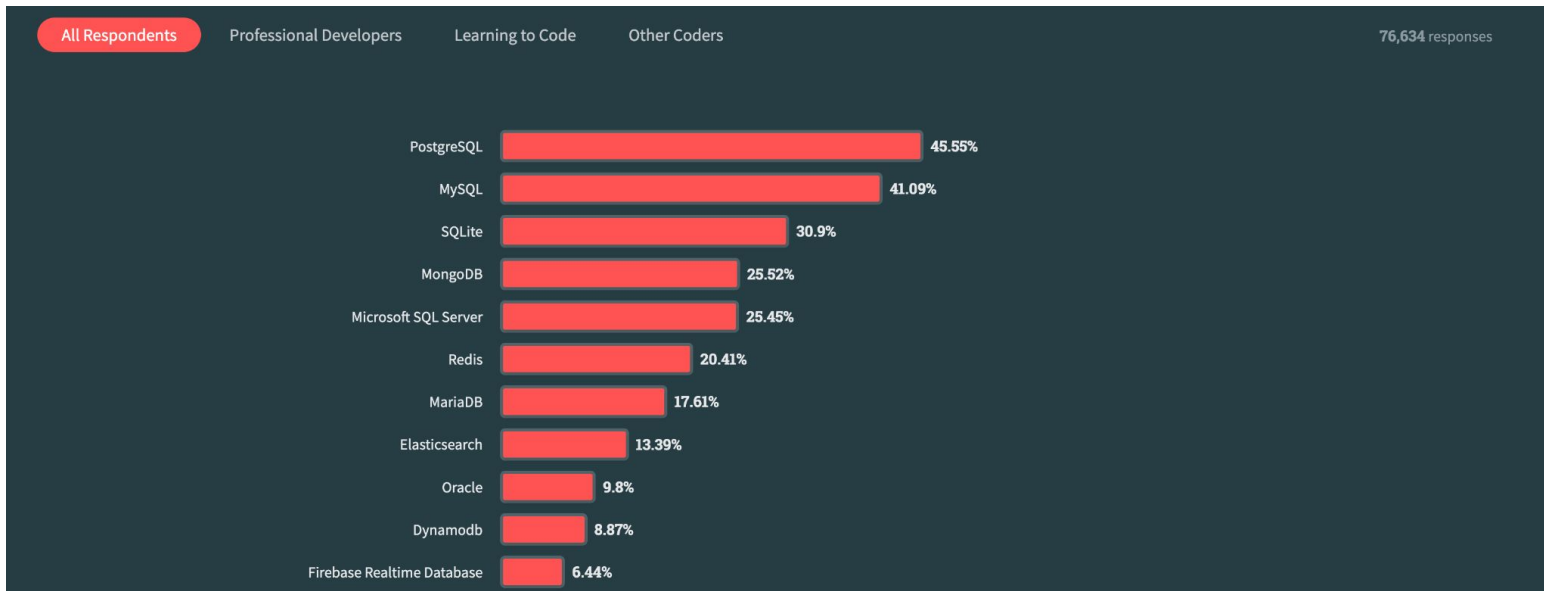
1. **Top databases**
2. **Where is encryption for these?**
3. **What's the situation in Postgres?**



What database do we want to use?



What database do we want to use?



What is the most popular database?



Google

most popular database

X

All Images News Videos Books Web Finance Tools

In the world Top 10 SQL NoSQL Graph Software Model Language

Top 10 Most Popular Databases in 2024

1. Oracle. The Oracle **database** was developed by the company namely Oracle Corporation. ...
2. MySQL. MySQL is an open-source **database** that is **most widely used** across the globe. ...
3. Microsoft SQL Server. ...
4. PostgreSQL. ...
5. MongoDB. ...
6. IBM Db2. ...
7. Redis. ...
8. ElasticSearch.

[More items...](#) • Feb 22, 2024

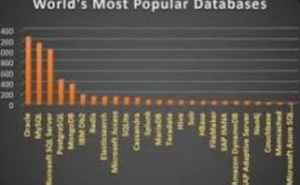
 **GeeksforGeeks**
<https://www.geeksforgeeks.org/most-popular-databases>

Most Popular Databases in 2024 - GeeksforGeeks



Database	Percentage
MySQL	48.88%
PostgreSQL	40.88%
SQL Server	38.75%
Oracle	36.87%
Redis	33.12%
MongoDB	27.08%
IBM Db2	22.25%
ElasticSearch	21.48%

2022 Dev Survey



Database	Percentage
MySQL	48.88%
PostgreSQL	40.88%
SQL Server	38.75%
Oracle	36.87%
Redis	33.12%
MongoDB	27.08%
IBM Db2	22.25%
ElasticSearch	21.48%

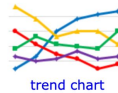
About featured snippets • Feedback

Top databases?

DB-Engines Ranking

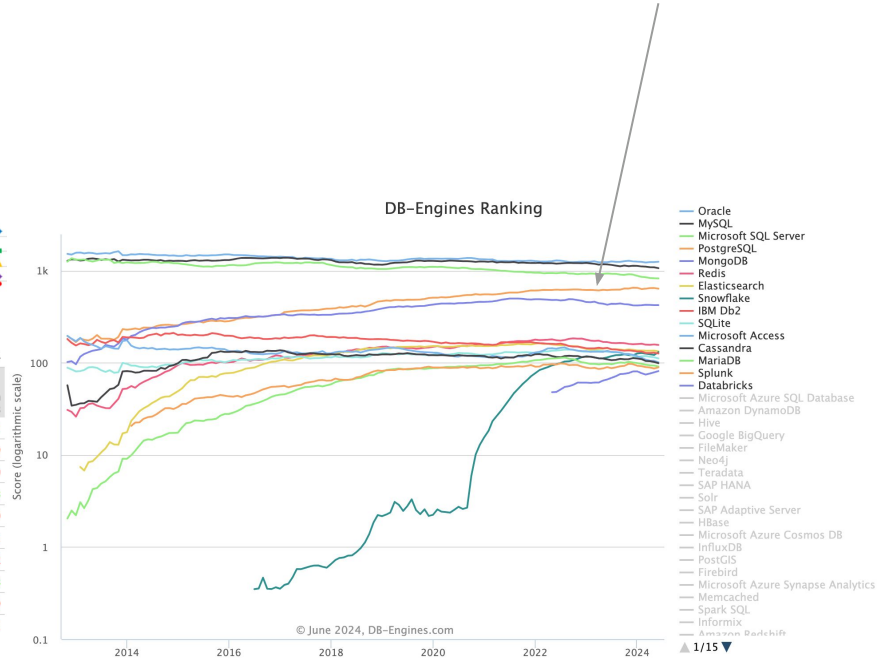
The DB-Engines Ranking ranks database management systems according to their popularity. The ranking is updated monthly.

Read more about the [method](#) of calculating the scores.



421 systems in ranking, June 2024

Rank			DBMS	Database Model	Score		
Jun 2024	May 2024	Jun 2023			Jun 2024	May 2024	Jun 2023
1.	1.	1.	Oracle +	Relational, Multi-model	1244.08	+7.79	+12.61
2.	2.	2.	MySQL +	Relational, Multi-model	1061.34	-22.39	-102.59
3.	3.	3.	Microsoft SQL Server +	Relational, Multi-model	821.56	-2.73	-108.50
4.	4.	4.	PostgreSQL +	Relational, Multi-model	636.25	-9.30	+23.43
5.	5.	5.	MongoDB +	Document, Multi-model	421.08	-0.58	-4.29
6.	6.	6.	Redis +	Key-value, Multi-model	155.94	-1.86	-11.41
7.	7.	8.	Elasticsearch	Search engine, Multi-model	132.83	-2.52	-10.92
8.	9.	11.	Snowflake +	Relational	130.36	+9.03	+16.23
9.	8.	7.	IBM Db2	Relational, Multi-model	125.90	-2.56	-18.99
10.	10.	10.	SQLite +	Relational	111.41	-2.91	-19.81



Top 5 databases vs DARE?



Rank			DBMS
Jun 2024	May 2024	Jun 2023	
1.	1.	1.	Oracle +
2.	2.	2.	MySQL +
3.	3.	3.	Microsoft SQL Server +
4.	4.	4.	PostgreSQL +
5.	5.	5.	MongoDB +

1. Oracle - TDE
2. MySQL - TDE
3. Microsoft SQL Server - TDE
4. PostgreSQL - ?
5. MongoDB - TDE

So is TDE a new topic for PostgreSQL?



- **First patch submitted in 2016**
 - Full cluster encryption
- **TDE adopted by proprietary solutions**
 - EDB Postgres Advanced Server
 - EDB Postgres Extended Server
 - Fujitsu Software Enterprise Postgres
 - Crunchy Hardened PostgreSQL
- **TDE past implementations**
 - Cybertec PostgreSQL Enterprise Edition
 - Available for PostgreSQL 12-15

3 criteria for TDE from PostgreSQL Community

Secure - protecting the data against *all intended attack vectors not against all potential attack vectors*

Minimal impact on the rest of PostgreSQL code:

- Assumed limited group of users interested
- Minimized testing
- Easier to maintained with core changes in future PostgreSQL versions

Meets regulatory requirements

Regulatory requirements

- **PCI DSS** - Requirement 3: Protect Stored Account Data - Payment Card Industry Data Security Standard
- **HIPAA** - Health Insurance Portability and Accountability Act - **requires the disclosure of exposed personal health information but only if they are unencrypted**
 - **HITECH** - extension to HIPAA - requires the disclosure of exposed personal health records but only if they are unencrypted
- **GDPR** - General Data Protection Regulation - requires to notify authorities of personal data breach if data is unencrypted
- **LGPD** (Lei Geral de Proteção de Dados) - Brazilian data protection law
- **CCPA** - California Consumer Privacy Act - implements data breach notification law unless data is encrypted
- **SOC 2** - section CC6: System and Organization Controls, a voluntary compliance certification requires data to be encrypted at all times
- **ISO 27001** - 2013 Annex A.10 - International standard to manage information security, defining cryptography requirements and good practices
- **GLBA** - The Gramm-Leach-Bliley Act - requires financial institutions to inform clients on the security measures to protect their sensitive information

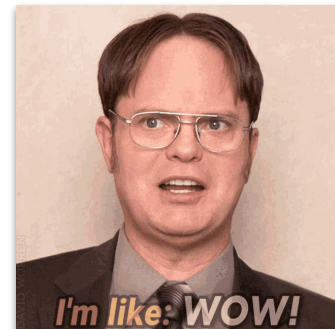
Observation



Encrypted data breach is not considered a data breach

Reason: it cannot be proven that the data has been successfully accessed

Observation



Encrypted data breach is not considered a data breach

Reason: it cannot be proven that the data has been successfully accessed



(...) engineers tend to focus on the technical side and stop on "technical solutions exist" (...)

Business

- Risk management
- Regulatory compliance
- Business scalability

Technical

- More flexibility and granularity in the scope of encrypted objects
- Agnostic of storage
- One point of security for all levels
 - Yes, I also mean backups
- Copying files doesn't expose data



Reasons for TDE?



(...) engineers tend to focus on the technical side and stop on "technical solutions exist" (...)

Business

- Risk management
- Regulatory compliance
- Business scalability

Technical

- More flexibility and granularity in the scope of encrypted objects
- Agnostic of storage
- One point of security for all levels
 - Yes, I also mean backups
- Copying files doesn't expose data



OSS solutions for PG?

1. **pgcrypto**
2. **pgsodium**





- Postgres [contrib module](#) included in vanilla Postgres
- Provides cryptographic functions for PostgreSQL
- Popular use cases:
 - Encrypting/decrypting data on column level
 - Hashing data
 - Generating and verifying digital signatures
 - Random data generation
 - Key generation
 - Hashed password handling



- Not transparent
- Column level only
- No indexes on encrypted columns
- User managed keys
 - No KMS integration
- Severe performance overhead

Why not pgcrypto? - pgsql-hackers Feb 2020



From: Andres Freund

- > I'd strongly advise against having any new infrastructure depend on*
- > pgcrypto. Its code quality imo is well below our standards and contains*
- > serious red flags like very outdated copies of cryptography algorithm*
- > implementations. I think we should consider deprecating and removing*
- > it, not expanding its use. It certainly shouldn't be involved in any*
- > potential disk encryption system at a later stage.*

This is an unsolicited public opinion about pgcrypto.

pgsodium



- PostgreSQL extension enabling algorithms from respected cryptographic library: **libsodium**
- Lightweight extension - heavy lifting done in the lib
- Minimal Server-Key Management
 - Server can preload a libsodium key on server start
- *Transparent Column Encryption** for one or more columns

* not really transparent



- Not transparent
- Column level only
- No indexes on encrypted columns



Percona contribution

1. Assumptions
2. Solution



ENCRYPTION
IS NOT A CRIME

PERCONA
LABS

Assumptions

TRUST

- Open culture:
 - Open source
 - Transparency
- Follow community:
 - Principles
 - Guidelines
 - Rules
- Establish partnership(s)

EASE

- Ease of adoption:
 - Foot in the door
 - Not only cluster
 - Multitenant
- Ease of use

VALUE

- Trust needs time:
 - Time to value
 - Extension vs fork
- Enterprise needs

3 criteria for TDE from PostgreSQL Community: **Reminder**

Secure - protecting the data against *all intended attack vectors not against all potential attack vectors*

Minimal impact on the rest of PostgreSQL code:

- Assumed limited group of users interested
- Minimized testing
- Easier to maintained with core changes in future PostgreSQL versions

Meets regulatory requirements

Assumptions put into plan



STEP 1

- **Start with PoC**
- **Focus on time to market**
- **Deliver a solution compatible for most**
 - Cut scope if needed
- **Have assumptions in mind**

STEP 2

- **Deliver rest of scope**
- **Focus on covering the enterprise needs**
- **Build community and partnerships**



Step 1: pg_tde

Minimal impact on the rest of PostgreSQL code:

- Assumed limited group of users interested
- Minimized testing
- Easier to maintained with core changes in future PostgreSQL versions

+

Deliver a solution compatible for most

=

Extension

→ no PG core changes

- **Fully open source - MIT license**

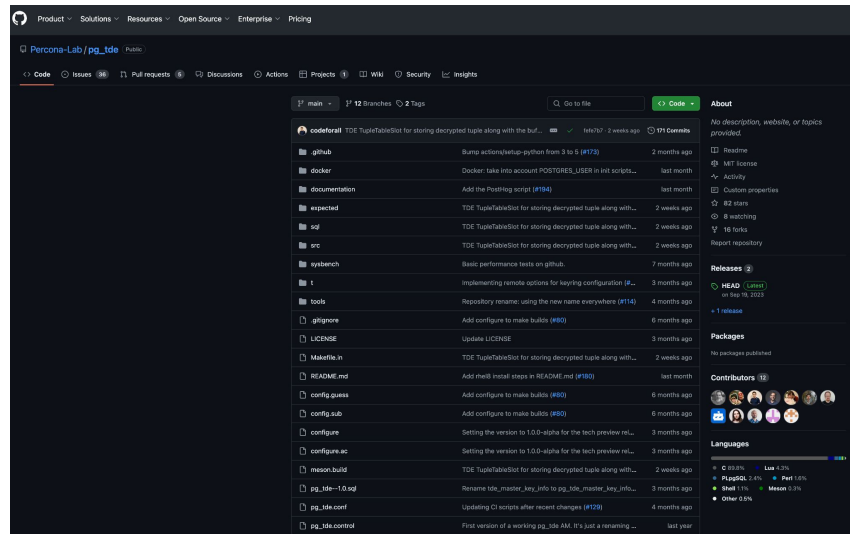
- no strings attached
- No vendor lock-in

- **Introducing new encrypted access method to PostgreSQL**

- Allows encryption of tables

- **Disadvantages of going with A extension**

- Cannot encrypt indexes



So....what's encrypted in the extension

- **User data in tables**
 - including TOAST tables, that are created using the extension.
- **Write-Ahead Log (WAL) data for tables created using the extension**
- **Temporary tables created during the database operation for data tables created using the extension**



So....what's not encrypted?



No index encryption



When life gives you lemons...

What about indexes?

We cannot warrant that no sensitive data is there!

...

Extension cannot encrypt
tables and indexes

= **LEMON**





In short - if you don't need indexes to be encrypted (no sensitive data indexed) use **the extension** alone, otherwise go with **extension and the patch**.

A

```
CREATE TABLE my_table (id SERIAL,  
pii_data VARCHAR(32), PRIMARY  
KEY(id)) USING pg_tde;
```

This encrypts only the table.

Also allows WAL encryption
(optional) for the whole cluster

Extension

B

Everything in option A and more

```
CREATE TABLE my_table (id SERIAL,  
pii_data VARCHAR(32), PRIMARY  
KEY(id)) USING pg_tde_full;
```

This encrypts the table and
everything related.

System tables for now stay not encrypted.
Potential future increment.

Extension

+

Patch

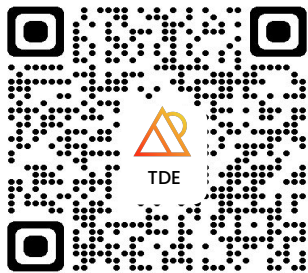
If you know better. Let
us know.

Superior features of pg_tde

- **Multitenancy**
 - Separate key per database
- **Key-Management**
- **Key-Rotation**
- **Table level granularity**
- **Vanilla Support or binary compatibility (drop-in)**



What's next?



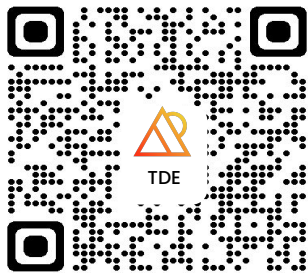
GitHub: https://github.com/Percona-Lab/pg_tde



1. Give it a try!
2. Stay active, share feedback!



What's next?



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



- We're working closely with upstream, to get needed SMGR changes into PostgreSQL 18.
 - If this succeeds, there is no need for a patch anymore
 - "Only" XLog would be left and we would focus on getting this change, also into the core
- End Goal - Full Encryption through an extension, without the need to patch PostgreSQL - Simply works with Vanilla/Upstream.



Rest assured!
Index encryption is coming!

Survey - We need your feedback



<https://shorturl.at/JqsWr>



Questions?



Thank You!

....