Beyond GDPR

Anonymization

true

Anonymization

Beyond GDPR

TITRE : Anonymization
SOUS-TITRE : Beyond GDPR

WHO I AM

- Damien Clochard
- PostgreSQL DBA & Co-founder at Dalibo
- President of PostgreSQLFr Association

WHO I AM NOT

- I Am Not A Lawyer
- I Am Not A Privacy Expert
- Don't take my word for it / Check the links !

MY STORY

MENU

- GDPR: 1 year later
- Why Anonymization is hard
- Anonymization Pipelines
- PostgreSQL Anonymizer

---

true

Anonymization

GDPR

- Individual Rights
- Principles
- Impact
- Pseudonymization vs Anonymization

GDPR: INDIVIDUAL RIGHTS

- The right to be informed
- The right of access
- The right to rectification
- **The right to erasure**
- **The right to restrict processing**
- The right to data portability
- The right to object
- etc.

(source: Individual Rights)

GDPR: PRINCIPLES & CONCEPTS

- Lawfulness, fairness and transparency
- Security
- Data Minization
- **Privacy By Design**
- Data Protection By Design
- **Pseudonymization**
- **Storage Limitation**
- Accuracy
- Purprose Limitation

(source: GDPR Principles)

true

SANCTIONS ARE COMING

- July 2019 : Marriott (UK) fined 110M€
- July 2019 : British Airways (UK) fined 204 M€
- June 2019 : Sergic (France) fined 400 k€
- June 2019 : LaLiga (Spain) fined 250 k€
- May 2019 : Municipality of Bergen (Norway) fined 170 k€
- April 2019 : Airbus (France) fined 200k€
- And many more

(source: GDPR Enforcement Tracker)

BEWARE OF ARTICLE 32 !

Most sanctions are linked to Article 32:

« Insufficient technical and organisational measures to ensure information security »

(source Article 32 - Security of processing)

IN OTHER WORDS: ``DATA LEAKS"

PSEUDONYMIZATION

« Personally identifiable information is pseudonymised when it is modified in a way that it can no longer be linked to a single data subject without the use of additional data. »

ANONYMIZATION

Not even mentioned in the GDPR !

DOES IT REALLY MATTER ?

YES

Pseudonymized data still falls within the scope of the Regulation.

---

2 DIFFERENT THINGS

- Pseudonymization is a security requirement
- Anonymization is an exit door

---

true

PSEUDONYMIZATION

The **additional data** should be kept separate from the pseudonymized data and subject to technical and organisational measures to make it hard to link a piece of data to someone's identity

---

EXAMPLE: ENCRYPTION

Encryption is not anonymization !

Encrypted data are still covered by GDPR because the original data can be retrieved with the encryption key.

WHY ANONYMIZATION IS HARD

- Singling out
- Linkability
- Inference

(source: WP29 Opinion on Anonymisation Techniques)

SINGLING OUT

The possibility to isolate a record and identify a subject in the dataset.

```
SELECT * FROM employees;
```

```
  id   |  name           | job  | salary
-------+-----------------+------+--------
 1578  | xkjefus3sfzd    | NULL |    1498
 2552  | cksnd2se5dfa    | NULL |    2257
 5301  | fnefckndc2xn    | NULL |   45489
 7114  | npodn5ltyp3d    | NULL |    1821
```

LINKABILITY

Identify a subject in the dataset using other datasets

- Netflix Ratings + IMDB Ratings
- Hospital visits + State voting records

true

Anonymization

(sources: Netflix prize + Hospital Reidentification)

INFERENCE

Identify a subject using a set of indirect identifiers.

87% of the U.S. population are uniquely identified by date of birth, gender and zip code

(source : Latanya Sweeney)

THIS IS A LOSING GAME !

you can't prove that **re-identification** is impossible

(source: De-indentification still doesn't work)

GDPR GIVES A MARGIN OF ERROR

« To determine [if] a person is identifiable, account should be taken of all the means **reasonably likely to be used** [...] to identify the person directly or indirectly.

« To ascertain whether means are reasonably likely to be used to identify the person, account should be taken of all objective factors, such as the **costs** of and the **amount of time** required for identification, taking into consideration the **available technology at the time** of the processing »

(source: Recital 26)

MESURE THE THREAT

This means you have to measure the “reasonable risk” of re-identification, on a regular basis.

ANONYMIZATION PIPELINES

Minimizing the risk of data leaks by reducing the attack surface

This is a direct implementation of the **“Storage Limitation”** principle

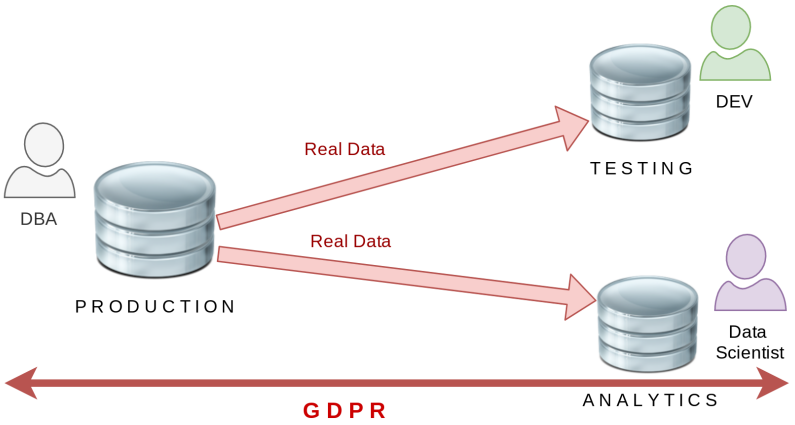
---

BASIC EXAMPLE

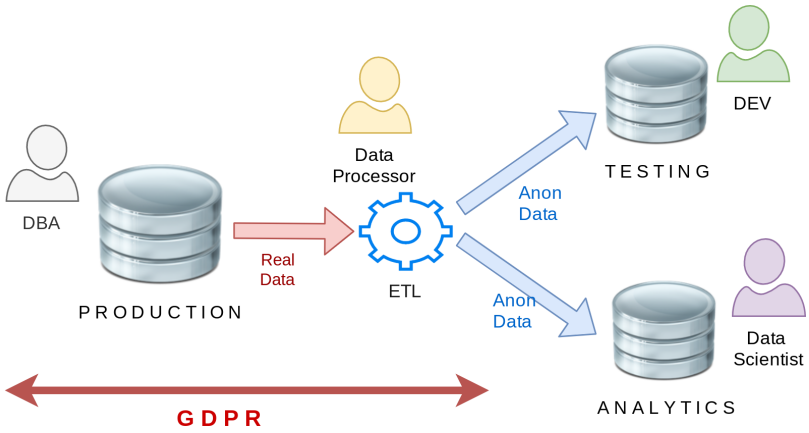
---

true

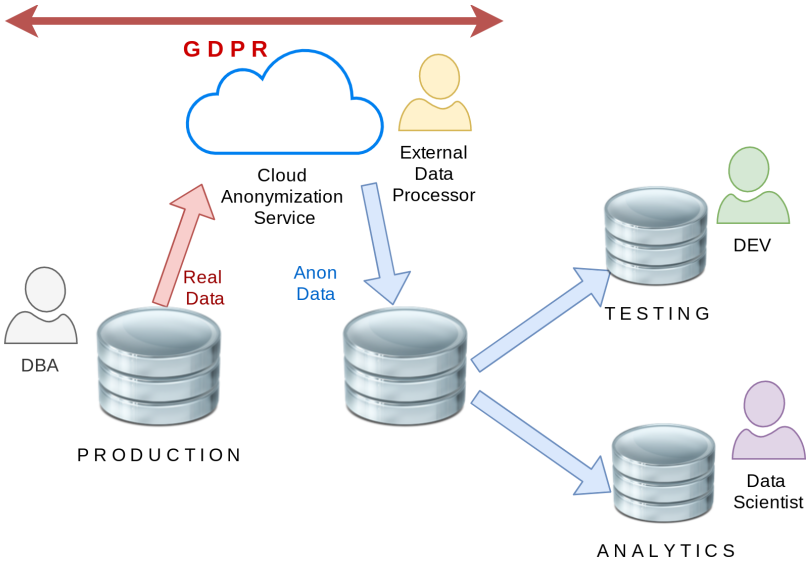
WORST SCENARIO



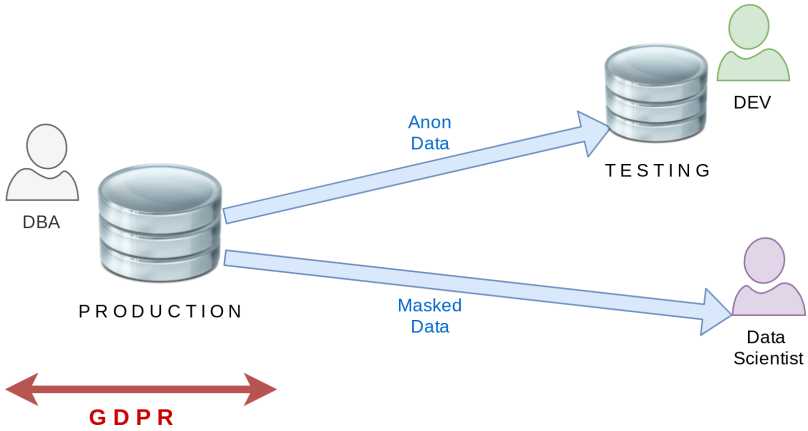
ETL



CLOUD ANONYMIZATION



POSTGRESQL ANONYMIZER



---

true

PostgreSQL Anonymizer

WHAT IS THIS ?

- Started as a personal project last year
- Now part of the “Dalibo Labs” initiative
- This is a prototype !
- Currently in version 0.4

GOALS

- Declare masking rules within the database model
- Anonymization is done internally
- Dynamic Masking or In-Place Substitution
- Batteries included : Builtin masking functions
- Inspired by MS SQL Server Dynamic Data Masking

EXAMPLE: REAL DATA

```
=# SELECT * FROM customer;
 id  |   full_name      |   birth    | zipcode | fk_shop
-----+------------------+------------+---------+---------
 911 | Chuck Norris     | 1940-03-10 | 75001   | 12
 112 | David Hasselhoff | 1952-07-17 | 90001   | 423
```

EXAMPLE: ANONYMIZED DATA

```
=# SELECT * FROM customer;
 id  |     full_name     |   birth    | zipcode | fk_shop
-----+-------------------+------------+---------+---------
 911 | Michel Duffus     | 1970-03-24 | 63824   | 12
 112 | Andromache Tulip  | 1921-03-24 | 38199   | 423
```

INSTALL

```
$ sudo pgxn install ddlx
$ sudo pgxn install postgresql_anonymizer
```

INSTALL

Using the Community RPM Repo:

```
$ yum install https://.../pgdg-redhat-repo-latest.noarch.rpm
$ yum install postgresql_anonymizer12
```

(thanks Devrim !)

---

true

CONFIGURE

```
shared_preload_libraries = '[...], anon'
```

LOAD

```
=# CREATE EXTENSION IF NOT EXISTS anon CASCADE;
=# SELECT anon.load();
```

DECLARE A MASKING RULE

```
SECURITY LABEL FOR anon
ON COLUMN customer.zipcode
IS 'anon.random_zipcode()';
```

(thanks Alvaro !)

NOW WE HAVE 3 OPTIONS

- In-Place Anonymization
- Anonymous Dumps
- Dynamic Masking

IN-PLACE ANONYMIZATION

```
=# SELECT anon.anonymize_column('customer','zipcode');
```

```
=# SELECT anon.anonymize_table('customer');
```

```
=# SELECT anon.anonymize_database();
```

IN-PLACE ANONYMIZATION

This will update all lines of all tables containing at least one masking rule.

This is gonna be slow and trigger heavy write workloads.

ANONYMOUS DUMPS

```
=# SELECT anon.dump();
```

ANONYMOUS DUMPS

```
$ psql [...] -qtA -c 'SELECT anon.dump()' your_dabatase > dump.sql
```

DYNAMIC MASKING

Let's take a basic example :

```
=# SELECT * FROM people;
 id | fistname | lastname |   phone
----+----------+----------+------------
 T1 | Sarah    | Conor    | 0609110911
(1 row)
```

---

true

DYNAMIC MASKING

Step 1 : Activate the dynamic masking engine

```
=# CREATE EXTENSION IF NOT EXISTS anon CASCADE;
=# SELECT anon.start_dynamic_masking();
```

DYNAMIC MASKING

Step 2 : Declare a masked user

```
=# CREATE ROLE skynet LOGIN;
=# SECURITY LABEL FOR anon ON ROLE skynet
-# IS 'MASKED';
```

The masked user has a read-only access to the anonymized data of the masked tables.

DYNAMIC MASKING

Step 3 : Declare the masking rules

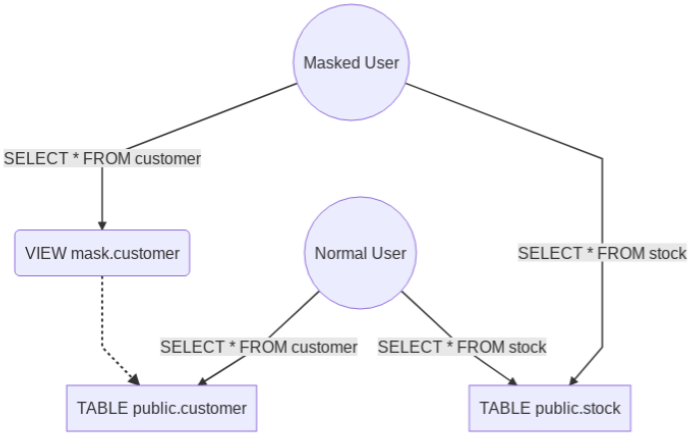
```
SECURITY LABEL FOR anon ON COLUMN people.name
IS 'MASKED WITH FUNCTION anon.random_last_name()';

SECURITY LABEL FOR anon ON COLUMN people.phone
IS 'MASKED WITH FUNCTION anon.partial(phone,2,$$******$$,2)';
```

DYNAMIC MASKING

Step 4 : Connect with the masked user

```
=# \! psql peopledb -U skynet -c 'SELECT * FROM people;'
 id | fistname | lastname  |   phone
----+----------+-----------+------------
 T1 | Sarah    | Stranahan | 06******11
(1 row)
```

HOW IT WORKS



HOW IT WORKS

Basically :

- 500 lines of pl/pgsql
- An event trigger on DDL commands
- Silently creates a “masking view” upon the real table
- Tricks masked users with `search_path`
- use of TABLESAMPLE with `tms_system_rows` for random functions

MASKING FUNCTIONS

The extension provides functions to implement 5 main anonymization techniques:

- Noise Addition
- Shuffling / Permutation
- Randomization
- Faking / Synthetizing
- Partial destruction

true

NOISE ADDITION

```
=# SECURITY LABEL FOR anon
-# ON COLUMN employee.salary
-# IS 'MASKED WITH FUNCTION
-#     anon.add_noise_on_numeric_column(user, salary, 0.33)
-# ';
```

All values of the column will be randomly shifted with a ratio of +/- 33%

NOISE ADDITION

- The dataset remains meaningful
- `AVG()` and `SUM()` are similar to the original
- works only for dates and numeric values
- “extreme values” may cause re-identification (“singling out”)

SHUFFLING

```
=# SECURITY LABEL FOR anon
-# ON COLUMN employee.fk_company
-# IS 'MASKED WITH FUNCTION
-#     anon.shuffle_column(employee, fk_company, id)
-# ';
```

SHUFFLING

- The dataset remains meaningful
- Perfect for Foreign Keys
- Works bad with low distribution (ex: boolean)
- The table must have a primary key

RANDOMIZATION

```
=# SECURITY LABEL FOR anon
-# ON COLUMN employee.birth
-# IS 'MASKED WITH FUNCTION
-#     anon.random_date_between(''01/01/1920'',now())
-#';
```

RANDOMIZATION

- Simple and Fast
- Usefull for columns with `NOT NULL` constraints
- Useless for analytics

FAKING

```
=# SECURITY LABEL FOR anon
-# ON COLUMN employee.lastname
-# IS 'MASKED WITH FUNCTION
-#     anon.fake_last_name()
-# ';
```

---

true

FAKING

- Just a more elaborate version of Randomization
- Great for developpers and CI tests
- You can load your own dictionnaries !

PARTIAL DESTRUCTION

```
=# SECURITY LABEL FOR anon
-# ON COLUMN employee.phone
-# IS 'MASKED WITH FUNCTION anon.partial(phone,4,'******',2)';
```

+33142928107 becomes +331******07

PARTIAL DESTRUCTION

- Perfect for phone number, credit cards, etc.
- The user can still recognize his/her own data
- Transformation is `IMMUTABLE`
- Works only for TEXT / VARCHAR types

KNOWN LIMITATIONS

- PostgreSQL 9.6 and later
- Dynamic Masking works with only one schema

FUTURE DEVELOPMENTS

- Research on K-Anonymity
- Mesure the risk of reidentification
- Suggest masking rules based on heuristics
- Implement Generalization functions

OTHER TOOLS FOR POSTGRES

- Differential Privacy extension by Google
- Smart Sampling with pg_sample
- pgantomizer

HOW TO CONTRIBUTE ?

- Feedback and bugs !
- Images and geodata
- Join the project at :

https://gitlab.com/dalibo/postgresql_anonymizer

IN A NUTSHELL

- GDPR sanctions are really real
- Data Leak is your main risk
- Reduce your attack surface (“Storage Limitation”)
- Anonymize whenever you can
- Anonymize inside the database
- Encryption is not Anonymization !

OUR NEXT CHALLENGE:

PRIVACY BY DESIGN

- Developpers should write the masking rules
- It’s hard…. PostgreSQL must help them.
- The Postgres community has won so many battles
- Now we have to focus on data privacy

WE'RE HIRING !

Dalibo is a french-speaking employee-owned remote-working company

We’re looking for:

- PostgreSQL Development DBAs
- PostgreSQL Production DBAs
- Python Backend Developer
- Key Account Manager

GRAZIE !

- Contact : damien.clochard@dalibo.com
- Follow : @daamien
- Feedback : https://2019.pgconf.eu/f
- Other Projects : Dalibo Labs

true