



# Практическое применение средств защиты СУБД

Андрей Гусаков

20.01.2026

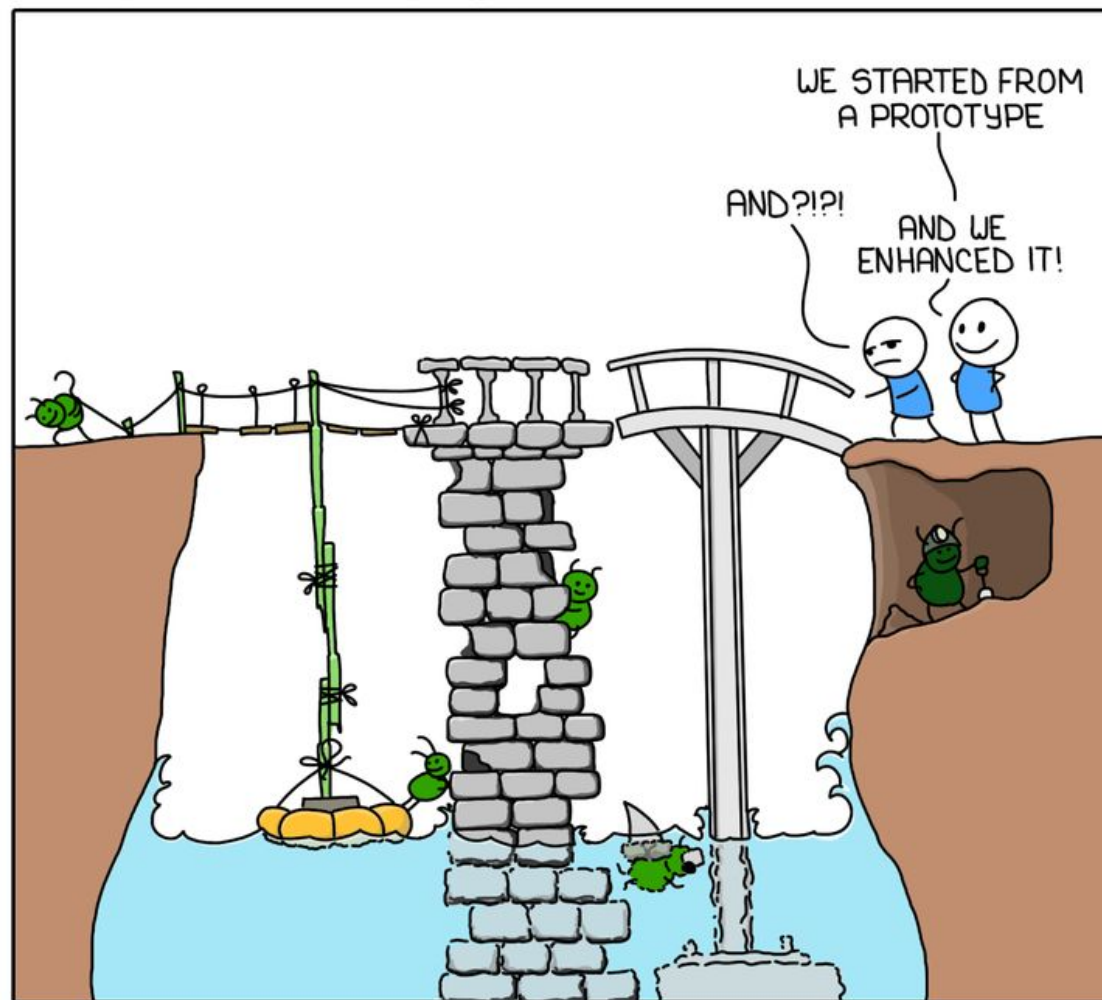
# Проблема – не используются доступные средства защиты СУБД

Помимо проблем разделения труда и ответственности между отделами ИТ и ИБ существуют проблема юридического пробела в законодательных документах ФСТЭК

Имеются требования к разработчикам СУБД по наличию средств защиты, но отсутствуют требования к разработчикам приложений по обязательному их применению

В итоге задачи ИБ игнорируются или вместо промышленных решений применяются сделанные «на коленке»

PRODUCTION READY



# Проблема – недостаточно базовых средств защиты PostgreSQL

## Имеются требования регуляторов:

- К распределению обязанностей (проблема суперпользователя – бесконтрольность, невозможность отследить опасные действия, риски использования superuser для работы сервисов)
- К сложности пароля и таймаутам
- К очистке памяти
- К стабильности настроек СУБД
- К перечню и содержанию регистрируемых событий безопасности
- Реестры, безопасная разработка, сертификация, исправления



# Проблема – недостаточно базовых средств защиты PostgreSQL

## Имеются требования рынка:

- Ограничение доступа к чувствительным данным привилегированных пользователей
- Замена разработок или OpenSource утилит для маскирования данных
- Повышение гибкости аудита и снижение нагрузки от него на СУБД, упрощение интеграции с SIEM-системами
- Нужна утилита для поиска чувствительной информации в СУБД
- Нужна синхронизация ролей и привилегий СУБД с группами LDAP
- Неудобная криптография – непрозрачно для приложений, утилит, имеются уязвимости
- Временное использование старого и нового пароля при его смене
- Соккрытие чувствительной информации в файлах аудита
- Обфускация кода разрабатываемых функций

# Компания Postgres Professional успешно дополнила OpenSource-разработки своими решениями

10 лет

на рынке  
с 2015

ТОП-5

в мире по вкладу  
в PostgreSQL  
(100+ патчей  
ежегодно)

№1

в России среди  
разработчиков  
СУБД,  
по данным  
ЦСР (2025)



500+

специалистов  
в команде,  
включая Major  
Contributors  
PostgreSQL



3000+ заказчиков

крупнейшие  
госкомпании,  
банки, телеком,  
критическая  
инфраструктура



## СИСТЕМА СЕРТИФИКАЦИИ СРЕДСТВ ЗАЩИТЫ ИНФОРМАЦИИ



### СЕРТИФИКАТ СООТВЕТСТВИЯ № 7

Выдан: 21 октября 2025 г.  
Действителен до: 21 октября 2030 г.

Настоящий сертификат удостоверяет, что процессы безопасной разработки, реализованные обществом с ограниченной ответственностью «Постгрес Профессиональный» (ООО «Постгрес Профессиональный»), соответствуют требованиям национального стандарта ГОСТ Р 56939-2024 «Защита информации. Разработка безопасного программного обеспечения. Общие требования», утвержденного и введенного в действие приказом Федерального агентства по техническому регулированию и метрологии от 24 октября 2024 г. № 1504-ст.

Сертификат выдан на основании результатов сертификации, проведенной органом по сертификации федерального государственного бюджетного учреждения науки Институт системного программирования им. В.П. Иванникова Российской академии наук (аттестат аккредитации от 24.05.2024 № СЗИ RU.0001.01БИ00.А009) - экспертное заключение от 26.09.2025.

ПЕРВЫЙ ЗАМЕСТИТЕЛЬ ДИРЕКТОРА ФСТЭК РОССИИ



В.Лютиков

# Иногда средства обеспечения ИБ появляются вследствие разработки других решений

## «Побочка»

**Pgbouncer → Proxima**  
(управление пулом соединений)

---

«Нативная» аутентификация без ведения отдельного списка пользователей с их паролями

**pgpro\_ilm + pgpro\_usage**

---

Статистика отношений и функций в разрезе пользователей  
→ возможность отъема неиспользуемых привилегий

**PREM**

---

Дополнительно к рабочему месту администратора СУБД запланировано рабочее место администратора ИБ



Пример построения защиты  
тестового приложения  
на уровне СУБД



# Что и какими инструментами мы будем защищать

Задача – воспользоваться средствами ИБ, встроенными в СУБД и прозрачно\* защитить ценную информацию

\* т.е. не меняя что-то в приложении и у клиентов)



Демонстрационная база данных «Авиаперевозки»

# 1.1 Разведка чувствительных данных – словарь

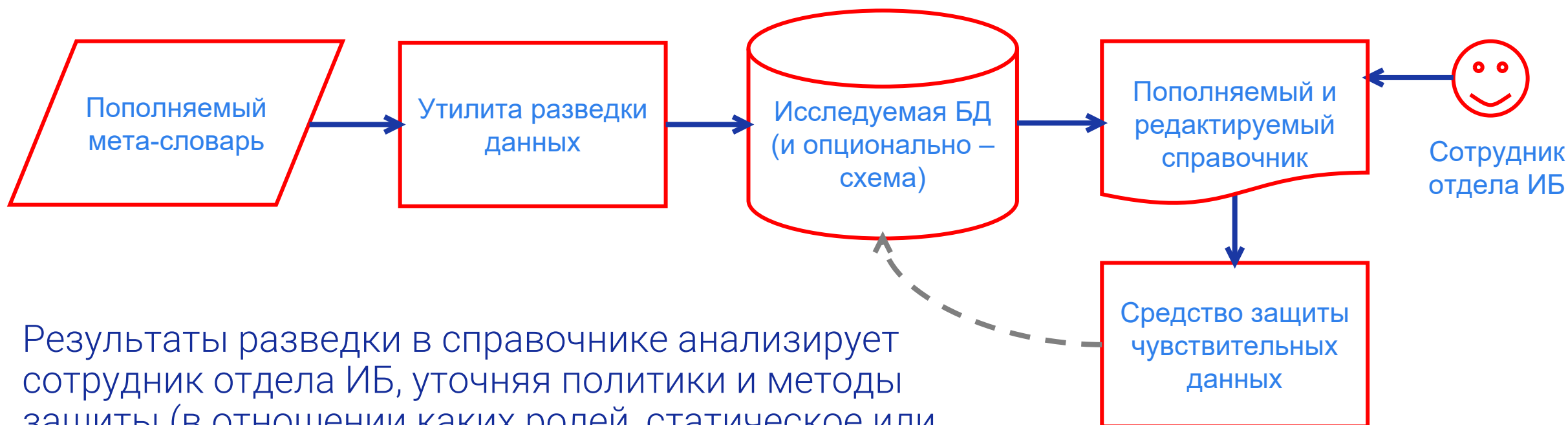
## Используем утилиту pgpro\_scout

Для разведки используется мета-словарь, содержащий маски для проверки имен и содержимого полей (в т.ч. JSON) по набору регулярных выражений и константным значениям (названия организаций, фамилии и т. д.)

```
scout:
  column_names:
    matchers:
      - key: passport
        match_values: [passport,
passports]
      - key: namecolumn
        pattern: (name|contact)
  column_values:
    case_sensitive: false
    matchers:
      - key: russian_surnames
        case_sensitive: false # the
option is optional, might be deleted
        pattern: (OVA|EVA|NOV|MOV|KOV)$
      - key: emailfield
        case_sensitive: false
        pattern: (email)
      - key: email_pattern
        case_sensitive: false
        pattern: ([A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.(ru|com))
      - key: phonefield
        case_sensitive: false
        pattern: (phone)
  exclude:
    schemas:
      - pgpro_sfile_data
      - dbms_lob
      - profile
      - information_schema
      - pg_catalog
      - pg_toast
    column_types:
      - dbms_lob.blob
      - dbms_lob.clob
      - sfile
```

## 1.2 Разведка чувствительных данных – утилита

```
./bin/pgpro_scout inspect -f scout.yml -d demo -h localhost -p 5432 -U postgres  
--search-path=bookings -o report.csv
```



Результаты разведки в справочнике анализирует сотрудник отдела ИБ, уточняя политики и методы защиты (в отношении каких ролей, статическое или динамическое маскирование, TDE и т.п.)

# 1.3 Разведка чувствительных данных – отчет

#report time: 2025-09-23T07:05:12-04:00					
matcher	location	column	key	comment	sample
column_name	bookings.airports_data	airport_name	namecolumn		airport_name
column_name	bookings.tickets	contact_data	namecolumn		contact_data
column_name	bookings.tickets	passenger_name	namecolumn		passenger_name
column_value	bookings.tickets	contact_data	phonefield		{"phone": "+70127117011"}
column_value	bookings.tickets	passenger_name	russian_surnames		VALERIY TIKHONOV
column_value	bookings.tickets	contact_data	phonefield		{"phone": "+70378089255"}
column_value	bookings.tickets	passenger_name	russian_surnames		EVGENIYA ALEKSEEVA
column_value	bookings.tickets	contact_data	phonefield		{"phone": "+70760429203"}
column_value	bookings.tickets	passenger_name	russian_surnames		ARTUR GERASIMOV
column_value	bookings.tickets	contact_data	emailfield		{"email": "volkova.alina_03101973@postgrespro.ru", "phone": "+70582584031"}
column_value	bookings.tickets	contact_data	email_pattern		{"email": "volkova.alina_03101973@postgrespro.ru", "phone": "+70582584031"}
column_value	bookings.tickets	contact_data	phonefield		{"email": "volkova.alina_03101973@postgrespro.ru", "phone": "+70582584031"}
column_value	bookings.tickets	passenger_name	russian_surnames		ALINA VOLKOVA
column_value	bookings.tickets	contact_data	emailfield		{"email": "m-zhukov061972@postgrespro.ru", "phone": "+70149562185"}
column_value	bookings.tickets	contact_data	email_pattern		{"email": "m-zhukov061972@postgrespro.ru", "phone": "+70149562185"}
column_value	bookings.tickets	contact_data	phonefield		{"email": "m-zhukov061972@postgrespro.ru", "phone": "+70149562185"}
column_value	bookings.tickets	passenger_name	russian_surnames		MAKSIM ZHUKOV
column_value	bookings.tickets	contact_data	emailfield		{"email": "kuznecova-t-011961@postgrespro.ru", "phone": "+70400736223"}
column_value	bookings.tickets	contact_data	email_pattern		{"email": "kuznecova-t-011961@postgrespro.ru", "phone": "+70400736223"}
column_value	bookings.tickets	contact_data	emailfield		{"email": "antonova.irina04121972@postgrespro.ru", "phone": "+70844502960"}
column_value	bookings.tickets	contact_data	email_pattern		{"email": "antonova.irina04121972@postgrespro.ru", "phone": "+70844502960"}
column_value	bookings.tickets	contact_data	emailfield		{"email": "kuznecova.valentina10101976@postgrespro.ru", "phone": "+70268080457"}
column_value	bookings.tickets	contact_data	email_pattern		{"email": "kuznecova.valentina10101976@postgrespro.ru", "phone": "+70268080457"}

## 1.4 Выбор средств защиты по отчету pgpro\_scout

Поскольку в таблице «tickets» обнаружены ПДн, к ней необходимо применить:

- **Маскирование**
  - Динамическое при запросах от недоверенных пользователей
  - Статическое при экспорте БД в недоверенную среду
- **Защитное преобразование** для данных в состоянии покоя (TDE)
  - На диске
  - В бэкапах

Кроме того, необходимо:

- Защитить все данные в схеме «bookings» **от привилегированных пользователей**
- **Проаудировать события ИБ** в БД «demo»
- Делегировать операции, обычно выполняемые суперпользователем СУБД, другим администраторам

## 2.1 Динамическое маскирование – задача

### Используем расширение pgpro\_anonymizer

Динамическое маскирование **изменяет представление реальных данных**, не модифицируя их. Некоторые пользователи могут читать только замаскированные данные, а другие могут получить доступ к исходной версии

Защитим от недоверенного пользователя имена пассажиров и контактные данные

```
ubuntu@ubuntu2204: ~  
postgres@demo=# CREATE ROLE untrusted_user LOGIN;  
CREATE ROLE  
postgres@demo=# GRANT USAGE ON SCHEMA bookings TO untrusted_user;  
GRANT  
postgres@demo=# GRANT SELECT ON ALL TABLES IN SCHEMA bookings TO untrusted_user;  
GRANT  
postgres@demo=# SECURITY LABEL FOR anon ON ROLE untrusted_user IS 'MASKED';  
SECURITY LABEL  
postgres@demo=# █
```

## 2.2 Динамическое маскирование – настройка правил преобразования

Воспользуемся одной из встроенных функций расширения и создадим кастомную функцию для данных в формате JSON

```
ubuntu@ubuntu2204: ~  
postgres@demo=# SECURITY LABEL FOR anon ON COLUMN tickets.passenger name IS 'MASKED WITH FUNCTION anon.fake first name()';  
SECURITY LABEL  
postgres@demo=# CREATE FUNCTION anon.fake_contact_data(src jsonb) RETURNS jsonb RETURN jsonb_build_object('email', anon.fake_email()  
, 'phone', anon.random_phone());  
CREATE FUNCTION  
postgres@demo=# SECURITY LABEL FOR anon ON COLUMN tickets.contact_data IS 'MASKED WITH FUNCTION anon.fake_contact_data(contact_data)  
';  
SECURITY LABEL  
postgres@demo=# ALTER DATABASE demo SET anon.sourceschema TO 'bookings';  
ALTER DATABASE  
postgres@demo=# \c  
You are now connected to database "demo" as user "postgres".  
postgres@demo=# SELECT anon.start_dynamic_masking();
```

Укажем схему, подлежащую маскированию, и опционально схему, в которой будут создаваться замаскированные представления

После переподключения можно запускать анонимизацию

## 2.3 Динамическое маскирование – результат

В ответе недоверенному пользователю заменены имена и контактные данные, остальная информация сохранена в исходном виде

```
ubuntu@ubuntu2204: ~  
postgres@demo=#  
postgres@demo=# SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM tickets LIMIT 5;  
 ticket_no | book_ref | passenger_id | passenger_name | contact_data  
-----+-----+-----+-----+-----  
 0005432000987 | 06B046 | 8149 604011 | VALERIY TIKHONOV | {"phone": "+70127117011"}  
 0005432000988 | 06B046 | 8499 420203 | EVGENIYA ALEKSEEVA | {"phone": "+70378089255"}  
 0005432000989 | E170C3 | 1011 752484 | ARTUR GERASIMOV | {"phone": "+70760429203"}  
 0005432000990 | E170C3 | 4849 400049 | ALINA VOLKOVA | {"email": "volkova.alina_03101973@postgrespro.ru", "phone": "+70582584031"}  
 0005432000991 | F313DD | 6615 976589 | MAKSIM ZHUKOV | {"email": "m-zhukov061972@postgrespro.ru", "phone": "+70149562185"}  
(5 rows)  
  
postgres@demo=# \c - untrusted user  
You are now connected to database "demo" as user "untrusted_user".  
untrusted_user@demo=> SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM tickets LIMIT 5;  
 ticket_no | book_ref | passenger_id | passenger_name | contact_data  
-----+-----+-----+-----+-----  
 0005432000987 | 06B046 | 8149 604011 | Dean | {"email": "hkelly@example.com", "phone": "+79224688356"}  
 0005432000988 | 06B046 | 8499 420203 | Sabrina | {"email": "susan61@example.org", "phone": "+79402616457"}  
 0005432000989 | E170C3 | 1011 752484 | Joe | {"email": "ksimpson@example.org", "phone": "+79599986028"}  
 0005432000990 | E170C3 | 4849 400049 | Valerie | {"email": "pflores@example.com", "phone": "+79328197064"}  
 0005432000991 | F313DD | 6615 976589 | Michelle | {"email": "njohnson@example.net", "phone": "+79782881778"}  
(5 rows)
```

## 3.1 Защита данных в состоянии покоя – задача

### Используем расширение pgpro\_tde

С помощью обратимого «математического преобразования» при записи информации БД на диск или ленту делать ее нечитаемой

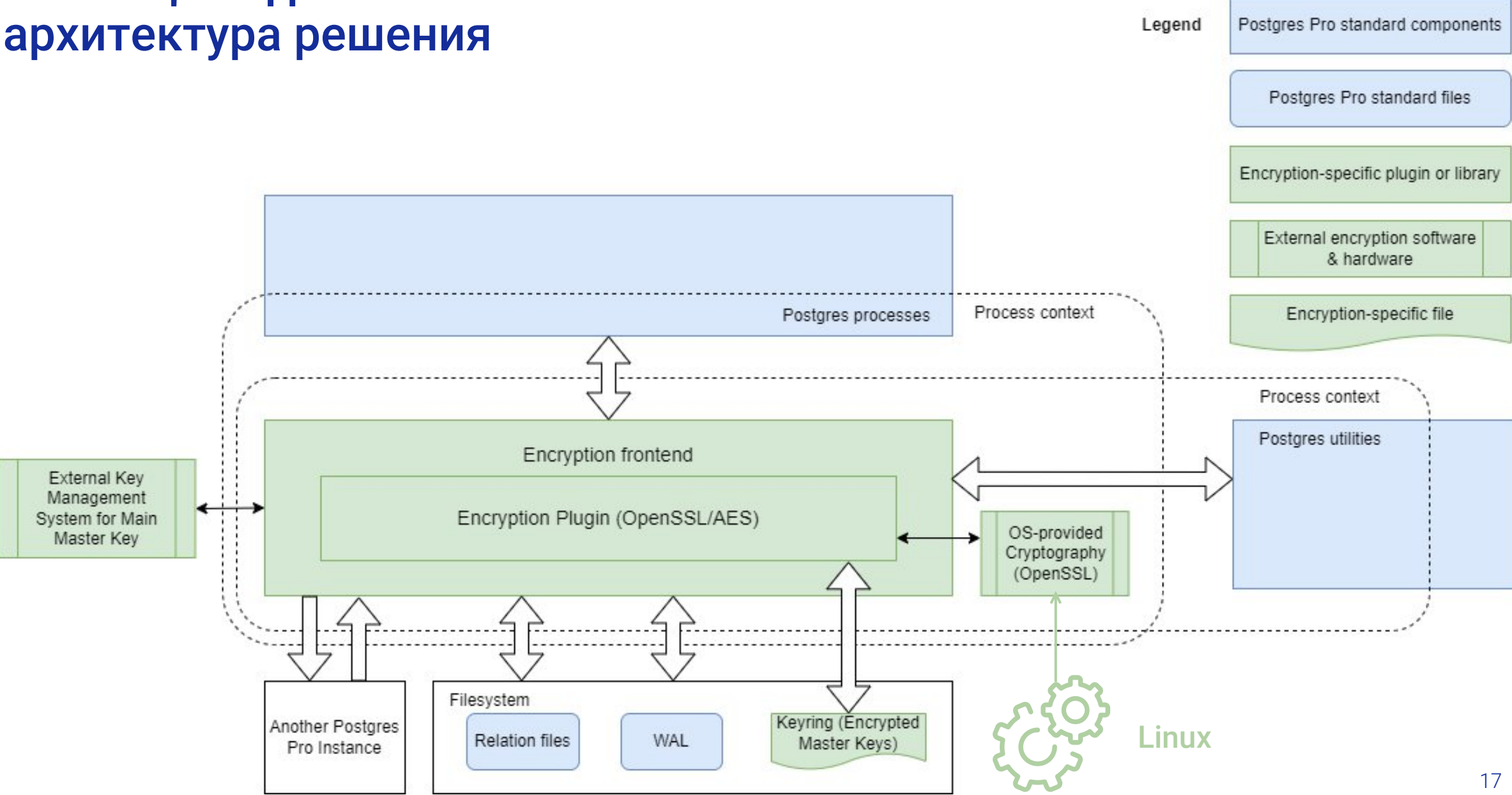
По возможности минимизировать дополнительную нагрузку на СУБД

Обеспечить прозрачное чтение защищенных данных клиентами и утилитами СУБД

Использовать ротацию секретов

Обеспечить работы БД в многоузловой конфигурации

# 3.2 Защита данных в состоянии покоя – архитектура решения



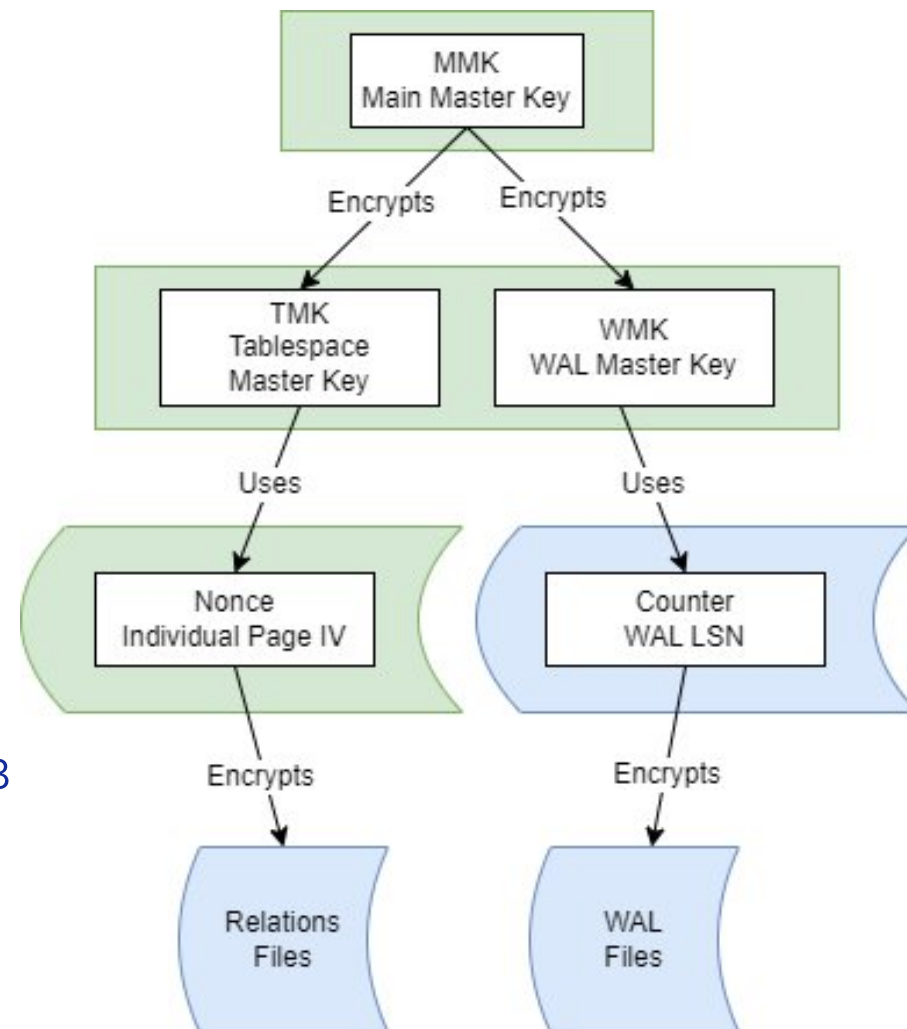
## 3.3 Защита данных в состоянии покоя – использование ключей

СУБД оперирует двумя типами ключей из защищенного хранилища секретов, и ей при старте необходимо сообщить алгоритм преобразования

Простейший способ – использование OpenSSL и пароля из файла (безопасней – из извлекаемой флешки), но возможно обращение к внешнему хранилищу секретов

Ключи табличного пространства модифицируются для вариативности, и метаданная об использованном ключе и модификатор заносится в сопроводительный файл рядом с защищаемым

Возможна ручная генерация (ротация) ts-ключей



## 3.4 Защита данных в состоянии покоя – активация TDE

В «postgresql.conf» вносятся параметры:

- encryption = on
- encryption\_key\_wrap\_command = 'openssl enc -pass file:/var/lib/pgpro/ent-17/secrets/passphrase.txt -aes-256-cbc -pbkdf2 -a >%p'
- encryption\_key\_unwrap\_command = 'openssl enc -pass file:/var/lib/pgpro/ent-17/secrets/passphrase.txt -aes-256-cbc -in %p -pbkdf2 -a -d'
- (в продуктиве лучше использовать алгоритм AES-GCM)

Защищаемые отношения переносятся в выделенное табличное пространство

- CREATE TABLESPACE encrypted LOCATION '/var/lib/pgpro/ent-17/meta/pg\_encryption';
- ALTER TABLESPACE encrypted SET (encryption=on);



## 3.5 Защита данных в состоянии покоя – перемещение таблицы в новое табличное пространство

На уровне СУБД данные не изменились

```
ubuntu@ubuntu2204: ~  
postgres@demo=# ALTER TABLE bookings.tickets SET TABLESPACE encrypted;  
ALTER TABLE  
postgres@demo=# SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM tickets LIMIT 3;  
 ticket_no | book_ref | passenger_id | passenger_name | contact_data  
-----+-----+-----+-----+-----  
 0005432000987 | 06B046 | 8149 604011 | VALERIY TIKHONOV | {"phone": "+70127117011"}  
 0005432000988 | 06B046 | 8499 420203 | EVGENIYA ALEKSEEVA | {"phone": "+70378089255"}  
 0005432000989 | E170C3 | 1011 752484 | ARTUR GERASIMOV | {"phone": "+70760429203"}  
(3 rows)  
  
postgres@demo=# SELECT pg_relation_filepath('bookings.tickets');  
 pg_relation_filepath  
-----  
 pg_tblspc/52266/PG_17_202407061/34507/52274  
(1 row)
```

Определим расположение таблицы «tickets» после переноса для того, чтобы проверить содержимое файла на уровне ОС

## 3.6 Защита данных в состоянии покоя – результат

```
ubuntu@ubuntu2204: ~  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/meta$ cd /var/lib/pgpro/ent-17/meta/pg_encryption/PG_17_202407061/34507  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/meta/pg_encryption/PG_17_202407061/34507$ ls  
52274 52274_fsm 52274-tde 52274_vm 52275 52275-tde 52276 52276-tde  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/meta/pg_encryption/PG_17_202407061/34507$ hexdump -C 52274 | grep phone  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/meta/pg_encryption/PG_17_202407061/34507$ hexdump -C 52274 | tail -5  
0303ffc0 2d 14 00 bb 29 dd c3 f3 26 89 14 11 2b b3 0e d5 |-...)...&...+...|  
0303ffd0 7d 37 1a 8a ef 02 d1 f0 15 12 08 31 ce 4b 2c a0 |}7.....1.K,..|  
0303ffe0 69 96 50 c0 46 d8 85 5a ec eb a0 9d 64 55 3e c4 |i.P.F..Z....dU>.|  
0303fff0 d6 89 fc f8 82 82 95 a2 c8 3a 78 b6 bd 55 f7 81 |.....:x..U..|  
03040000  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/meta/pg_encryption/PG_17_202407061/34507$ hexdump -C 52274 | head -5  
00000000 00 00 00 00 90 ee d3 3a 0e ce 04 00 04 01 20 01 |.....:..... .|  
00000010 e8 1f fe 20 80 01 f7 e8 1e 06 12 ab 23 bd d0 6a |... ..#..j|  
00000020 d8 7d b7 6b bc 53 04 92 dc 87 89 32 73 8e e6 3c |.}.k.S.....2s..<|  
00000030 ef 37 7f 35 c1 ac 27 08 90 7d 74 24 ed 97 74 66 |.7.5..'..}t$..tf|  
00000040 03 39 4f 2d 68 45 c8 fc e0 45 aa 09 8f f9 28 0c |.90-hE...E....(.|
```

Поиск в файле не работает, данные нечитаемы

## 3.7 Защита данных в резервных копиях

```
ubuntu@ubuntu2204: ~  
ubuntu@ubuntu2204:~$ sudo su - postgres  
postgres@ubuntu2204:~$ pg_probackup backup -b FULL -B /var/probackup/ --instance ent-17  
NOTICE: Backup start, pg_probackup version: 2.8.10, instance: ent-17, backup ID: T707C1, backup mode: FULL, wal mode: ARCHIVE, remote: false, compress-algorithm: none, compress-level: 1  
INFO: Database backup start  
INFO: Enable CFS flock feature  
INFO: wait for pg_backup_start()  
INFO: Wait for WAL segment /var/probackup/wal/ent-17/00000001000000000000000DF to be archived  
INFO: PGDATA size: 334MB  
INFO: Current Start LSN: 0/DF000070, TLI: 1  
INFO: Start transferring data files  
INFO: Data files are transferred, time elapsed: 3s  
INFO: wait for pg_stop_backup()  
INFO: pg_stop_backup() successfully executed  
INFO: Wait for LSN 0/E00000C0 in archived WAL segment /var/probackup/wal/ent-17/00000001000000000000000E0  
INFO: Getting the Recovery Time from WAL  
INFO: Syncing backup files to disk  
INFO: Backup files are synced, time elapsed: 3s  
INFO: Validating backup T707C1  
INFO: validate_tablespace_map_content start  
INFO: Backup T707C1 data files are valid  
INFO: Backup T707C1 resident size: 334MB  
INFO: Backup T707C1 completed  
postgres@ubuntu2204:~$
```

## 3.8 Защита данных в резервных копиях – результат

```

ubuntu@ubuntu2204: ~
postgres@ubuntu2204: /var/probackup/backups/ent-17/T707C1/database/pg_tblspc/52266/PG_17_202407061/34507$ tree
.
├── 76921
├── 76921-tde
├── 76925
└── 76925-tde

1 directory, 4 files
postgres@ubuntu2204: /var/probackup/backups/ent-17/T707C1/database/pg_tblspc/52266/PG_17_202407061/34507$ hexdump -C 76921 | tail -5
03038070 e1 08 bc 35 87 cc 3f 63 c2 37 1d 20 6e e6 c7 2c |...5..?c.7. n.,|
03038080 e9 d2 1d 08 61 ac f7 87 2a ac 0a b4 1d 65 5f 78 |....a...*....e_x|
03038090 e7 ae 7d 1c 34 e2 74 66 d0 fd 9f e6 fb 05 2a c7 |..}.4.tf.....*.|
030380a0 91 f0 7e b8 6f 02 45 30 19 7c ba 89 fd 35 af 38 |..~.o.E0.|...5.8|
030380b0

postgres@ubuntu2204: /var/probackup/backups/ent-17/T707C1/database/pg_tblspc/52266/PG_17_202407061/34507$ hexdump -C 76925 | tail -5
00001fd0 4b 5f 74 7f 49 00 50 f7 0b e7 4b ef f9 2a 22 4f |K_t.I.P...K..*"O|
00001fe0 fb 35 41 96 2b cd 6f 1f b2 25 1a 0c 1d 12 c5 56 |.5A.+..o..%.....V|
00001ff0 e0 c8 e2 3b a2 dc c9 5b 7e 81 90 6e 8e 8c d3 ea |...;...[~..n....|
00002000 3d 44 8c 44 aa b6 27 3d | =D.D..' =|
00002008

postgres@ubuntu2204: /var/probackup/backups/ent-17/T707C1/database/pg_tblspc/52266/PG_17_202407061/34507$

```

Данные нечитаемы

## 3.9 Защита ключей в резервных копиях

Не забываем про ключницу, если вы перенесли ее из \$PGDATA

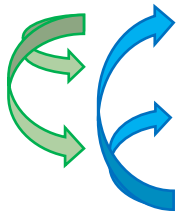
```
ubuntu@ubuntu2204: ~  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/data$ cd pg_encryption/  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/data/pg_encryption$ ls  
keys keys.old  
postgres@ubuntu2204:/var/lib/pgpro/ent-17/data/pg_encryption$ cd /var/probackup/backups/ent-17/T707C1/database/  
postgres@ubuntu2204:/var/probackup/backups/ent-17/T707C1/database$ tree | grep keys  
| └─ keys  
|   └─ keys.old  
postgres@ubuntu2204:/var/probackup/backups/ent-17/T707C1/database$
```

Также надо бэкапировать файл с passphrase, если вы его использовали (но отдельно от БД!)

## 4.1 Разграничение обязанностей

### Ограничение доступа администратора СУБД к данным

Роли администратора СУБД и администратора БД



позволят освободить суперюзера от рутинных операций

```
ubuntu@ubuntu2204: ~
postgres@demo=# \du+
List of roles
Role name | Attributes | Description
-----|-----|-----
backup | Replication |
bookings_owner | Create role |
dbms_admin | Create role, Create DB, Replication |
demo_admin | Create role |
pgpro_dbms_admin | Create role, Create DB, Cannot login, Replication |
pgpro_demo_admin | Create role, Cannot login |
postgres | Superuser, Create role, Create DB, Replication, Bypass RLS |
ppem |
ppem_agent | Superuser, Create role, Create DB, Replication |
sec_officer |
```

Другие администраторы не должны переключаться в роли владельца и администратора доступа защищенной схемы

При создании администраторов им выдаются права на набор predefined ролей и системных функций

## 4.2 Смена владельца у схемы и у всех ее объектов

```

ubuntu@ubuntu2204: ~
postgres@demo=# ALTER SCHEMA bookings OWNER TO bookings_owner;
ALTER SCHEMA
postgres@demo=# ALTER SCHEMA bookings SECURITY OFFICER TO sec_officer;
ALTER SCHEMA
postgres@demo=# GRANT CREATE ON TABLESPACE encrypted TO bookings_owner;
GRANT
postgres@demo=# \dp

```

Schema	Name	Type	Access privileges	Column privileges	Policies
bookings	aircrafts	view	bookings_owner=arwdDxtm/bookings_owner		
bookings	aircrafts_data	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	airports	view	bookings_owner=arwdDxtm/bookings_owner		
bookings	airports_data	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	boarding_passes	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	bookings	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	flights	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	flights_flight_id_seq	sequence	bookings_owner=rwU/bookings_owner		
bookings	flights_v	view	bookings_owner=arwdDxtm/bookings_owner		
bookings	pg_proaudit_log	table			
bookings	pg_proaudit_settings	view	postgres=arwdDxtm/postgres		
bookings	routes	view	bookings_owner=arwdDxtm/bookings_owner		
bookings	seats	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	ticket_flights	table	bookings_owner=arwdDxtm/bookings_owner		
bookings	tickets	table	bookings_owner=arwdDxtm/bookings_owner		

(15 rows)

## 4.3 Теперь только владелец схемы имеет доступ к ее данным

```

ubuntu@ubuntu2204: ~
postgres@ubuntu2204:~$ psql demo -U bookings_owner
psql (17.6)
Type "help" for help.

bookings_owner@demo=> SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM bookings.tickets LIMIT 3;
 ticket_no | book_ref | passenger_id | passenger_name | contact_data
-----+-----+-----+-----+-----
 0005432000987 | 06B046 | 8149 604011 | VALERIY TIKHONOV | {"phone": "+70127117011"}
 0005432000988 | 06B046 | 8499 420203 | EVGENIYA ALEKSEEVA | {"phone": "+70378089255"}
 0005432000989 | E170C3 | 1011 752484 | ARTUR GERASIMOV | {"phone": "+70760429203"}
(3 rows)

bookings_owner@demo=> \c - sec_officer
You are now connected to database "demo" as user "sec_officer".
sec_officer@demo=> SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM bookings.tickets LIMIT 3;
ERROR: permission denied for table tickets
sec_officer@demo=> \c - demo_admin
You are now connected to database "demo" as user "demo_admin".
demo_admin@demo=> SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM bookings.tickets LIMIT 3;
ERROR: permission denied for schema bookings
LINE 1: ..., passenger_id, passenger_name, contact_data FROM bookings.t...
                                     ^

demo_admin@demo=> \c - dbms_admin
You are now connected to database "demo" as user "dbms_admin".
dbms_admin@demo=> SELECT ticket_no, book_ref, passenger_id, passenger_name, contact_data FROM bookings.tickets LIMIT 3;
ERROR: permission denied for schema bookings
LINE 1: ..., passenger_id, passenger_name, contact_data FROM bookings.t...
                                     ^

dbms_admin@demo=>
dbms_admin@demo=> SET ROLE bookings_owner ;
ERROR: permission denied to set role "bookings_owner"
dbms_admin@demo=>

```

## 5.1 Аудит действий пользователей

### Используем расширение pg\_proaudit

- Оптимизированный механизм фильтрации событий – правило для аудирования определяется комбинацией параметров из:
  - Имени базы данных
  - Типа события (можно указать как конкретные команды, так и классы событий)
  - Типа объекта
  - Имени объекта
  - Имени роли (можно указать как конкретного пользователя, так и групповую роль)
- Можно логировать любую команду SQL (в т.ч. – какой пользователь выполнил действие) и указывать важность события
- Высокое быстродействие благодаря параллельной обработке
- Доработки – исключение временных таблиц, экспорт данных в Common Event Format для SIEM, маскирование паролей; в планах – отслеживание продолжительности сессии и объема передаваемых данных

## 5.2 Настройка аудита по примеру из документации

```
ubuntu@ubuntu2204: ~
postgres@demo=# SELECT * FROM pg_proaudit_settings;
```

db_name	event_type	object_type	object_name	role_name	comment
demo	authenticate	ALL			Any authentication in the current DB
demo	disconnect	ALL			Any disconnect from the current DB
demo	ALL	table			Any operations with any table in the current DB
demo	ALL	ALL		bookings_owner	Any operation by "bookings_owner" user in the current DB
demo	ALL	ALL	bookings.tickets		

(5 rows)

```
postgres@demo=# SELECT pg_proaudit_save();
pg_proaudit_save
```

```
ubuntu@ubuntu2204: ~
postgres@demo=# SELECT to_char(log_time, 'DD.MM.YY HH24:MI:SS') AS when, current_usr_name,
      session_pid, event_type, query_text, session_usr_name
FROM bookings.pg_proaudit_log
WHERE object_name = 'public.test_table';
```

when	current_usr_name	session_pid	event_type	query_text	session_usr_name
19.12.25 05:32:21	demo_admin	13706	DROP TABLE	DROP TABLE test_table;	demo_admin
19.12.25 05:32:31	demo_admin	13706	CREATE TABLE	CREATE TABLE test_table (id int, name text);	demo_admin
19.12.25 05:32:41	demo_admin	13706	INSERT	INSERT INTO test_table VALUES (1, 'first');	demo_admin
19.12.25 05:32:56	demo_admin	13706	SELECT	SELECT * FROM test_table;	demo_admin

(4 rows)

## 5.3 Аудируем разграничение прав в действии

Владелец защищенной схемы может создать бизнес-пользователя, но для предоставления ему доступа к базе нужен администратор БД (или СУБД)

```
ubuntu@ubuntu2204: ~  
bookings_owner@demo=> GRANT CONNECT ON DATABASE demo TO bookings_user;  
2025-12-19 07:41:57.177 EST [23376] WARNING: no privileges were granted for "demo"  
WARNING: no privileges were granted for "demo"  
GRANT
```

Владелец защищенной схемы не может предоставить пользователю доступа к объектам схемы, для этого нужно разрешение администратора доступа

```
ubuntu@ubuntu2204: ~  
demo_admin@demo=> \c - bookings_owner  
You are now connected to database "demo" as user "bookings_owner".  
bookings_owner@demo=> GRANT USAGE ON SCHEMA bookings TO bookings_user;  
2025-12-19 07:46:28.436 EST [23876] WARNING: no privileges were granted for "bookings"  
WARNING: no privileges were granted for "bookings"  
GRANT
```

## 5.4 Особенности записи событий (WARNING не считается)

Права на просмотр, добавление и сохранение новых правил аудита делегированы суперпользователем администратору доступа; его собственные действия аудируются

```
ubuntu@ubuntu2204: ~
sec_officer@demo=> SELECT to_char(log_time, 'DD.MM.YY HH24:MI:SS') AS when, current_usr_name,
    session_pid, event_type, query_text, session_usr_name
FROM bookings.pg_proaudit_log order by log_time DESC limit 13;
```

when	current_usr_name	session_pid	event_type	query_text	session_usr_name
19.12.25 07:51:11	sec_officer	24122	SELECT	SELECT to_char(log_time, 'DD.MM.YY HH24:MI:SS') AS when, current_usr_name,+ session_pid, event_type, query_text, session_usr_name + FROM bookings.pg_proaudit_log;	sec_officer
19.12.25 07:49:43	bookings_owner	23876	DISCONNECT		bookings_owner
19.12.25 07:49:43	sec_officer	24122	AUTHENTICATE		sec_officer
19.12.25 07:49:30	bookings_owner	23876	SELECT	SELECT * FROM bookings.flights LIMIT 3;	bookings_owner
19.12.25 07:46:28	bookings_owner	23876	GRANT	GRANT USAGE ON SCHEMA bookings TO bookings_user;	bookings_owner
19.12.25 07:46:20	demo_admin	23808	DISCONNECT		demo_admin
19.12.25 07:46:20	bookings_owner	23876	AUTHENTICATE		bookings_owner
19.12.25 07:45:24	bookings_owner	23376	DISCONNECT		bookings_owner
19.12.25 07:45:24	demo_admin	23808	AUTHENTICATE		demo_admin
19.12.25 07:40:23	bookings_owner	23376	GRANT	GRANT CONNECT ON DATABASE demo TO bookings_user;	bookings_owner
19.12.25 07:40:12	bookings_owner	23376	CREATE ROLE	CREATE USER bookings_user;	bookings_owner
19.12.25 07:40:02	bookings_owner	23376	AUTHENTICATE		bookings_owner
19.12.25 07:39:33	postgres	21400	DISCONNECT		postgres

(13 rows)

Ошибочно записаны события предоставления прав владельцем схемы

Не записаны события предоставления прав администратором БД и администратором доступа (присутствуют только аутентификации)

## 6.1 Статическое маскирование – задача

Для тестировщиков ПО нужно создать набор данных, скрыв персональную информацию пассажиров, исказив даты и суммы бронирований, а также привязку билетов к бронированиям, но сохранив при этом ссылочную целостность таблиц.

```
$ psql postgres -U dbms_admin
=>CREATE DATABASE masked_demo WITH TEMPLATE demo;
```

```
ubuntu@ubuntu2204: ~
bookings_owner@masked_demo=> SELECT book_ref, book_date, passenger_name, contact_data, total_amount
FROM bookings.bookings JOIN bookings.tickets USING (book_ref) ORDER BY book_ref LIMIT 5;
 book_ref |          book_date          | passenger_name |          contact_data          | total_amount
-----+-----+-----+-----+-----
 00000F | 2017-07-04 20:12:00-04 | ANNA ANTONOVA | {"email": "annaantonova-19021973@postgrespro.ru", "phone": "+70938049942"} | 265700.00
 000012 | 2017-07-14 02:02:00-04 | TAMARA ZAYCEVA | {"email": "tamarazayceva-1971@postgrespro.ru", "phone": "+70749401734"} | 37900.00
 000068 | 2017-08-15 07:27:00-04 | TATYANA PETROVA | {"email": "t_petrova1970@postgrespro.ru", "phone": "+70886117503"} | 18100.00
 000181 | 2017-08-10 06:28:00-04 | EVGENIYA KARPOVA | {"email": "karpovaevgeniya.1985@postgrespro.ru", "phone": "+70669289906"} | 131800.00
 000181 | 2017-08-10 06:28:00-04 | ALEKSANDR ZHUKOV | {"email": "aleksandrzhukov111972@postgrespro.ru", "phone": "+70411811316"} | 131800.00
(5 rows)

bookings_owner@masked_demo=> UPDATE bookings.tickets SET passenger_name = anon.concat(anon.fake first name(), ' ', anon.fake last name());
UPDATE 366733
bookings_owner@masked_demo=> UPDATE bookings.tickets SET contact_data = jsonb_build_object('email', anon.fake_email(), 'phone', anon.random_phone(
));
UPDATE 366733
```

## 6.2 Статическое маскирование – искажение данных

```
ubuntu@ubuntu2204: ~  
bookings_owner@masked_demo=> SELECT anon.shuffle_column('tickets', 'book_ref', 'ticket_no');  
  shuffle_column  
-----  
 t  
(1 row)  
  
bookings_owner@masked_demo=> SELECT anon.add_noise_on_datetime_column('bookings', 'book_date', '3 days 1 hour');  
  add_noise_on_datetime_column  
-----  
 t  
  
bookings_owner@masked_demo=> SELECT anon.add_noise_on_numeric_column('bookings', 'total_amount', '0.2');  
  add_noise_on_numeric_column  
-----  
 t  
(1 row)  
  
bookings_owner@masked_demo=> UPDATE bookings.bookings SET total_amount = round(total_amount, -2);  
UPDATE 262788
```

↑  
указание внешнего первичного ключа  
позволяет сохранить ссылочную целостность

## 6.3 Статическое маскирование – результат

```
ubuntu@ubuntu2204: ~
bookings_owner@masked_demo=> SELECT book_ref, book_date, passenger_name, contact_data, total_amount
FROM bookings JOIN bookings.tickets USING (book_ref) ORDER BY book_ref LIMIT 5;
```

**исходные данные**

book_ref	book_date	passenger_name	contact_data	total_amount
00000F	2017-07-04 20:12:00-04	ANNA ANTONOVA	{"email": "annaantonova-19021973@postgrespro.ru", "phone": "+70938049942"}	265700.00
000012	2017-07-14 02:02:00-04	TAMARA ZAYCEVA	{"email": "tamarazayceva-1971@postgrespro.ru", "phone": "+70749401734"}	37900.00
000068	2017-08-15 07:27:00-04	TATYANA PETROVA	{"email": "t_petrova1970@postgrespro.ru", "phone": "+70886117503"}	18100.00
000181	2017-08-10 06:28:00-04	EVGENIYA KARPOVA	{"email": "karpovaevgeniya.1985@postgrespro.ru", "phone": "+70669289906"}	131800.00
000181	2017-08-10 06:28:00-04	ALEKSANDR ZHUKOV	{"email": "aleksandrzhukov111972@postgrespro.ru", "phone": "+70411811316"}	131800.00

(5 rows)

```
ubuntu@ubuntu2204: ~
bookings_owner@masked_demo=> SELECT book_ref, book_date, passenger_name, contact_data, total_amount
FROM bookings JOIN tickets USING (book_ref) ORDER BY book_ref LIMIT 5;
```

**замаскированные данные**

book_ref	book_date	passenger_name	contact_data	total_amount
00000F	2017-07-06 15:22:21.290659-04	Alexa Rivers	{"email": "brewerjacob@example.com", "phone": "0381219314"}	219100.00
000012	2017-07-13 02:16:44.323021-04	Jordan Schaefer	{"email": "ygibson@example.net", "phone": "0165873553"}	37900.00
000068	2017-08-16 02:35:15.354986-04	Katelyn Ali	{"email": "jessica57@example.net", "phone": "0951861646"}	17900.00
000181	2017-08-09 15:39:12.977099-04	Edwin Floyd	{"email": "brittanystewart@example.net", "phone": "0767614987"}	149600.00
000181	2017-08-09 15:39:12.977099-04	Abigail Sosa	{"email": "hamiltonevelyn@example.com", "phone": "0473129802"}	149600.00

(5 rows)

В выборке из копии БД заменены имена и контактные данные, модифицированы даты рейсов и стоимость билетов



Результаты применения  
нескольких решений ИБ  
от Postgres Pro



## 7.1 Итоги – защита данных

Инвентаризация мест размещения чувствительных данных; выбор средств защиты

Динамическое маскирование при запросах от недоверенных пользователей

Статическое маскирование копий БД при экспорте в недоверенную среду

Ограниченный доступ к объектам выделенной схемы, даже со стороны привилегированных пользователей, даже на чтение

Защитное преобразование данных при записи на диск и в СРК; минимизация нагрузки за счет работы с выделенным табличным пространством; возможность использования внешних систем управления ключами

Соккрытие паролей в журналах аудита

## 7.2 Итоги – регистрация событий безопасности

Настройка требует суперпользователя; просмотр и добавление новых правил доступны администратору доступа

Аудируются действия всех администраторов и других пользователей

Высокоскоростная обработка аудируемых событий за счет параллельного запуска воркеров записи

События отправляются в SIEM в «нативном» CEF

## 7.3 Итоги – разграничение обязанностей

Настройка требует суперпользователя; но впоследствии он освобождается от большинства операций:

- Резервное копирование выполняется от его имени на уровне ОС (без логина в СУБД). Также права на управление репликацией и бэкапом выданы администратору экземпляра
- Недоверенные расширения устанавливаются с помощью администратора инфраструктуры
- Создание БД, табличных пространств, управление профилями и репликацией возможно силами администратора СУБД
- Выделенные доверенный администратор (владелец защищенной схемы) и администратор доступа не требуют 100% утилизации
- В исключениях – ALTER SYSTEM, ... , восстановление защищенной схемы из логической копии

## 8 Что изменится в типичных инфраструктурных задачах при усилении ИБ

Для конечных пользователей системы – ничего; но появляется возможность разделить их на доверенных и недоверенных

Для администраторов:

- Создание физических резервных копий и восстановление – без изменений
- Создание логических копий и восстановление – усложнение из-за защищенной схемы
- Создание маскированных копий БД – упрощение



**Спасибо за внимание!**

**Ждем вас на  
PGPro TechDay 2026!**

[presales@postgrespro.ru](mailto:presales@postgrespro.ru)

[pgproday.ru/pgprotechday](http://pgproday.ru/pgprotechday)

