Основы технологий баз данных

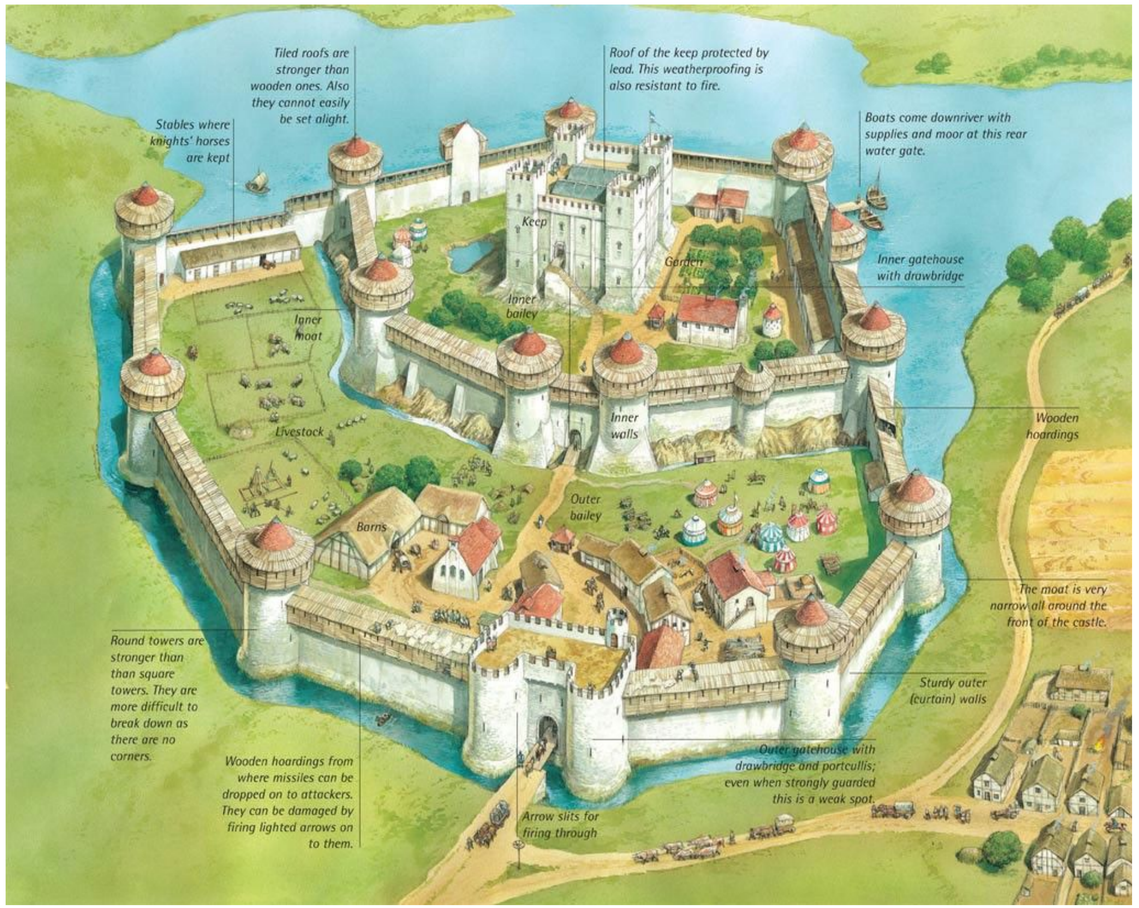
Б. А. Новиков
Санкт-Петербургский университет, JetBrains Research

Безопасность баз данных

PROFESSIONAL Postgres

Защита должна быть похожа на крепость

- Физическая защита оборудования
- Защита в вычислительной сети
- Операционная система
- Файловая система
- Сервер приложений и приложения
- База данных



Модели защиты данных

- Защита от несанкционированного доступа
- Разграничение доступа на основе ролей: RBAC
- Разграничение доступа по значениям: ABAC

Понятия, связанные с разграничением доступа

- Принципал
- Пользователь
- Роль

- Класс объектов, таблица
- Объект (экземпляр), строка таблицы

- Операция (функция)
- Привилегия (право)

Защита в PostgreSQL: сущности

- Роли
- Объекты: база данных, схема, таблица, представление, функция
- Операции: зависят от типа объекта (чтение, изменение, удаление, выполнение, …)
- Пользователи (в PostgreSQL мало отличаются от ролей)
- Суперпользователи

Пользователи и роли

- CREATE ROLE demo_reader;
- ALTER ROLE demo_reader LOGIN;
- DROP ROLE …

Владельцы и передача привилегий

- Пользователь, создавший объект (базу данных, схему, таблицу, представление, функцию и т. д.) становится владельцем этого объекта

- Владелец объекта может передавать и отзывать права доступа к объектам

```
GRANT SELECT on SCHEMA bookings TO demo_reader;

REVOKE SELECT ON . . . FROM demo_reader;

GRANT reader TO writer;
```

Функции

- Основная привилегия — EXECUTE
- Возможно выполнение с правами доступа к объектам той роли, которая создала функцию
  - Гибкое ограничение прав (например, обновление только через функцию)

Разграничение доступа на уровне строк

```
CREATE USER alice WITH LOGIN;
CREATE USER bob WITH LOGIN;

CREATE SCHEMA rowlevel;
CREATE TABLE rowlevel.notes (
  note_owner text NOT NULL,
  note_key   text NOT NULL,
  note_text  text NOT NULL,
  PRIMARY KEY (note_key)
);
```

```
ALTER TABLE rowlevel.notes
  ENABLE ROW LEVEL SECURITY;
CREATE POLICY per_user
  ON rowlevel.notes
  FOR ALL
  USING (note_owner IN (
    SELECT current_user))
  WITH CHECK (note_owner IN (
    SELECT current_user));
```

Разграничение доступа по значениям

```
demo=> SELECT * FROM rowlevel.notes;
 note_owner | note_key |      note_text
------------+----------+---------------------
 alice      | alice-1  | first note of Alice
 alice      | alice-2  | second note of Alice
 bob        | bob-1    | first note of Bob
 bob        | bob-2    | one more note of Bob
(4 rows)
```

```
demo=# \connect demo alice
You are now connected to database "demo" as user "alice".
demo=> SELECT * FROM rowlevel.notes;
 note_owner | note_key |      note_text
------------+----------+---------------------
 alice      | alice-1  | first note of Alice
 alice      | alice-2  | second note of Alice
(2 rows)

demo=> \connect demo bob
You are now connected to database "demo" as user "bob".
demo=> SELECT * FROM rowlevel.notes;
 note_owner | note_key |      note_text
------------+----------+---------------------
 bob        | bob-1    | first note of Bob
 bob        | bob-2    | one more note of Bob
(2 rows)
```

Шифрование

- Обеспечивает дополнительные уровни защиты
  - На уровне файловой системы
  - На уровне без данных
- Надежность определяется размерами и надежностью хранения ключей
- В принципе можно скрыть содержимое базы данных от администратора

Внешние серверы

- Сервер, на котором хранятся данные, может не заслуживать доверия
- Задачи:
  - Как индексировать зашифрованные данные?
  - Как выполнять (арифметические) операции над зашифрованными данными?
  - . . .

Разграничение доступа при большом количестве пользователей

- Невозможно зарегистрировать всех пользователей интернета как пользователей базы данных
- Реализация защиты на уровне приложения
- Необходимо применять разграничение на уровне строк
- Отдельные роли для анонимного пользователя, зарегистрированного в приложении, зарегистрированного в БД

Защита базы данных: итоги

- Средства безопасности должны быть многоуровневыми
- СУБД обеспечивают полноценную реализацию модели RBAC для объектов базы данных
- Применимость средств защиты на уровне базы данных ограничивается количеством пользователей, зарегистрированных на сервере базы данных
- Возможности доступа неограниченного количества пользователей реализуются на уровне приложений

Postgres PROFESSIONAL