Безопасность данных в FinTech-проекте

Руслан Каримов

Вот эти ребята

Зачем это все?

- Разработчики игнорируют требования
- Разработчики не осознают требования
- Разработчики думают, что это сложно

Про проект

- стартап в области финансов
- составляем инвестиционные портфолио
- упрощаем обработку документов
- интегрируем банки и аналитику

Данные

- SSN (номер социального страхования)
- ФИО, email, водительские права, etc.
- финансовое состояние
- пароли и токены к сторонним сервисам

Угроза 1: внутренняя

- Кража ноутбуков
- “Засланные” сотрудники
- Outsourcing

Решение: обфускация

- Gem DataAnonymization

```
require 'data-anonymization'

database 'DatabaseName' do
  strategy DataAnon::Strategy::Blacklist
  table 'User' do
    primary_key 'id' # composite key is also supported
    anonymize('first_name').using FieldStrategy::RandomFirstName.new
    anonymize('last_name').using FieldStrategy::RandomLastName.new
    anonymize('email').using FieldStrategy::RandomEmail.new('example','com')
    anonymize('Password') { |field| "password" }
  end
```

Плюсы

- Данные похожи на реальные
- Воспроизведение багов с production
- У разработчиков не утекут данные

Минусы

- Несовпадение данных при денормализованной схеме
- Процесс обновления требует настройки
- Скрипт обфускации нужно обновлять

Угроза 2: взлом сервера

Где хранить credentials?

- в файлах
- в переменных окружения
- в Vault

Пароли в файлах

- слишком очевидно

- уязвимо к “source code leakage”

Пароли в переменных окружения

- второе очевидное место
- часто попадают в crash dump и логи
- можно быстро их получить

Решение: Hashicorp Vault

A tool for managing secrets.

Пароли в Hashicorp Vault

- Зашифрованы
- Сперва нужно получить доступ к Vault
- Можно быстро заблокировать доступ
- Умеет генерировать временные ключи сам

Как использовать Vault

- консольный доступ: для записи значений
- API для чтения сервисом
- многоуровневый доступ
- разделяемые ключи

А данные?

- не храните ненужного
- храните хеши
- шифруйте

Средства шифрования

- Transparent Data Encryption
- Шифрование раздела
- Шифрование на уровне приложения

TDE

- Только Oracle и SQL server
- Защищает только data at rest
- Легко включается

Шифрование раздела

- Не требует изменений в приложении
- Защищает только data at rest

Шифрование в приложении

- Нужно писать код
- Нужно писать код внимательно!
- Требует меньше согласований
- Нет гарантий, но замедлит злоумышленника

Шифрование в приложении

- Pgcrypto для Postgres
- DoctrineEncryptBundle для Doctrine/Symfony
- Можно написать самим

Руслан
Каримов
rk@4xxi.com

Вопросы?