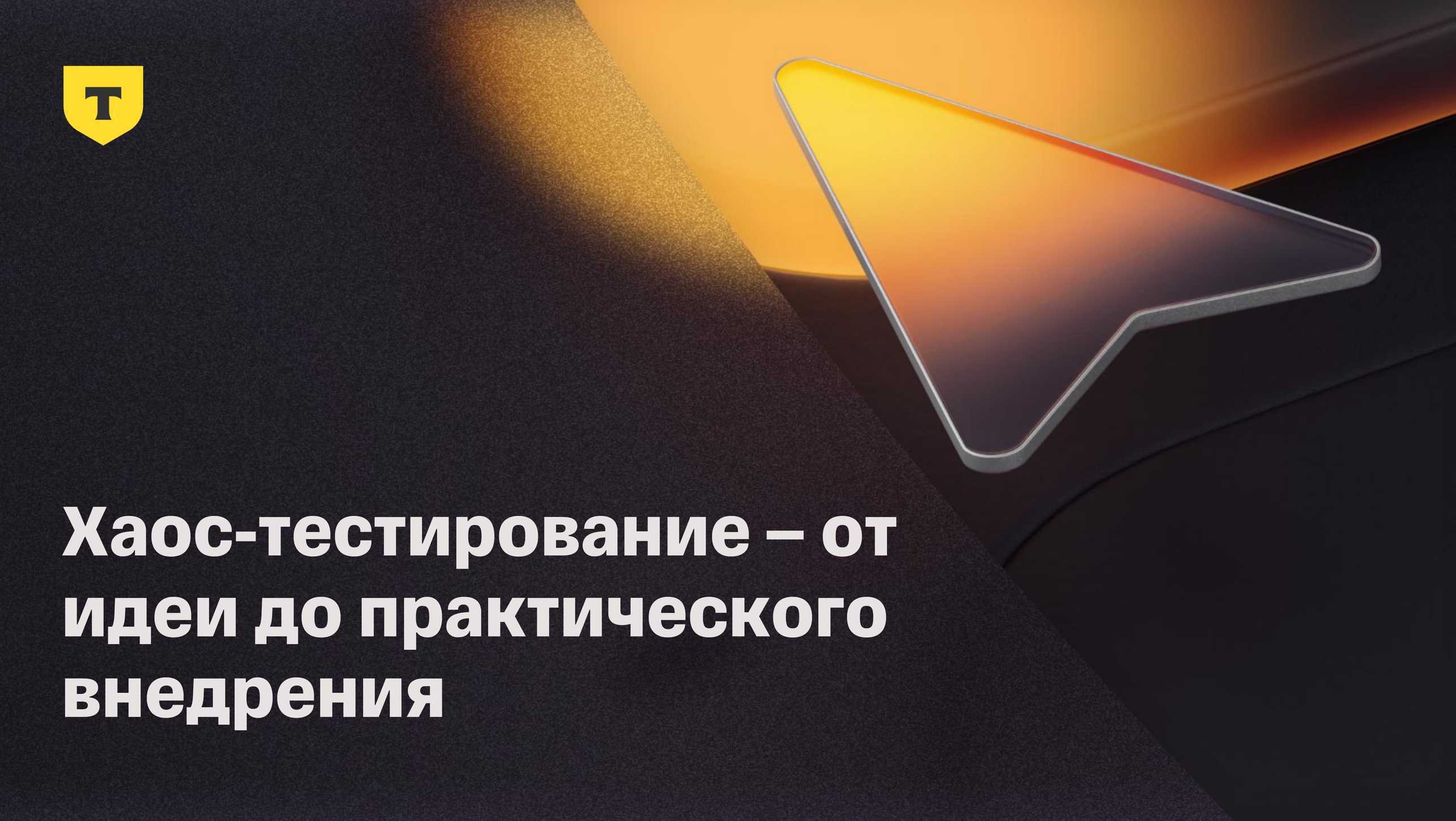


Хаос-тестирование – от идеи до практического внедрения

Фадеев Александр

- Ведущий инженер по обеспечению качества
- Нулевой пользователь платформы хаос-тестирования

Горбацевич Иван

- Ведущий инженер по надежности
- Разработчик платформы хаос-тестирования

О чём поговорим?

- Проблематика
- Примеры
- Концепция и разработка
- Результаты и рекомендации

Что заберёте с собой?

Понимание концепции хаос-тестирования

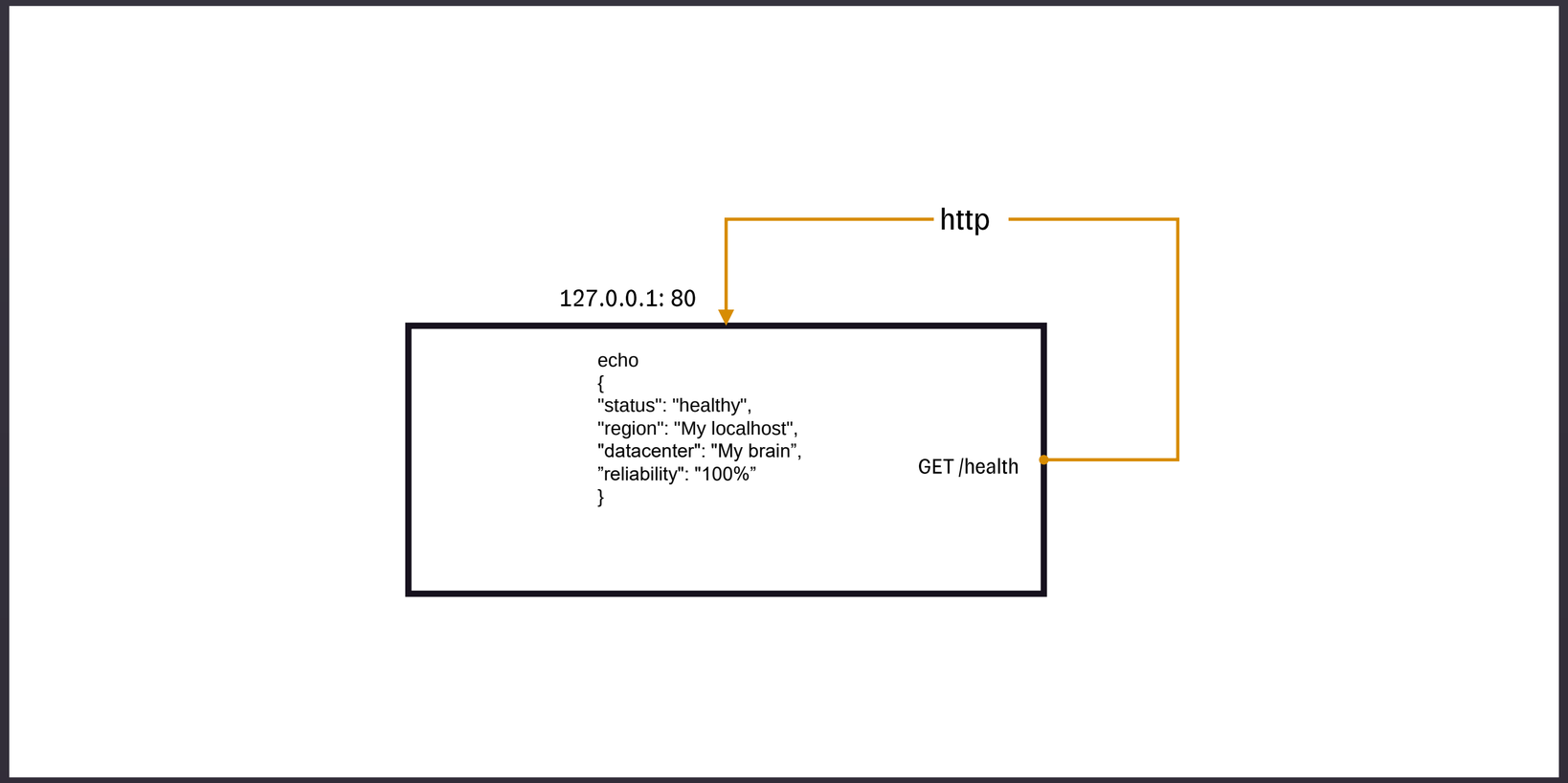
Поймёте, нужна ли вам практика

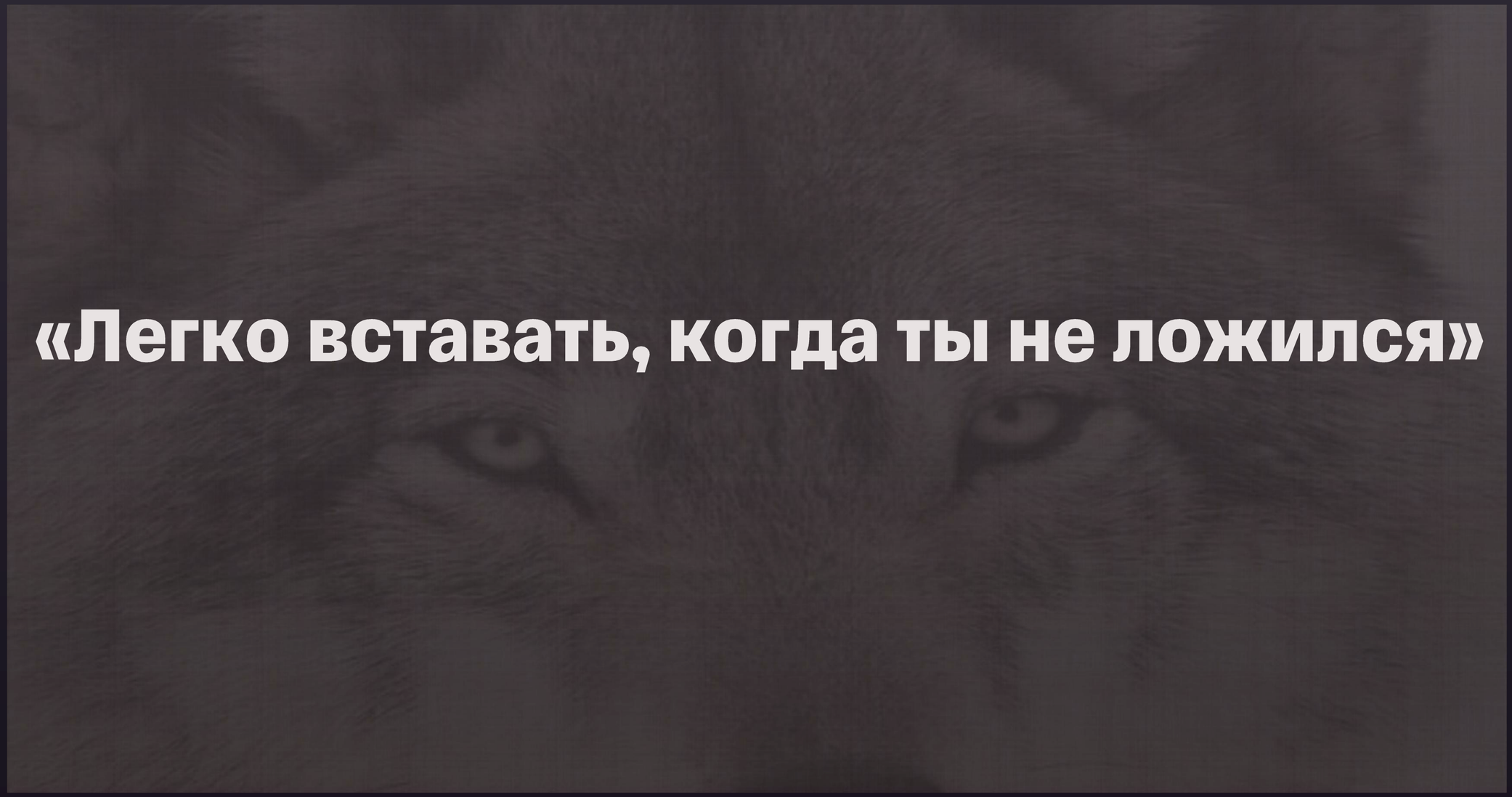
Проблематика

Готов ли ваш сервис к сбоям в инфраструктуре?

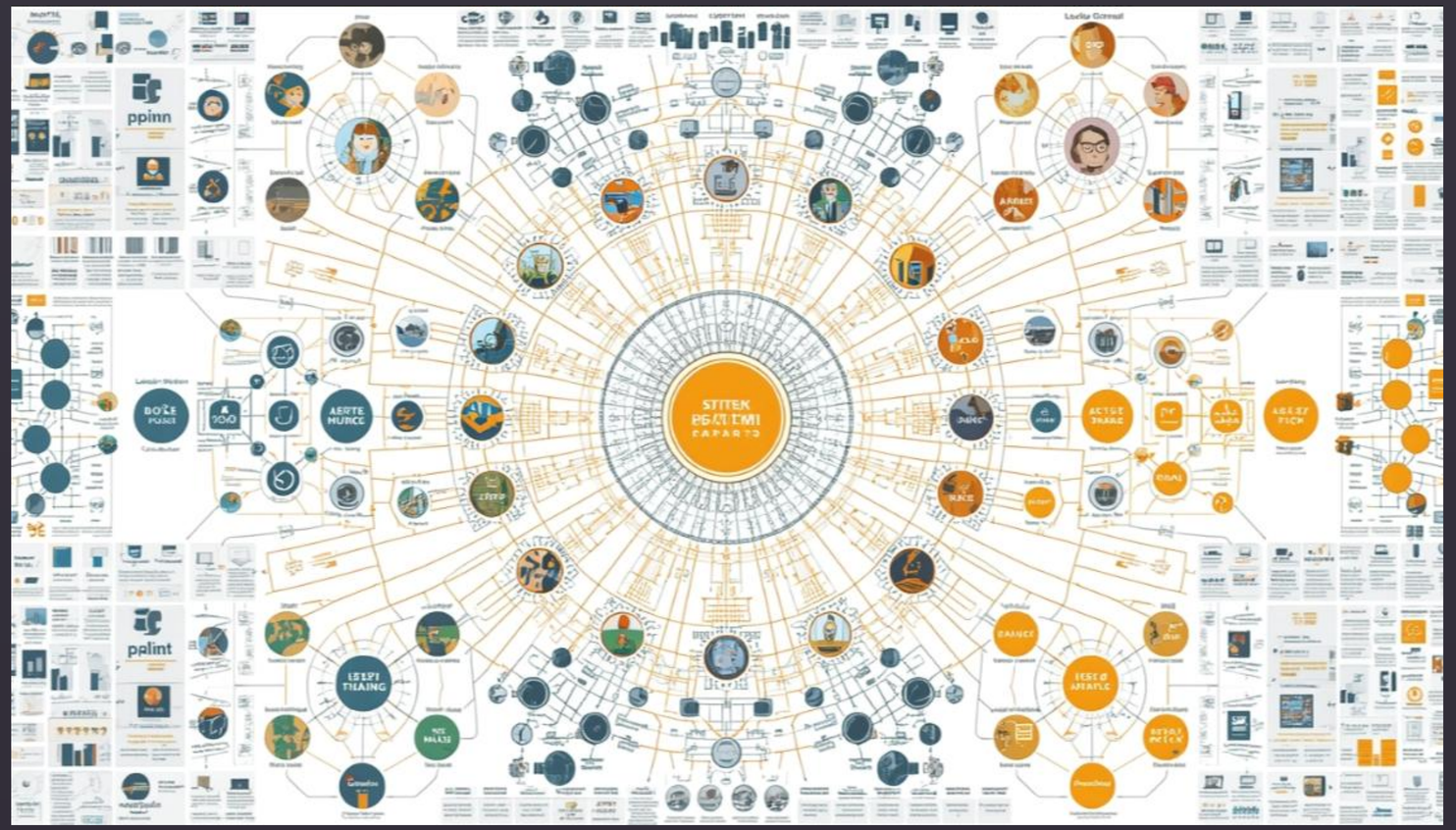
Да! Но это не точно

Можно ли предусмотреть все возможные негативные сценарии отказов?

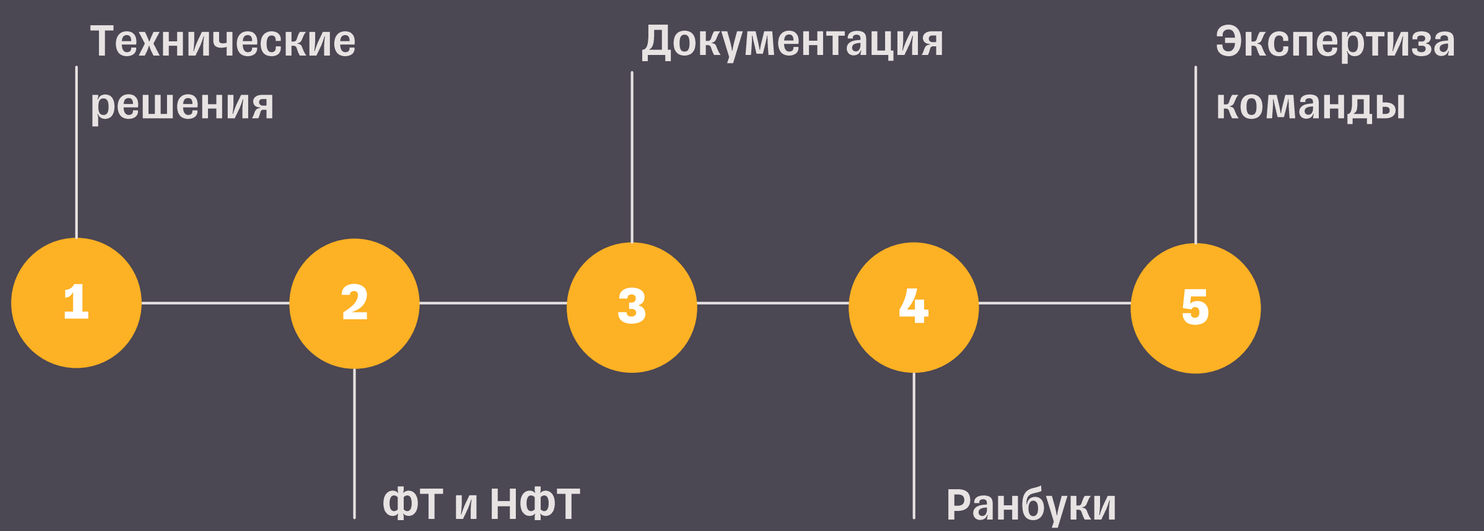



«Легко вставать, когда ты не ложился»

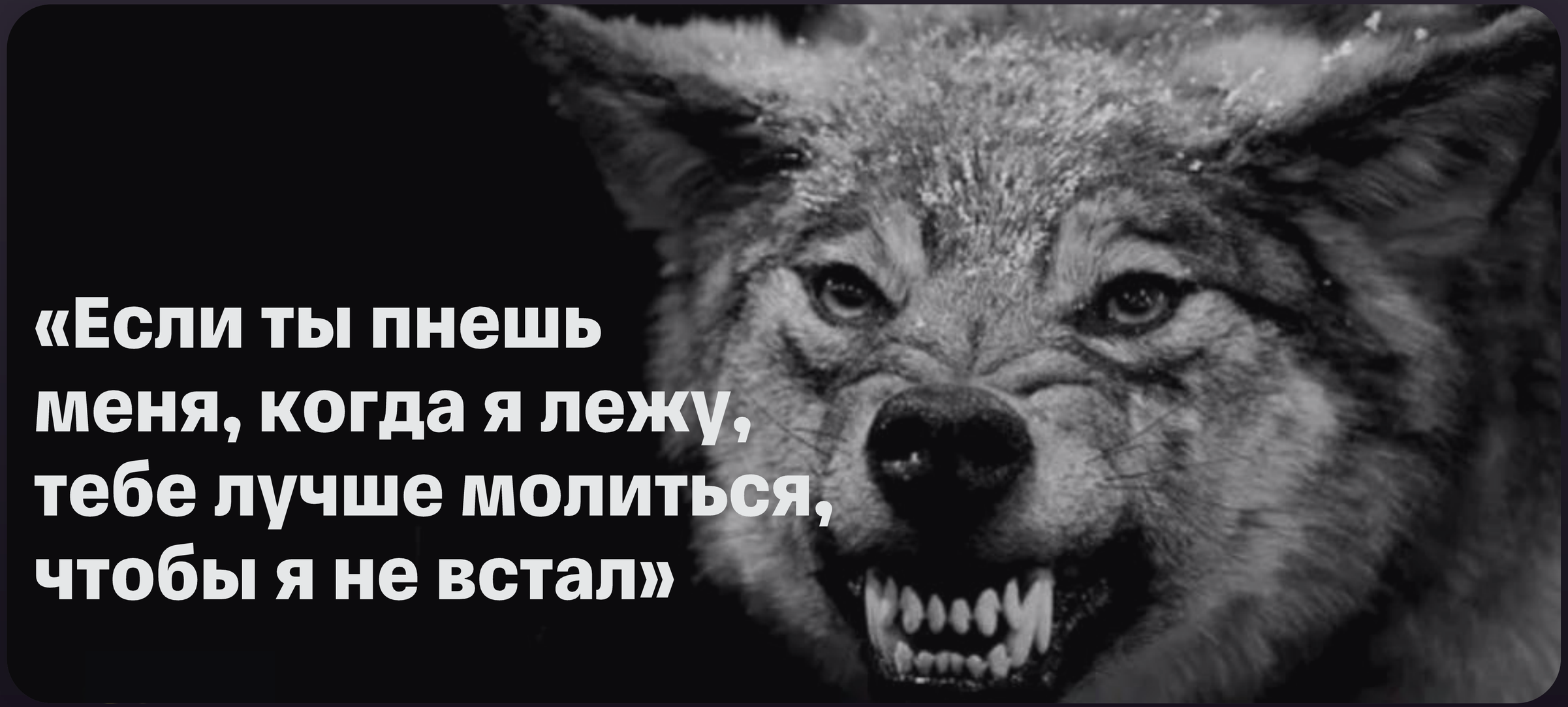
Можно ли предусмотреть все возможные негативные сценарии отказов?



Работаем над отказоустойчивостью



Сервисы все равно падают



«Сбои будут всегда! Важно на них учиться!»

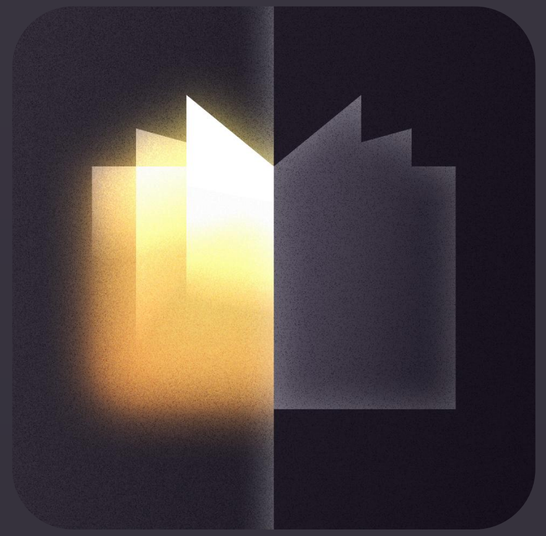
Некоторые проблемы можно обнаружить только через краш-тест

Концепция

Концепция

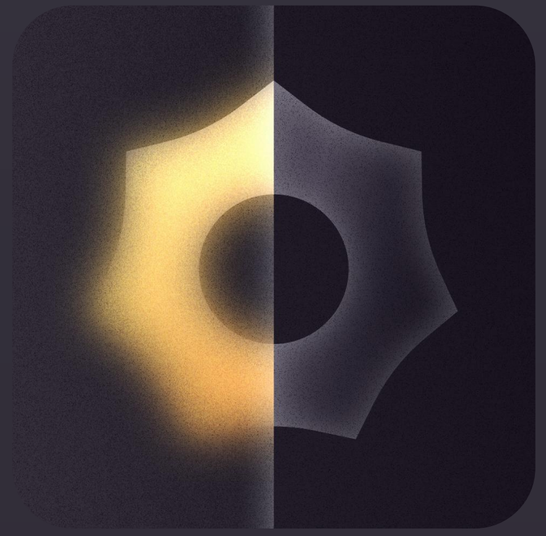
Определение

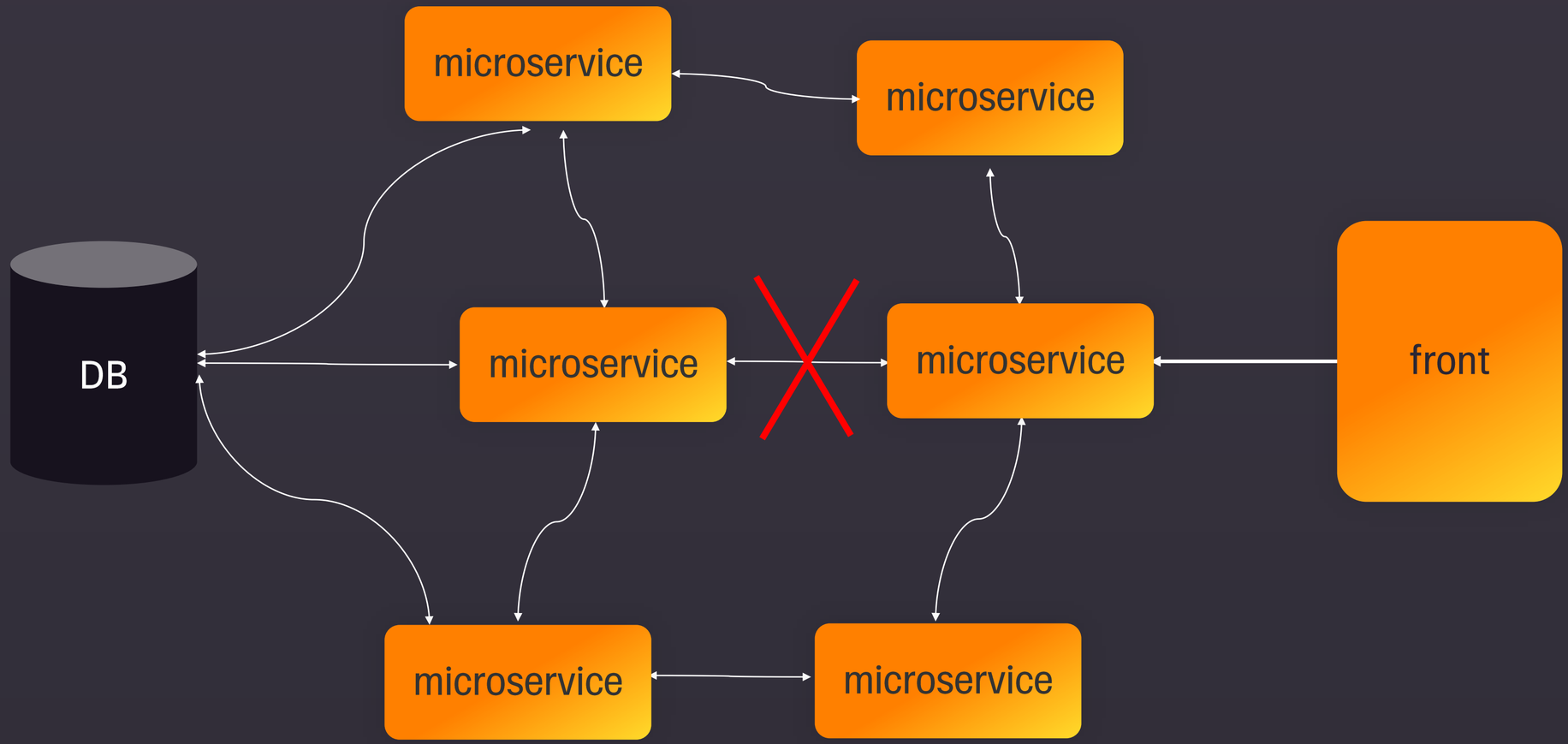
Практика, направленная на проверку отказоустойчивости системы путём намеренного внесения сбоев в её работу в контролируемых условиях.



Цель

Обнаружить слабые места в поведении системы до того, как они проявятся на продакшене при реальных сбоях.




microservice
microservice
DB
microservice
microservice
front
microservice
microservice

Как проводить хаос-тестирование?

01

Формулировка гипотезы

02

Моделирование сбоя

03

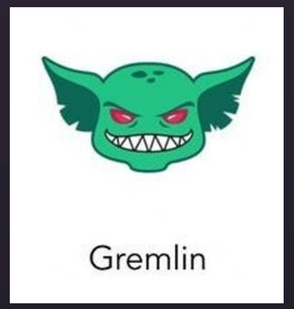
Проведение эксперимента

04

Анализ результатов

Разработка

Готовые решения

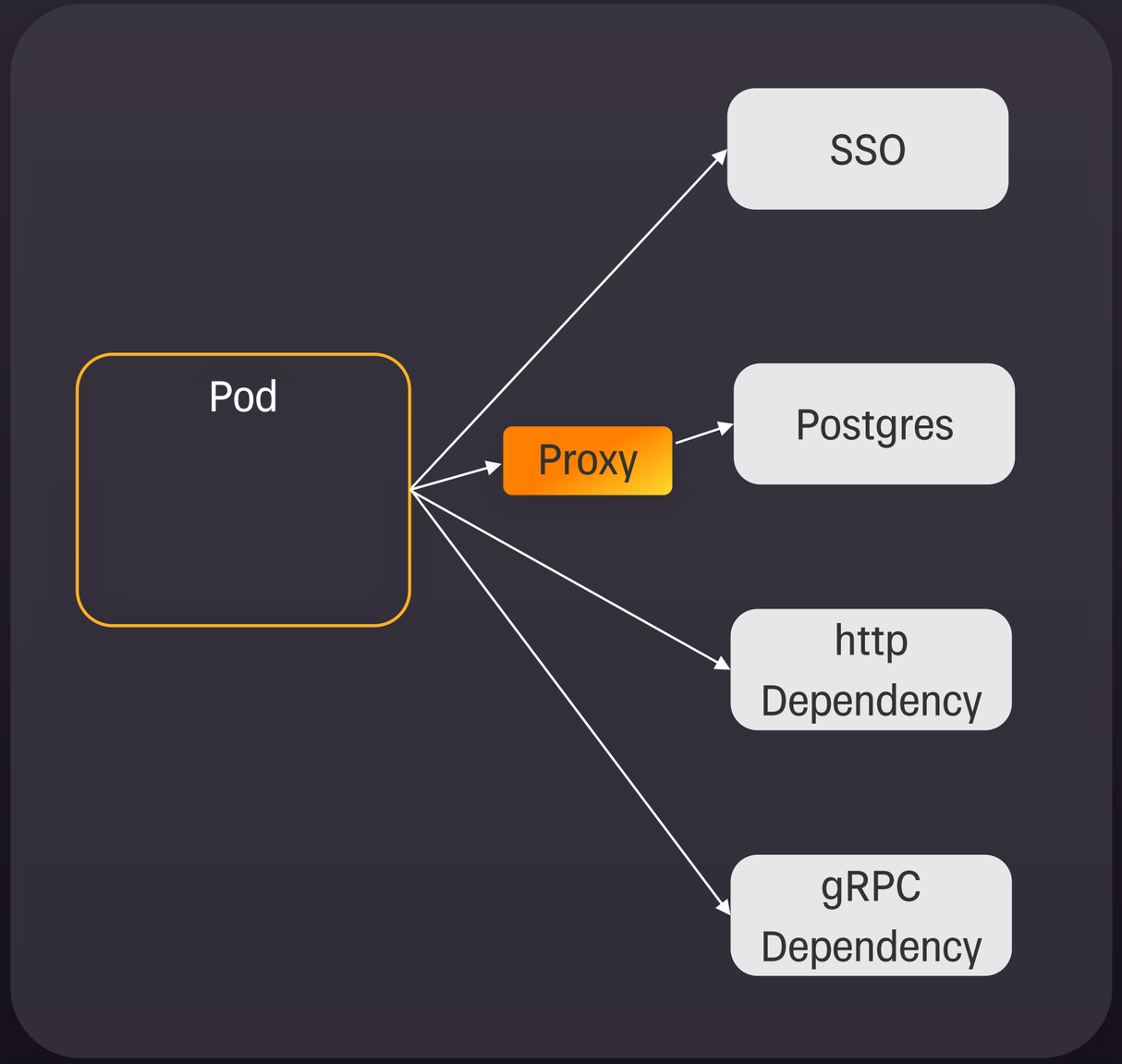


Предлагают большой выбор типов экспериментов - сеть, CPU, Java heap и т.д.

Инструменты почти всегда представляют из себя Kubernetes оператор и требуют широких полномочий внутри кластера

Основной фокус – сетевой хаос

Требования к инструменту

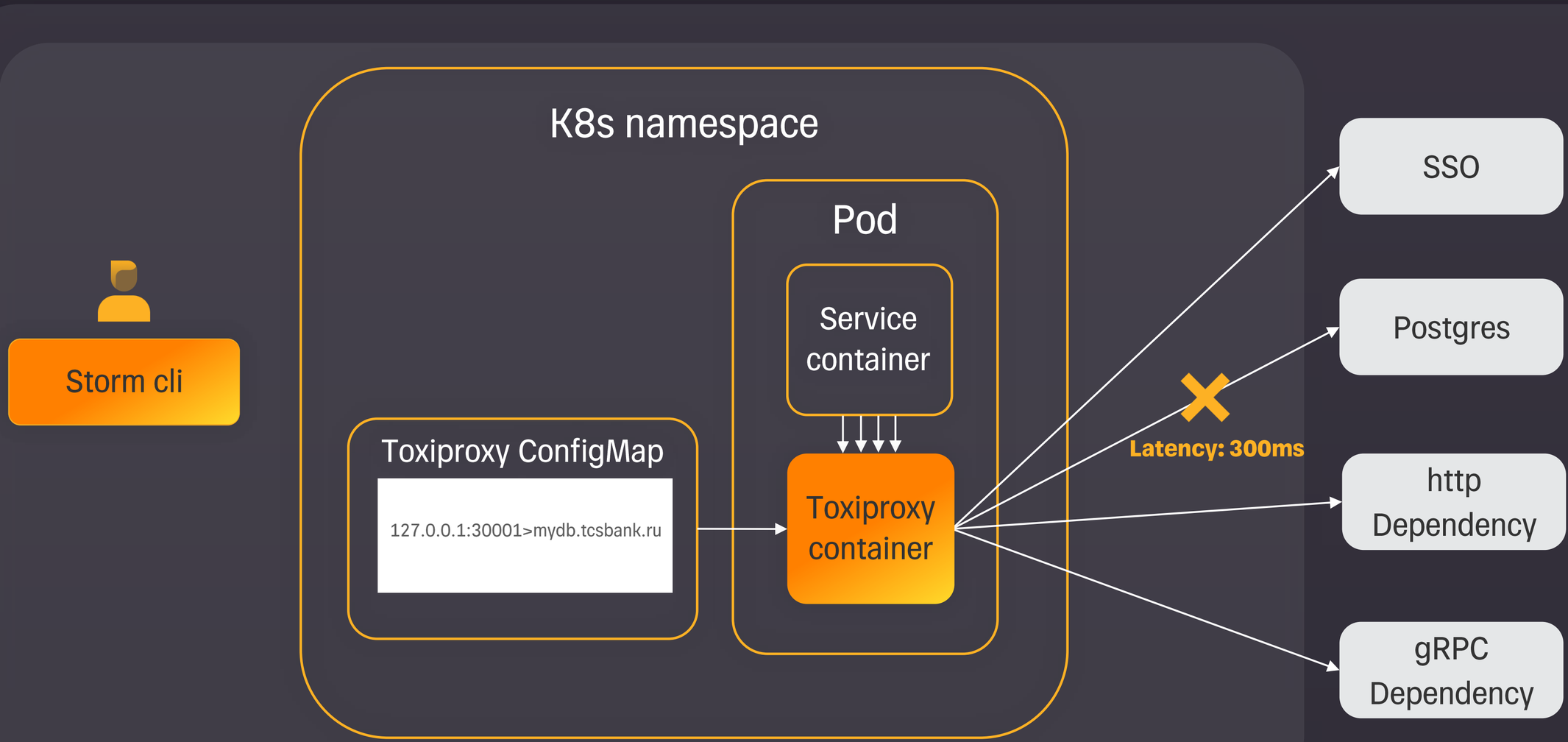


Какой тип экспериментов хотим получить?

Сетевое влияние на любые зависимости сервиса, имитирующие его работу в нестабильной среде

Собственная реализация Chaos Engineering

Добавляем sidecar прокси

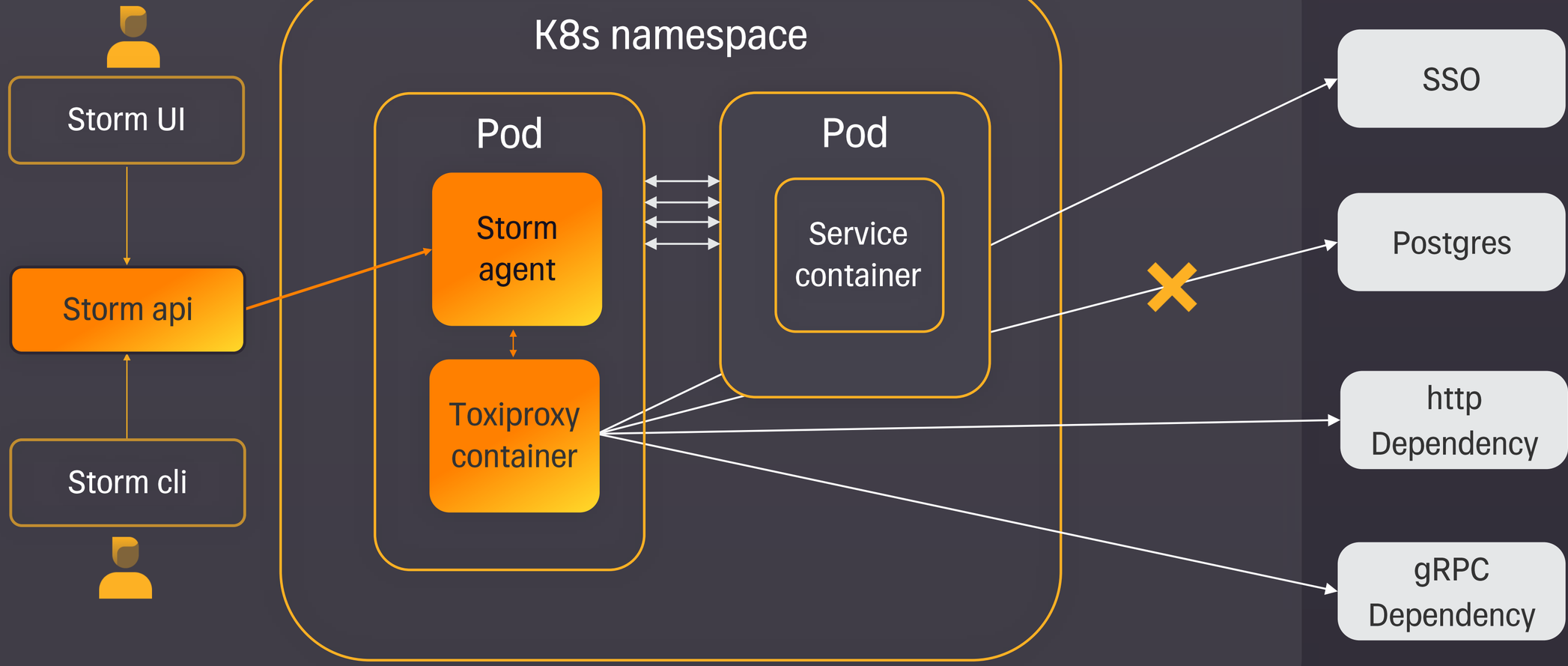


От MVP к продукту

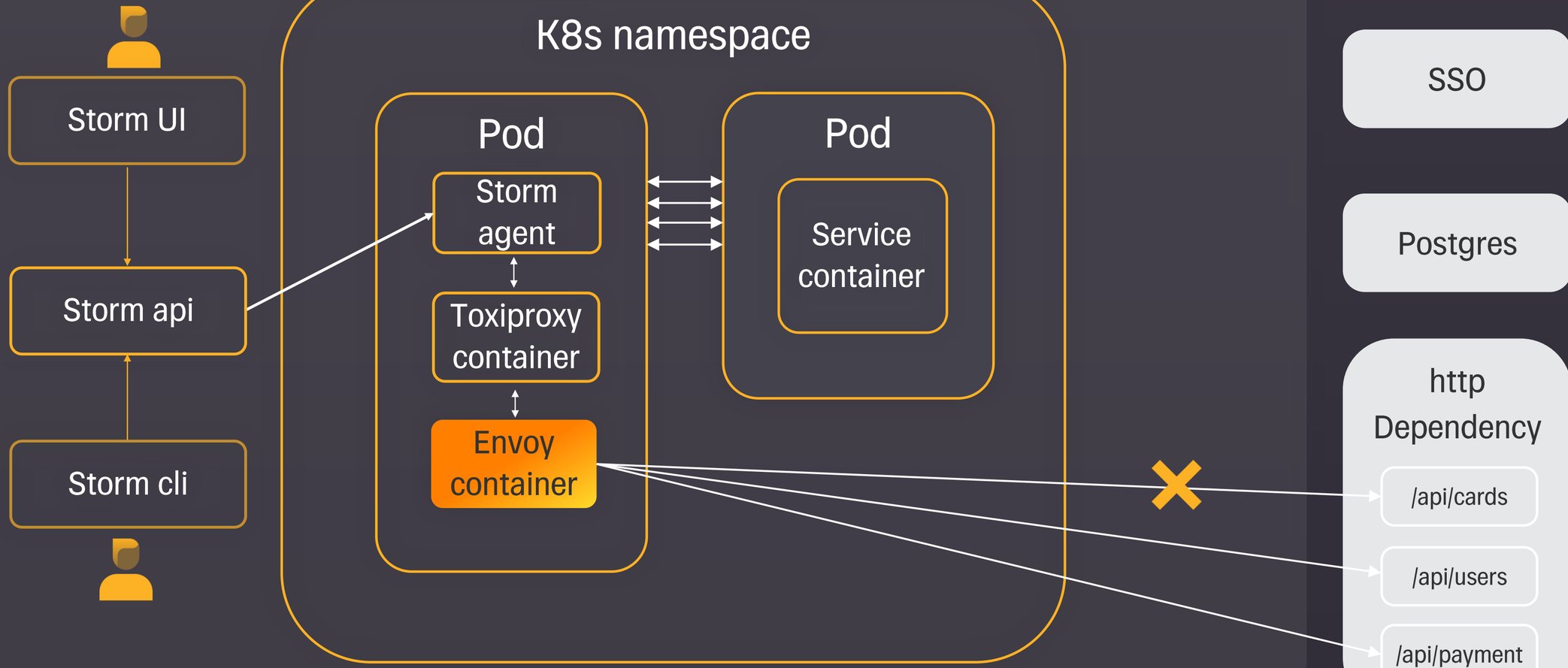
CLI на питоне - не продакшн решение

Обновление ConfigMap слишком долго обрабатывается в k8s

Добавляем STORM API



Добавляем Envoy прокси



Примеры

Выставление заявки

Выставление заявки

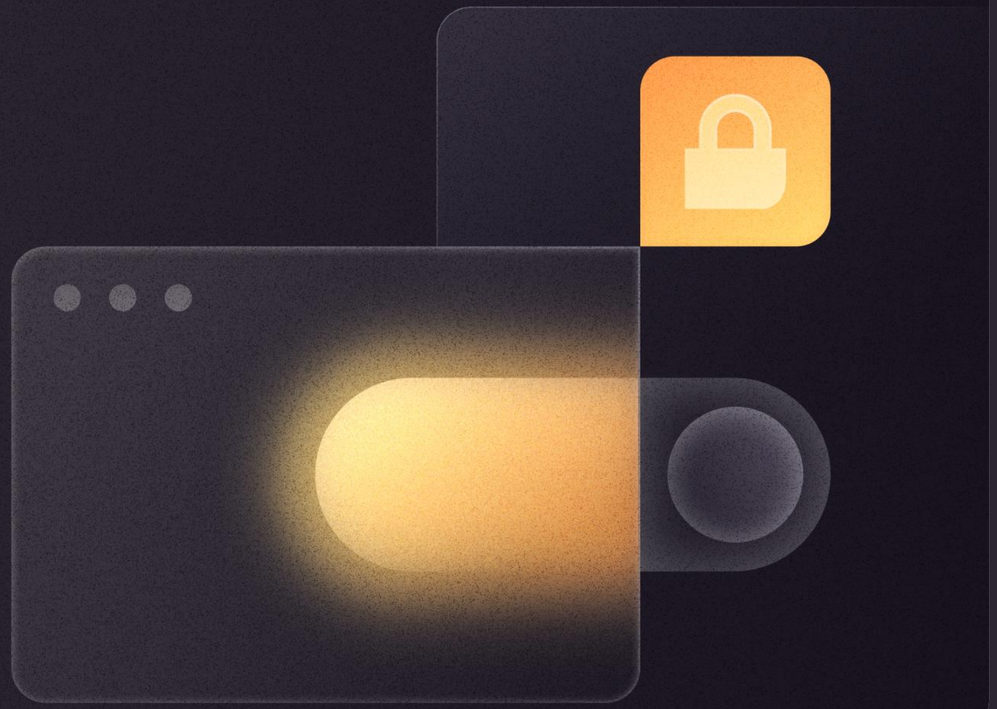
1	Sending GRPC request	rest/request
2	GRPC call process9 completed in 4 millis	rest/request
3	GRPC call process8 completed in 4 millis	rest/request
4	GRPC call process7 completed in 4 millis	rest/request
5	GRPC call process6 completed in 4 millis	rest/request
6	GRPC call process5 completed in 4 millis	rest/request
7	GRPC call process4 completed in 4 millis	rest/request
8	GRPC call process3 completed in 4 millis	rest/request
9	GRPC call process2 completed in 4 millis	rest/request
10	GRPC call process1 completed in 4 millis	rest/request

reset_peer:
attributes:
timeout: 10
toxicity: 0.3

0.7^9 * 0.3 = 0.012105 ~ 1.21%

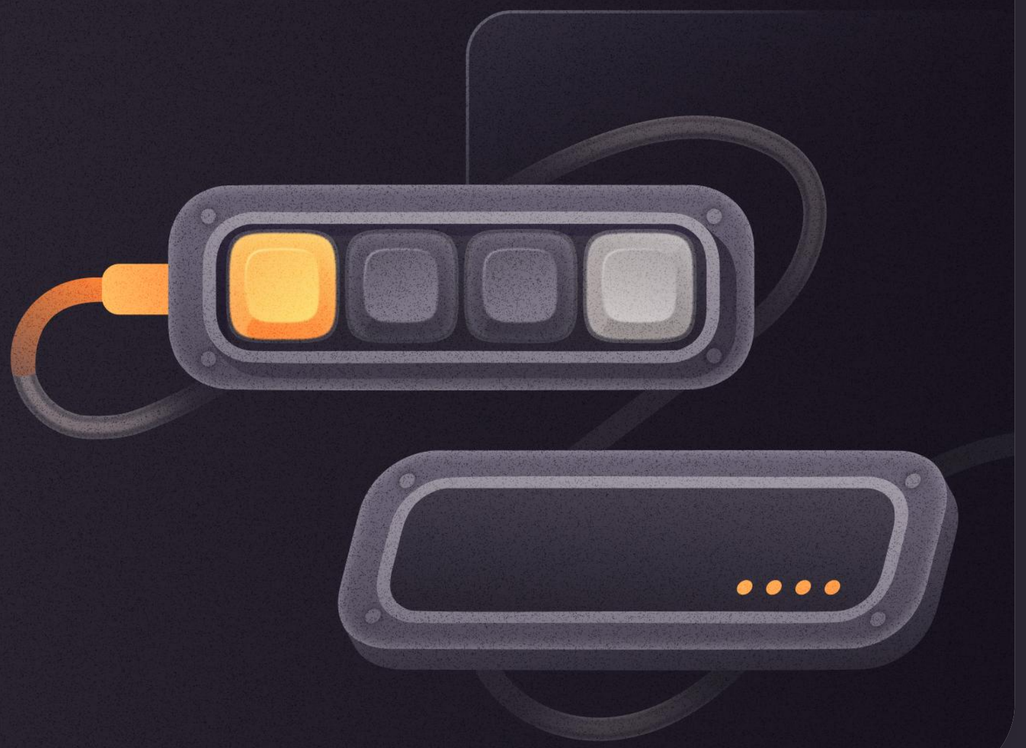
Пустить трафик (нагрузочное тестирование)

Результаты эксперимента



- Неправильная обработка операции
- Некорректные лимиты
- Можем завести клиента в минус
- Повторение ситуации при другом аффекте

Что дает хаос на тесте?



- Безопасное тестирование функциональностей до выкатки на продакшн
- Возможность проверять исправления раннее найденных сбоев
- Экономия ресурсов
- Проведение «учебных сбоев»

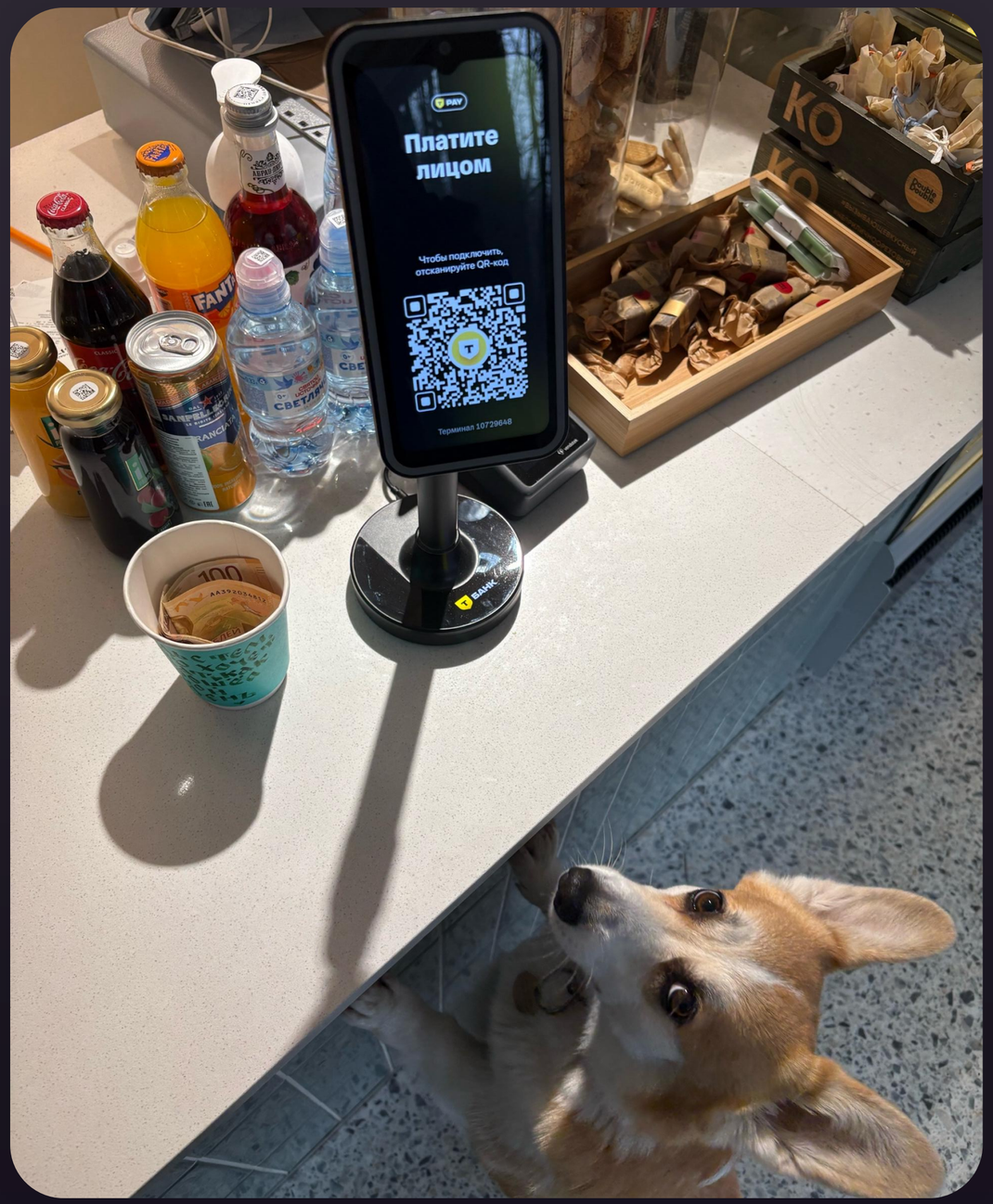
Тестирование в продакшене

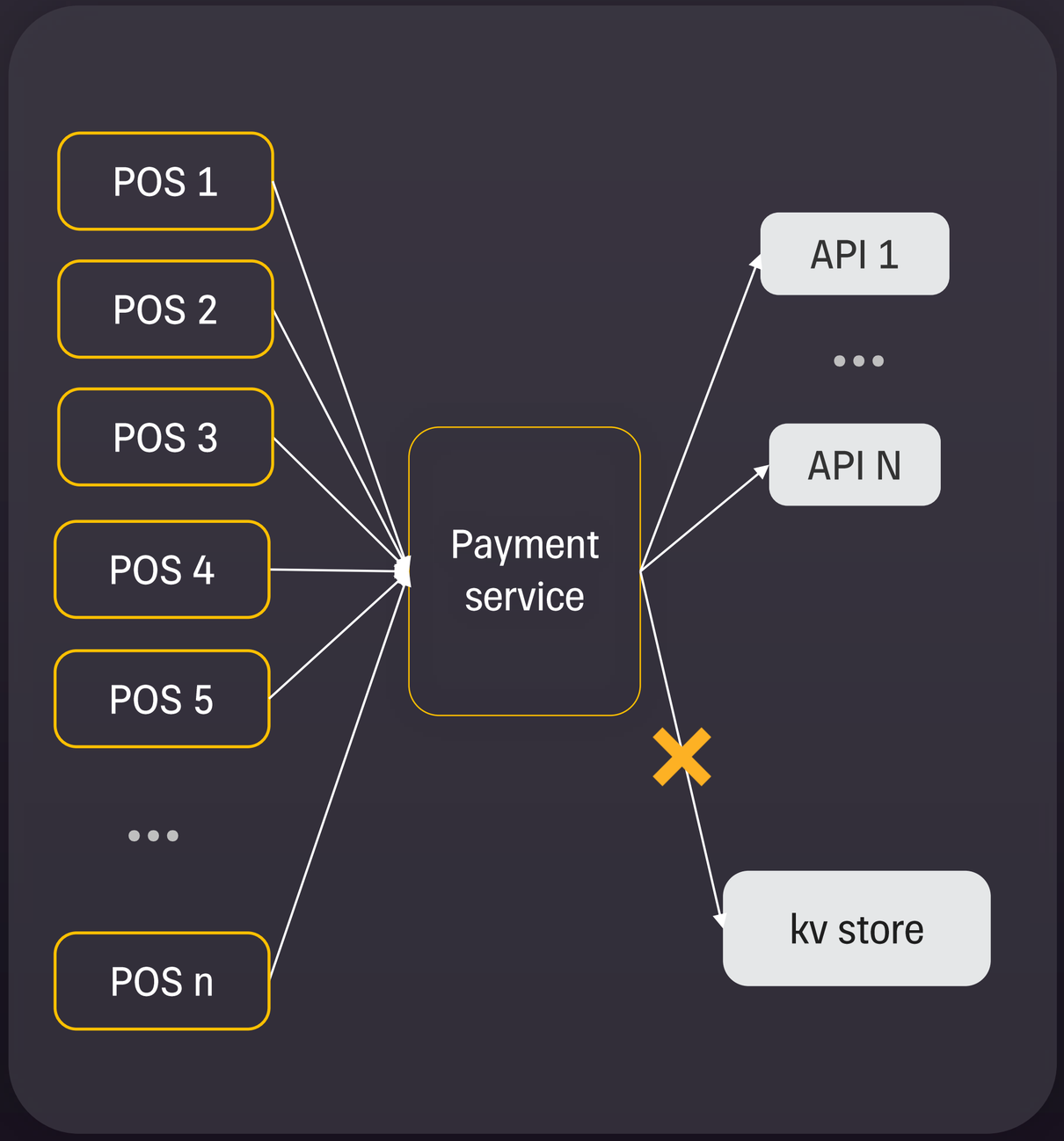
Что дает хаос на продакшене?

Хаос тестирование на тесте не гарантирует, что на продакшене тоже все будет хорошо

Можем тестировать не только сервис, но и процессы, инструменты и самих инженеров

Оплата через POS терминалы

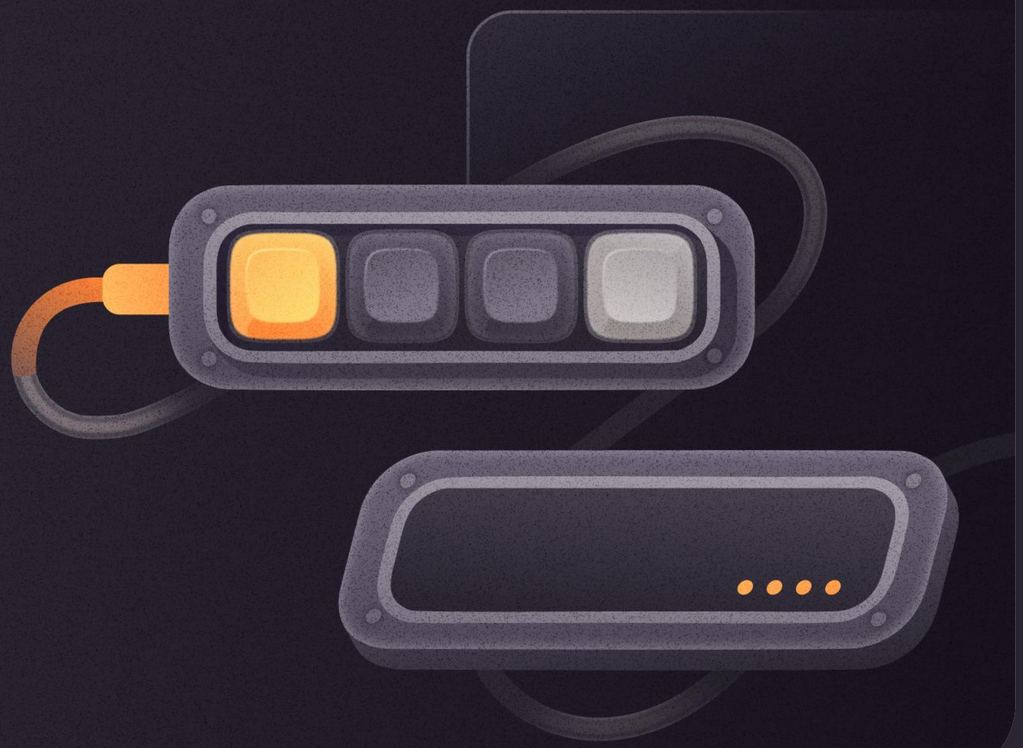




Тестируем работу сервиса в условиях недоступности kv-store

Гипотеза: обработка платежа вырастет на ~200мс

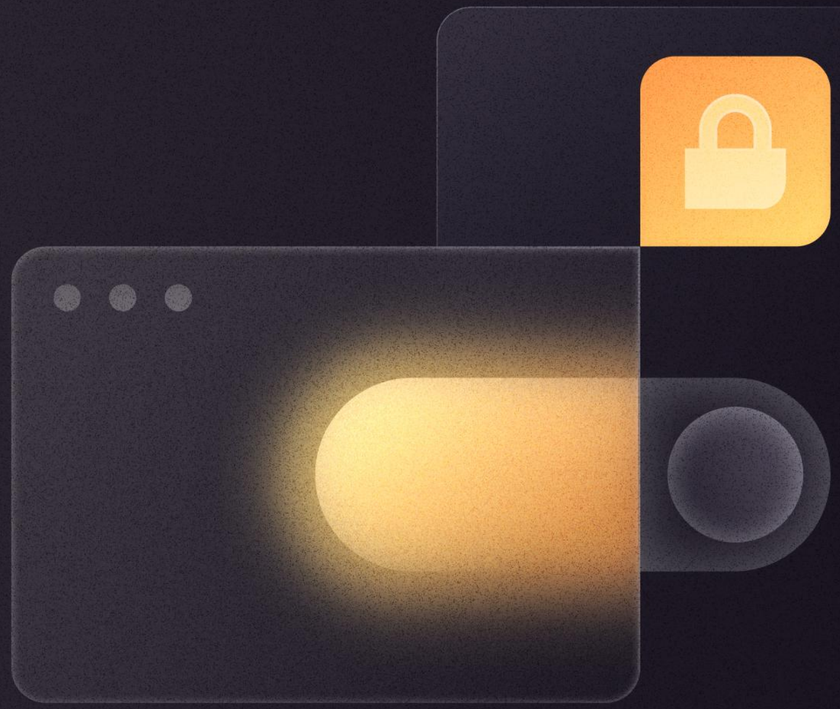
Результаты эксперимента



- Отказ kv-store может положить весь сервис
- Мониторинг находился в неактуальном состоянии
- Проблема обнаружилась только в прайм тайм

Результаты и рекомендации

Как работаем с хаосом?



- Тестируем хаосом новые функциональности перед релизом
- Тестируем хаосом уже существующие критичные фичи
- Проводим «учебные сбои» на тесте и продакшене
- Проводим митапы для расширения аудитории инструмента

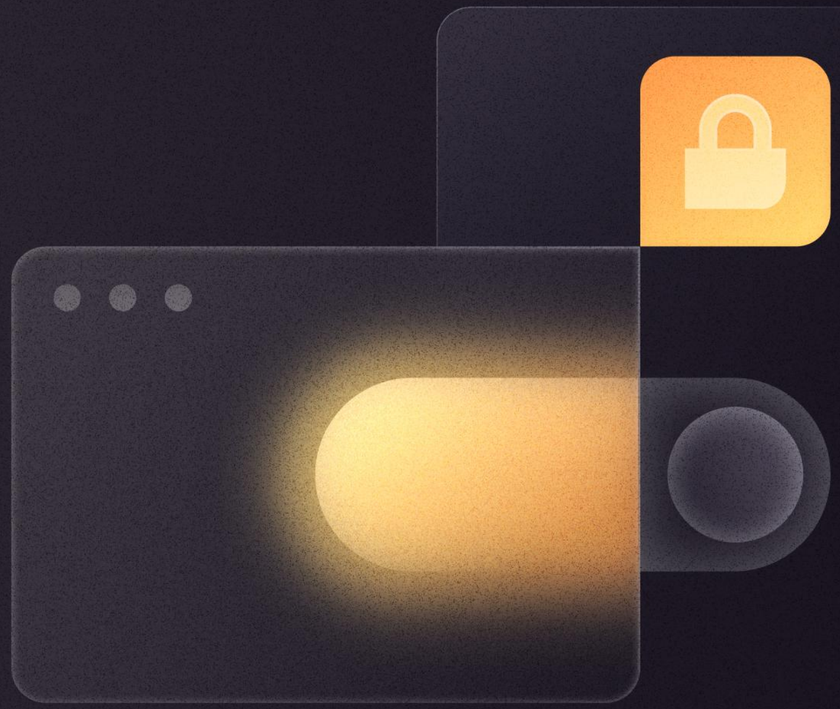
Рекомендации к применению

Понять, нужна ли вам практика

Выделить ресурсы на создание инструмента

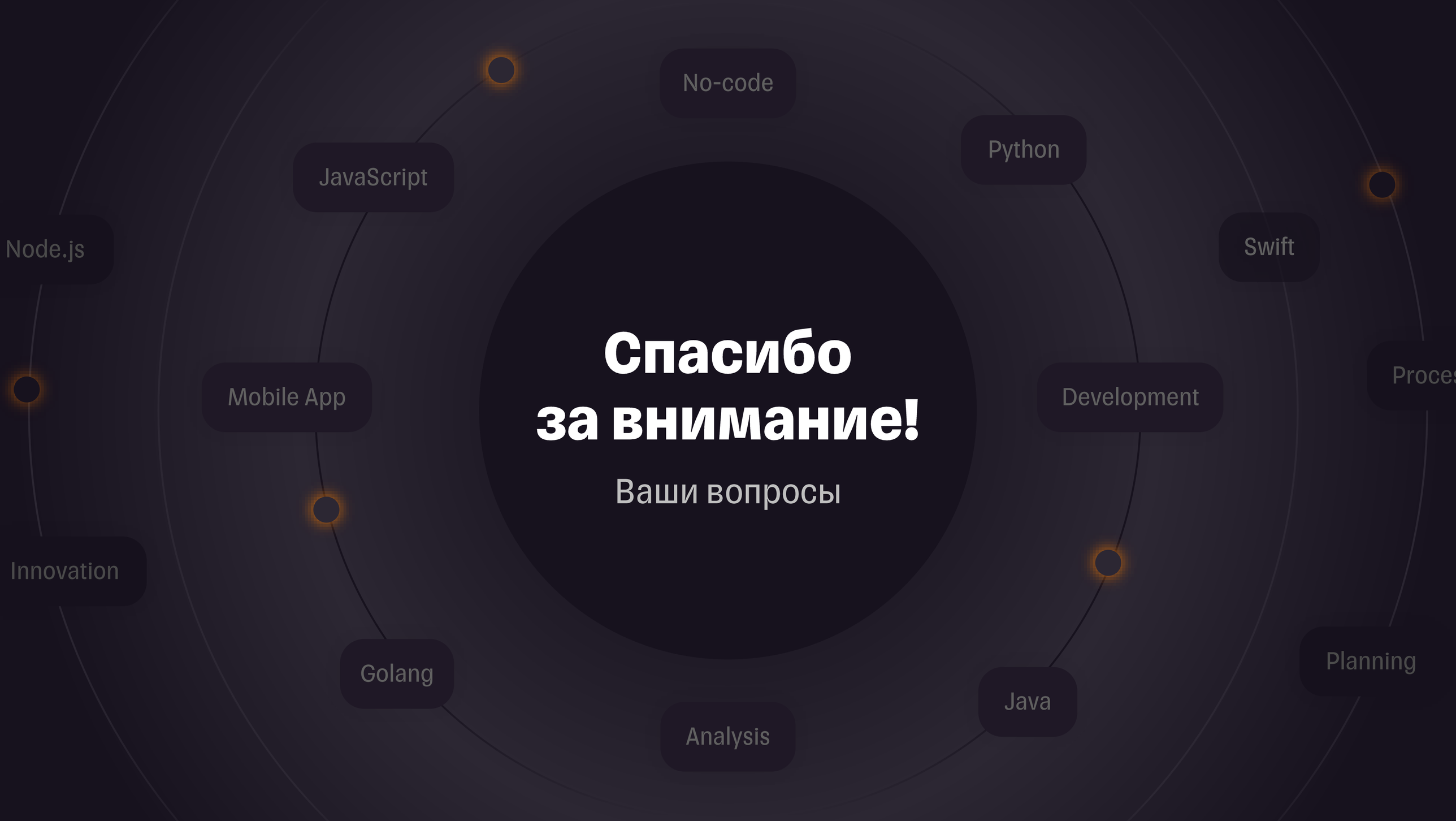
Обучить практике и внедрить в команды

Советы



- Начинайте с малого:
  один сервис – один тип сбоя
- Оповещайте коллег о проведении экспериментов
- Проводите эксперименты в рабочее время
- Делитесь итогами
  внутри/вне команды

«Не важно, сколько раз ты падал.
Важно, сколько раз ты вставал»



Спасибо за внимание!

Ваши вопросы