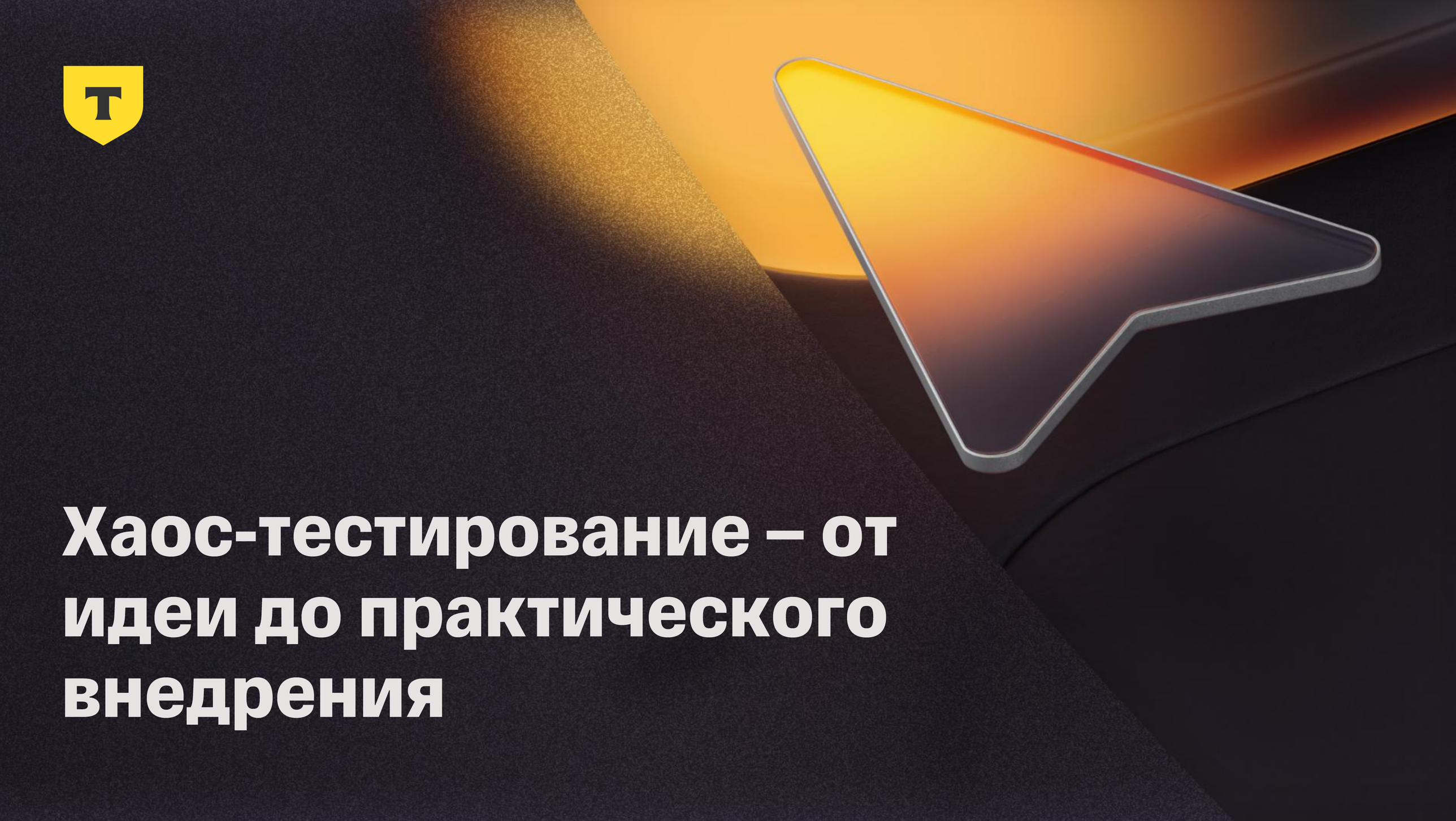


Хаос-тестирование – от идеи до практического внедрения

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

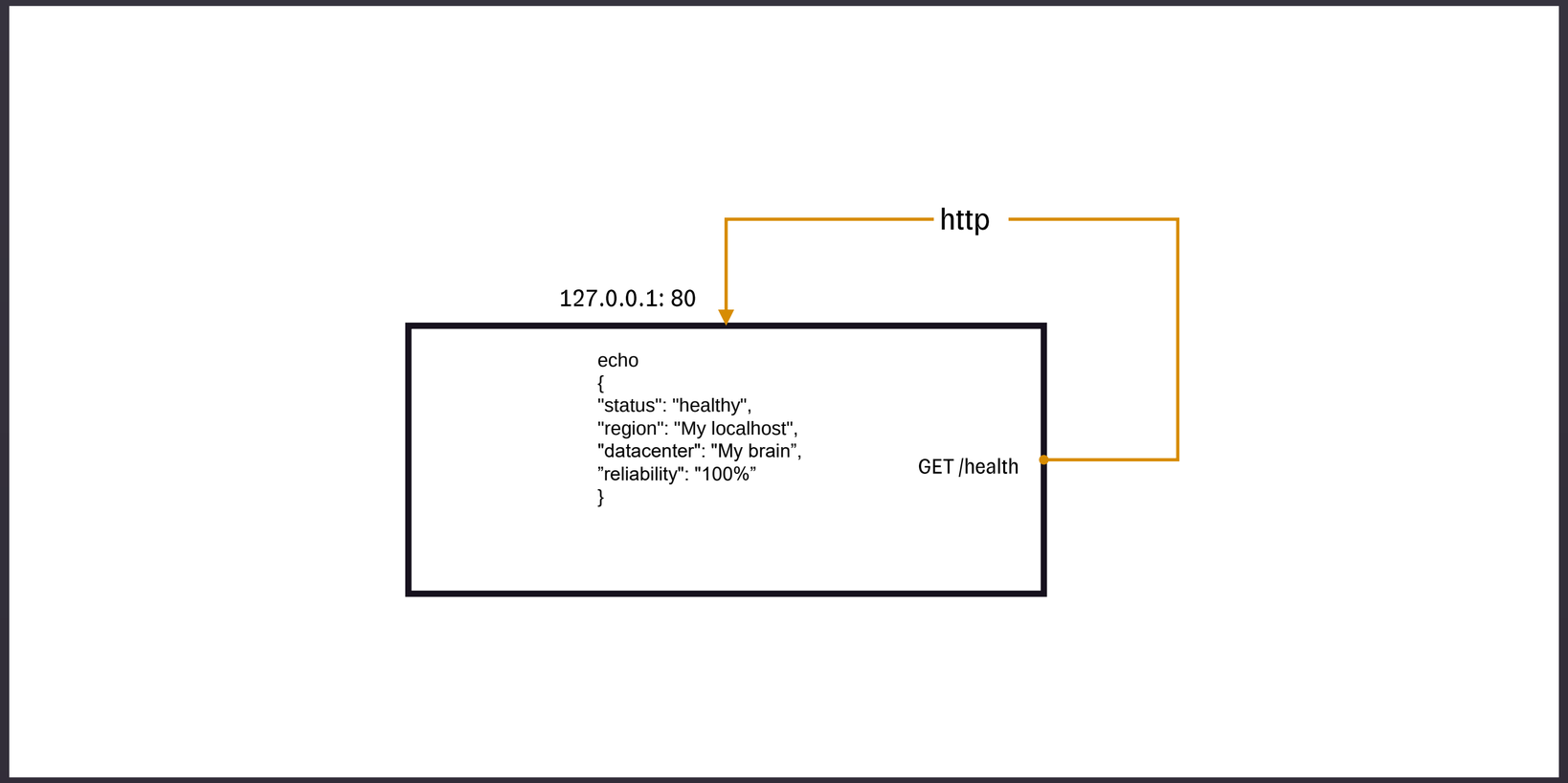
Проблематика

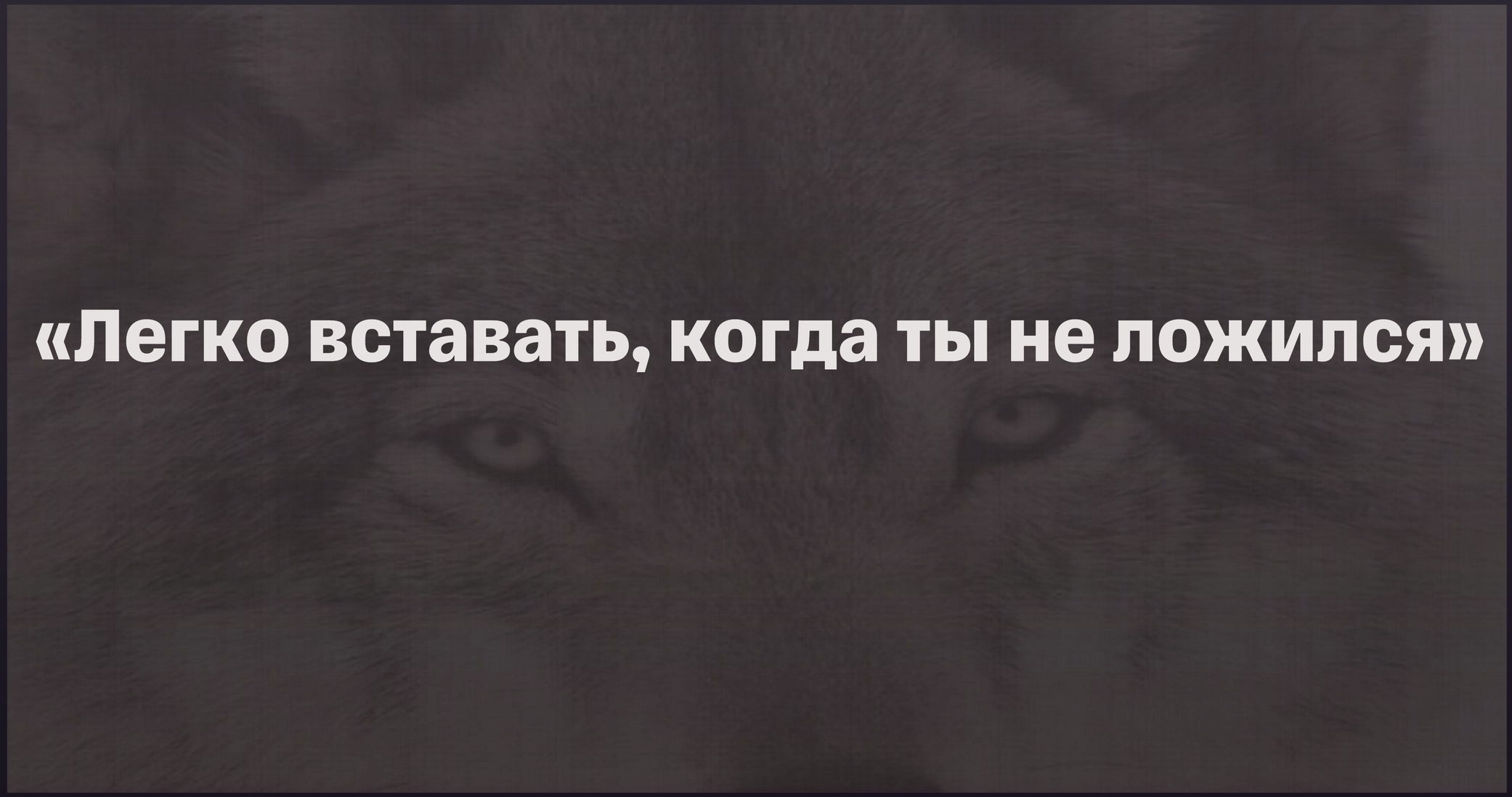
Готов ли ваш сервис к сбоям в инфраструктуре?

Готов ли ваш сервис к сбоям в инфраструктуре?

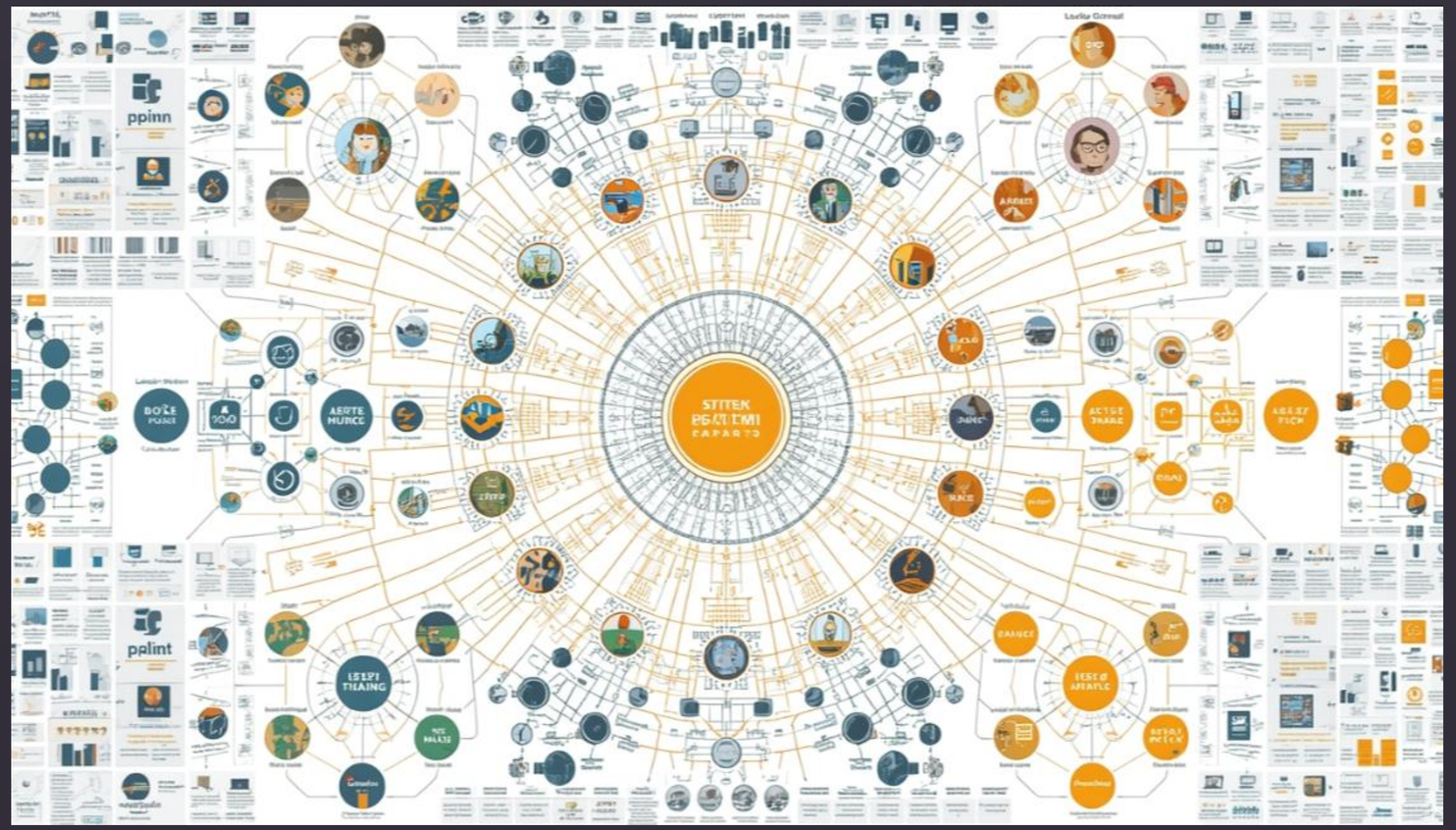
Да! Но это не точно

Можно ли предусмотреть все возможные негативные сценарии отказов?

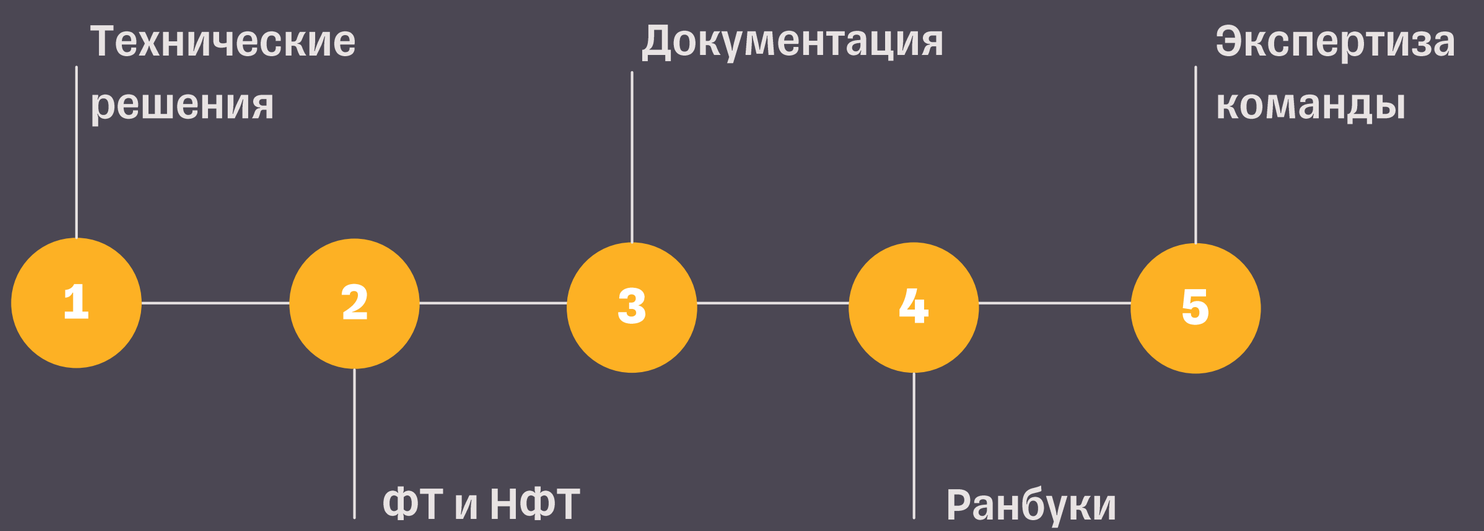



«Легко вставать, когда ты не ложился»

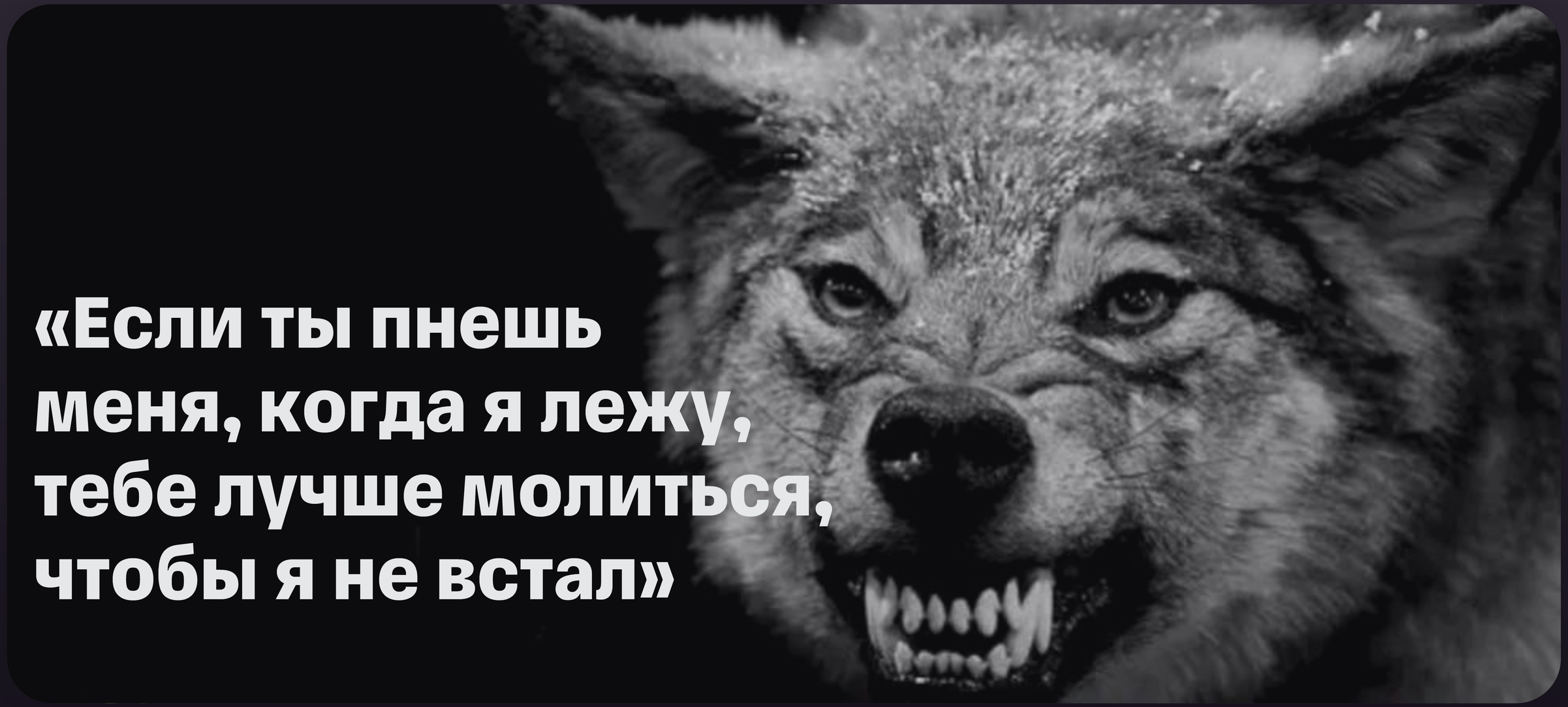
Можно ли предусмотреть все возможные негативные сценарии отказов?



Работаем над отказоустойчивостью



Сервисы все равно падают



«Сбои будут всегда! Важно на них учиться!»

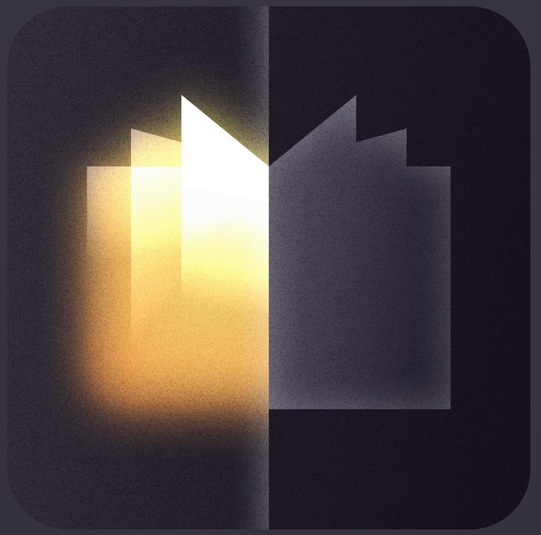
Некоторые проблемы можно обнаружить только через краш-тест

Концепция

Концепция

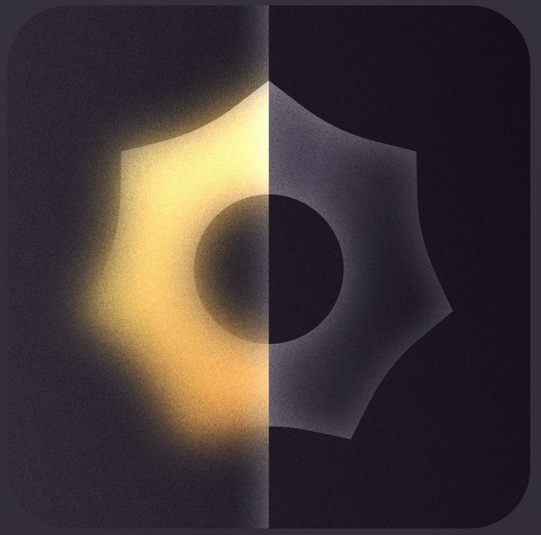
Определение

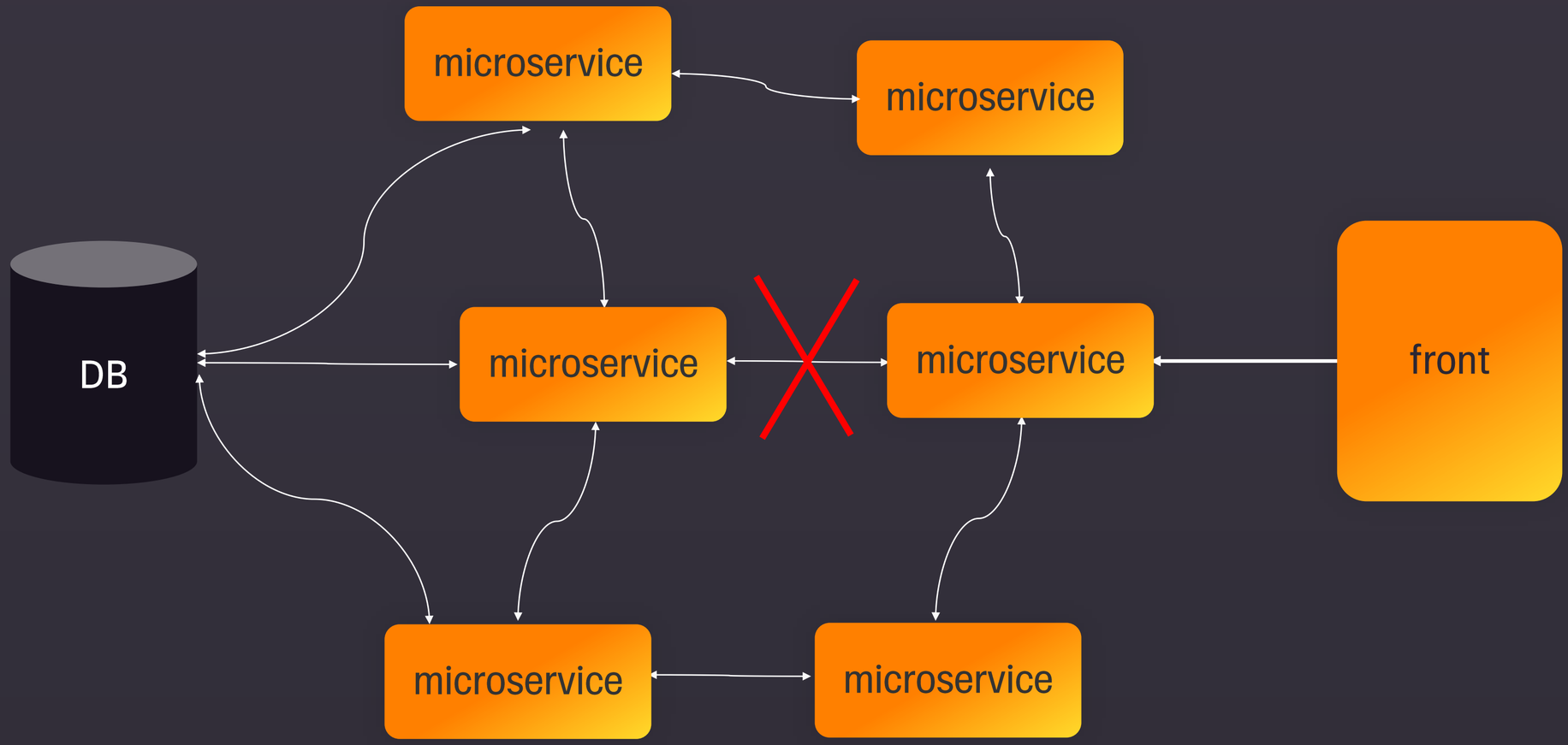
Практика, направленная на проверку отказоустойчивости системы путём намеренного внесения сбоев в её работу в контролируемых условиях.



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

Как проводить хаос-тестирование?

01

Формулировка гипотезы

02

Моделирование сбоя

03

Проведение эксперимента

04

Анализ результатов

Гипотеза

Предполагаем, что «ничего не сломается» или «сломается конкретным образом»

Воспроизведение уже произошедших сбоев

Опыт коллег и SRE

Как проводить хаос-тестирование?

01

Формулировка гипотезы

02

Моделирование сбоя

03

Проведение эксперимента

04

Анализ результатов

Моделирование сбоя

Выбор гипотезы и аффекта

Определение источника подаваемого трафика

Определение метрик и алертов

Как проводить хаос-тестирование?

01

Формулировка гипотезы

02

Моделирование сбоя

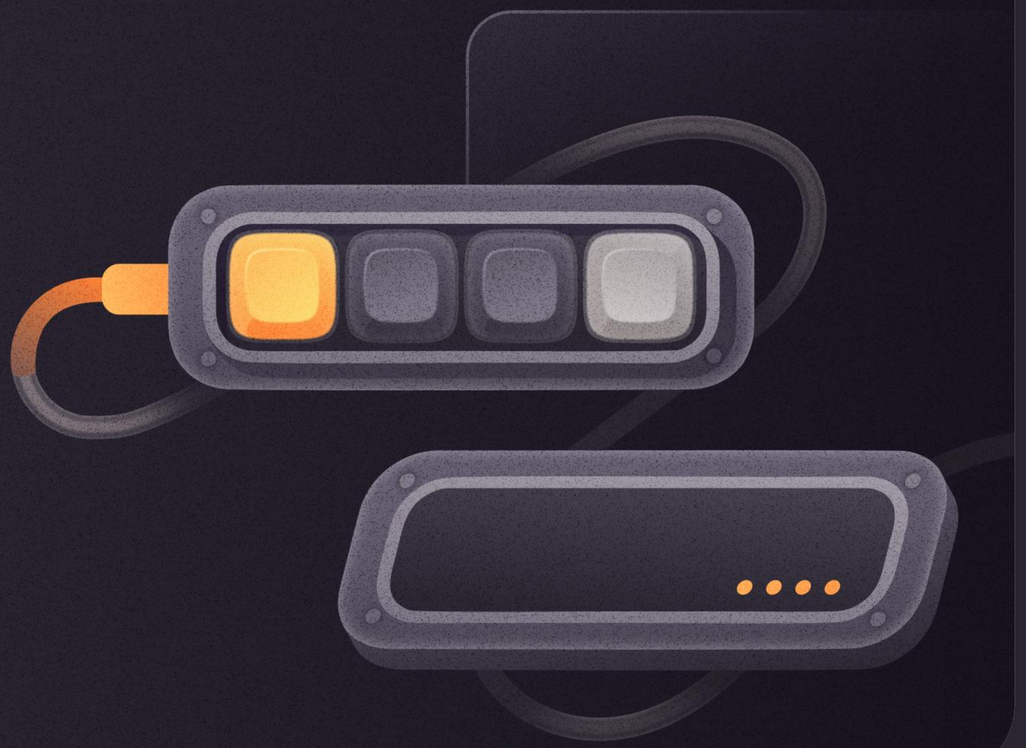
03

Проведение эксперимента

04

Анализ результатов

Проведение эксперимента



- Запуск эксперимента
- Наблюдение за поведением приложения
- Наблюдение за метриками и алертами
- Выключение эксперимента

Как проводить хаос-тестирование?

01

Формулировка гипотезы

02

Моделирование сбоя

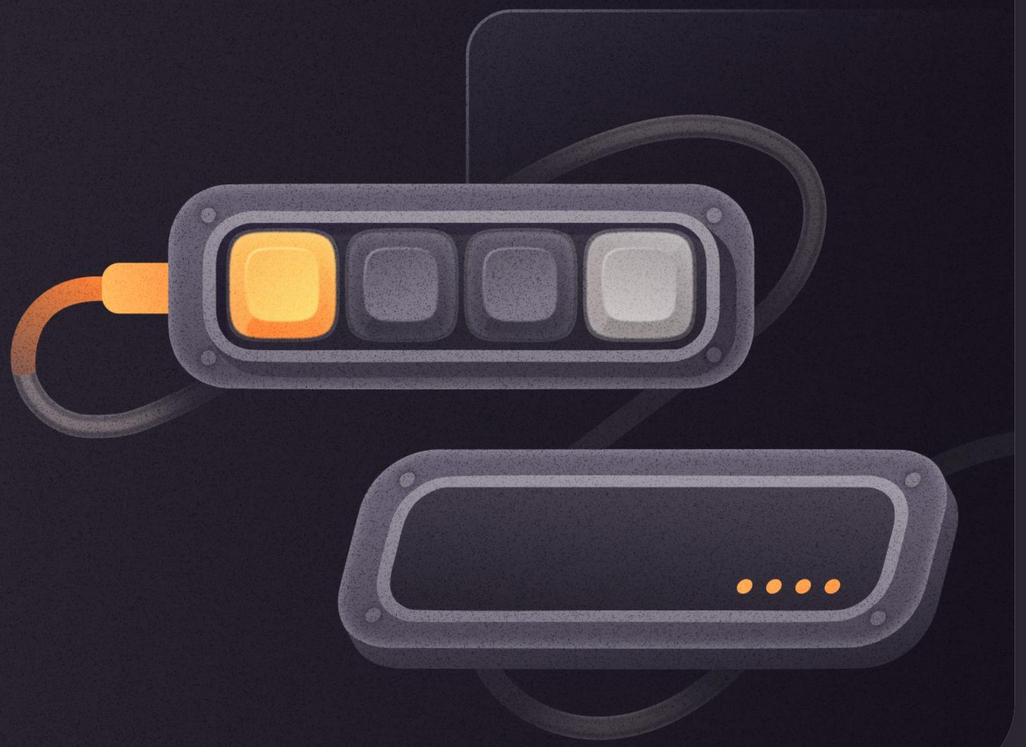
03

Проведение эксперимента

04

Анализ результатов

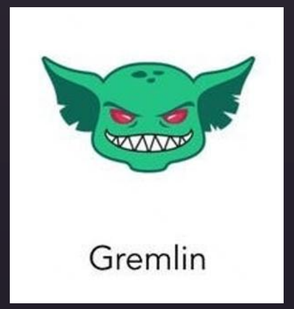
Анализ результатов



- Сравнение результатов с гипотезой
- Проверка восстановления системы
- Анализ алертов и метрик
- Заведение дефектов
- Документирование результатов эксперимента

Разработка

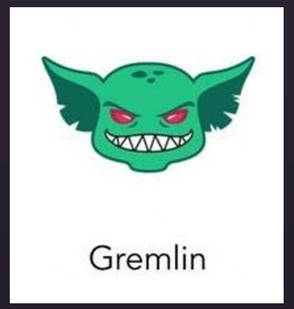
Готовые решения



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

Готовые решения

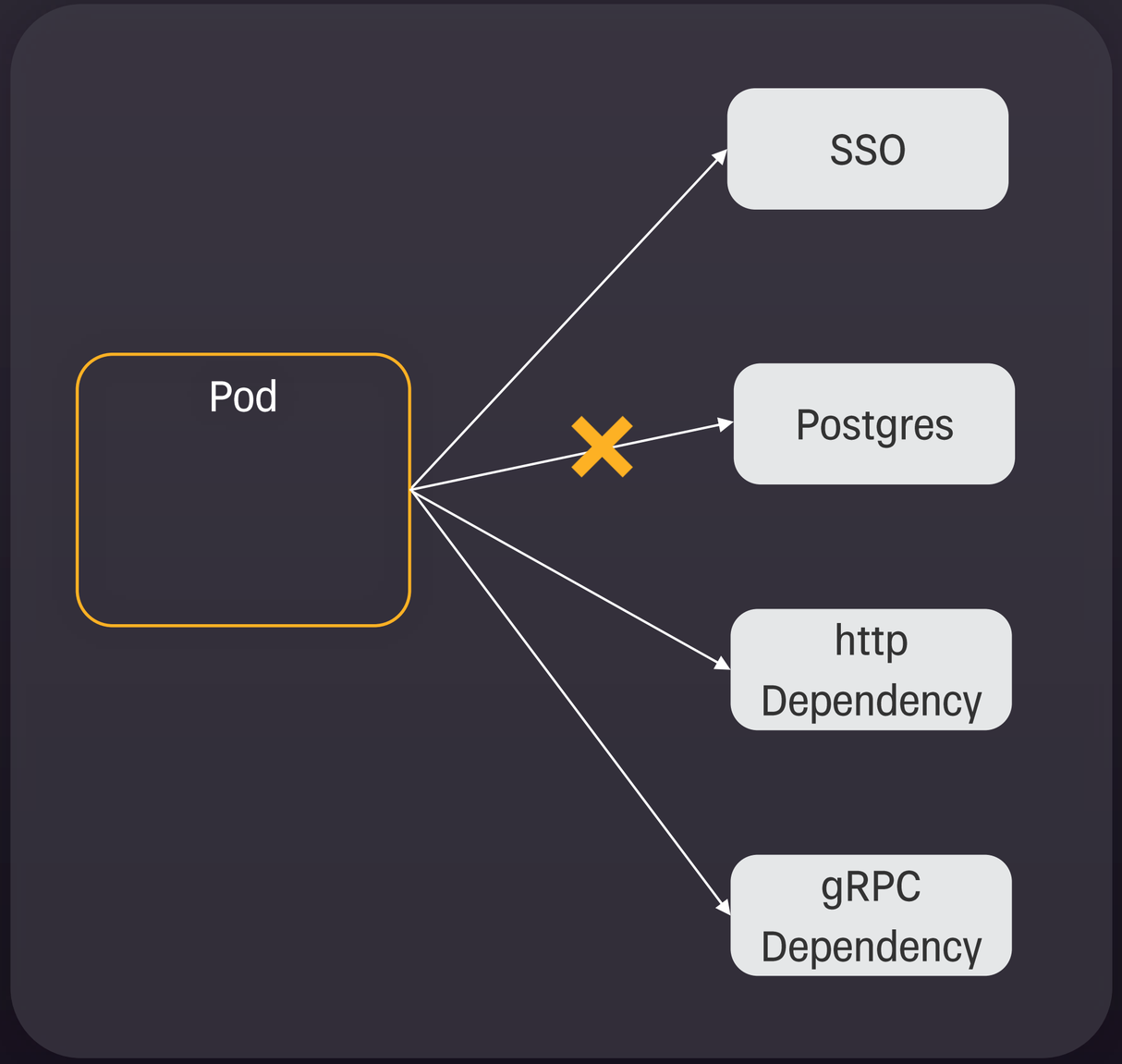


Предлагают большой выбор типов экспериментов - сеть, CPU, Java heap и т.д.

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

Основной фокус – сетевой хаос

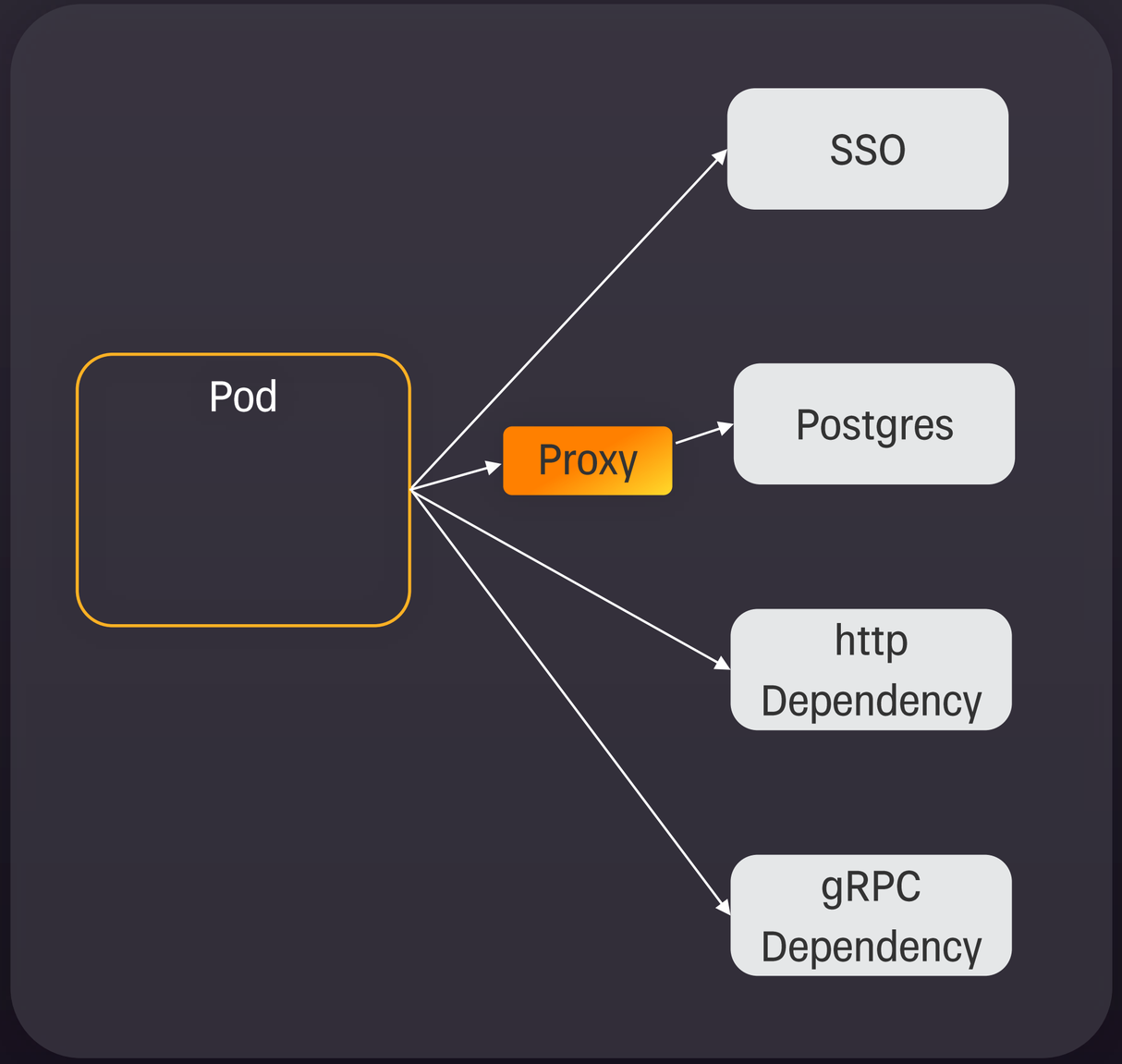
Требования к инструменту



Какой тип экспериментов хотим получить?

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

Требования к инструменту

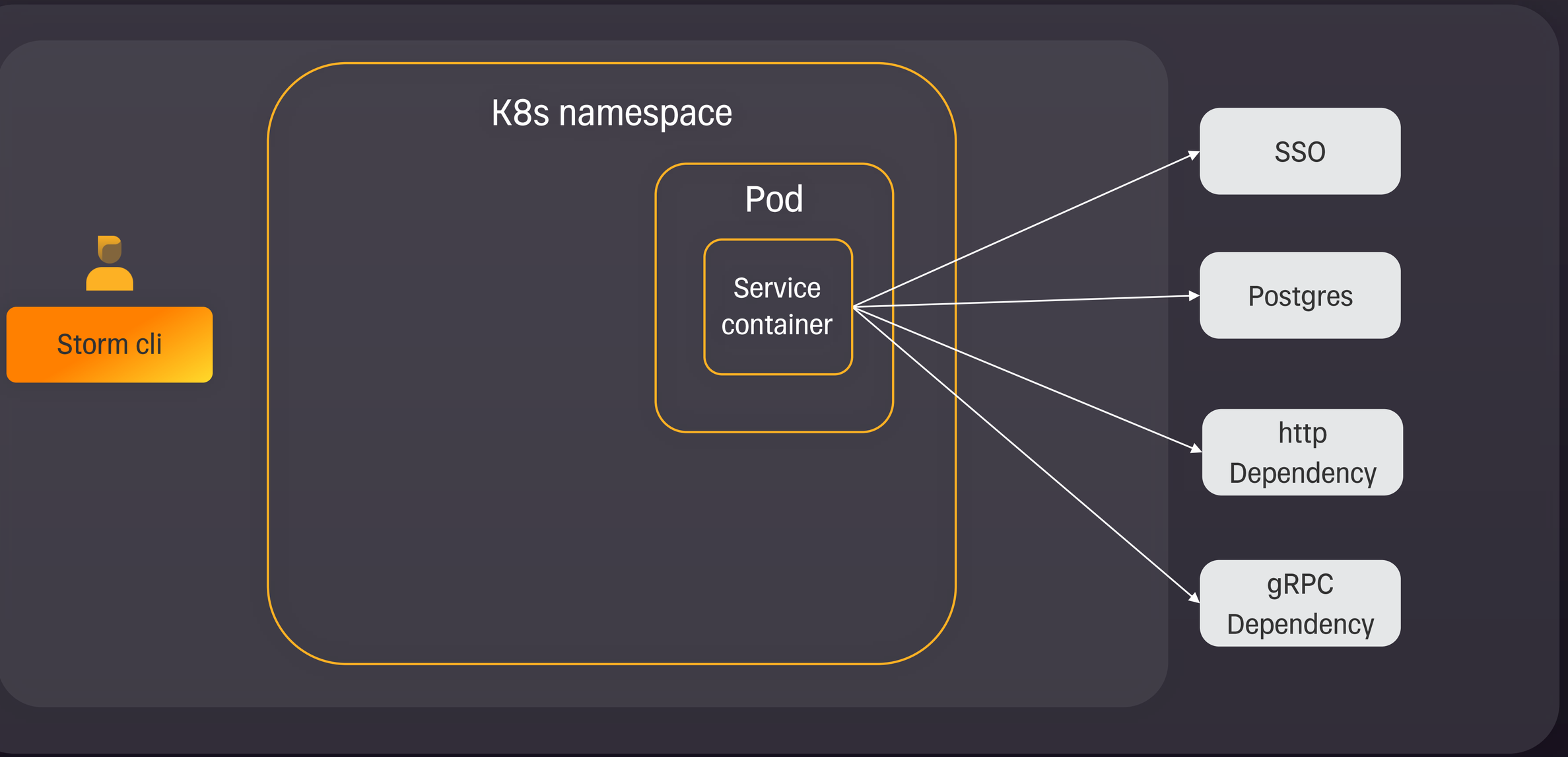


Какой тип экспериментов хотим получить?

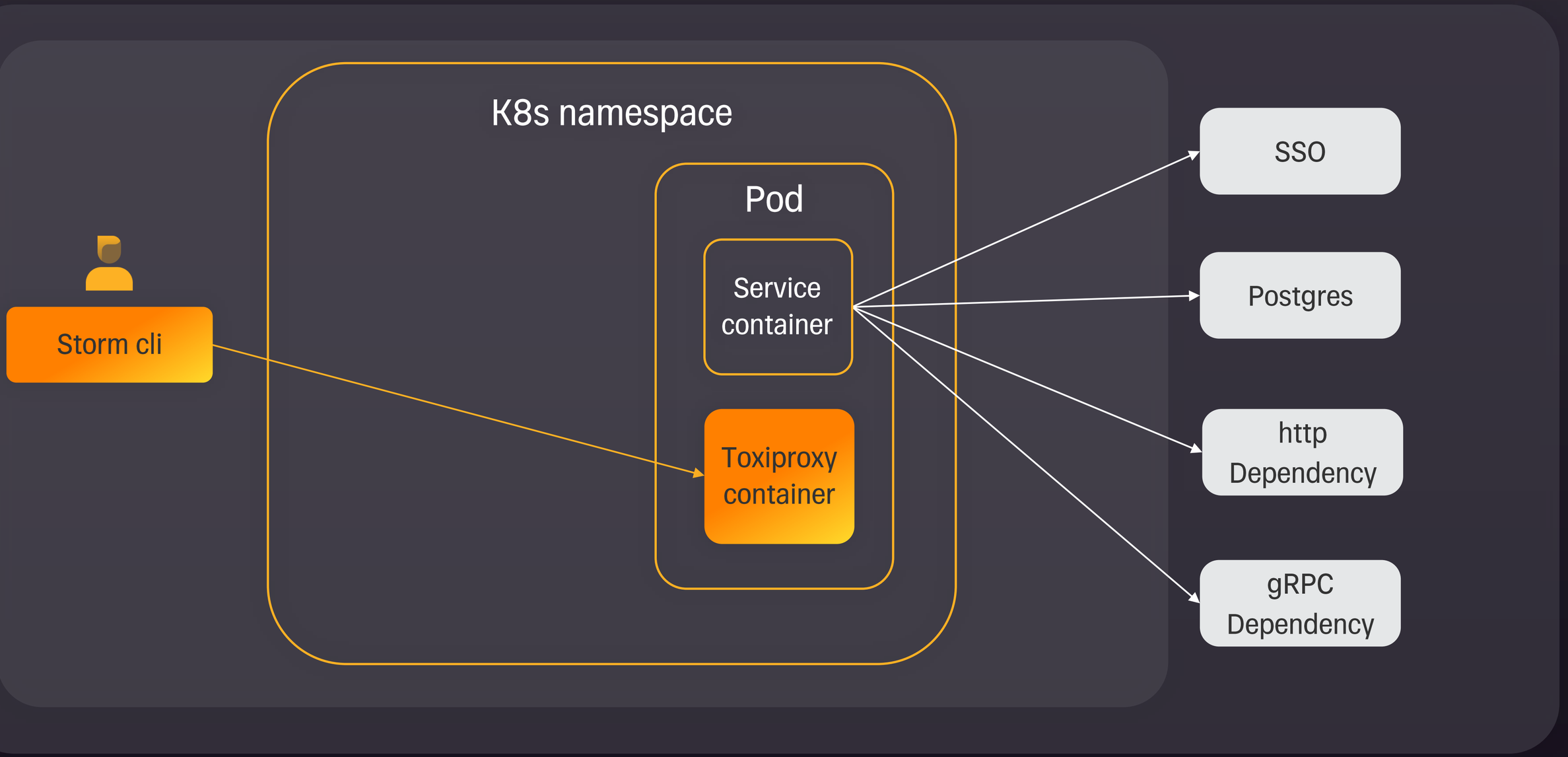
Сетевое влияние на любые зависимости сервиса, имитирующие его работу в нестабильной среде

Собственная реализация Chaos Engineering

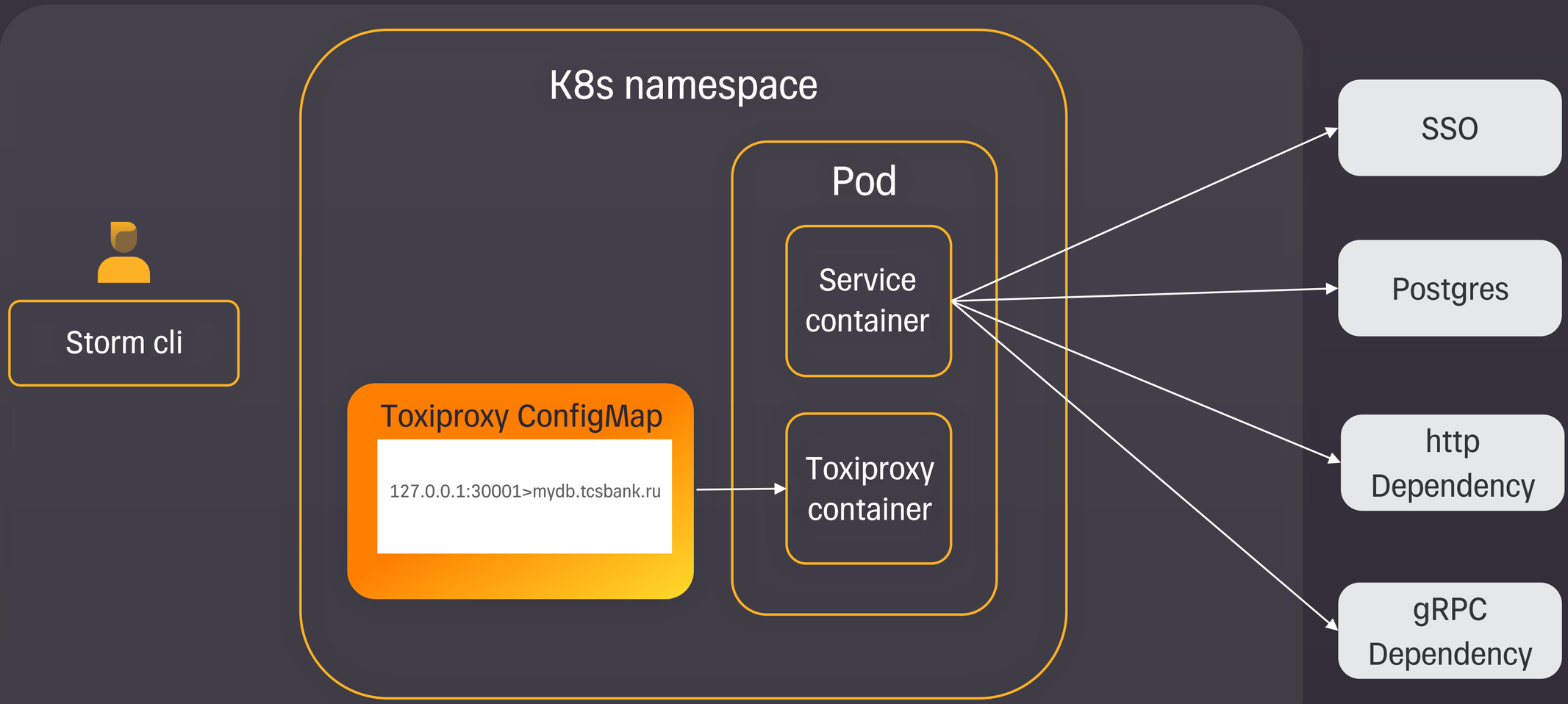
Добавляем sidecar прокси



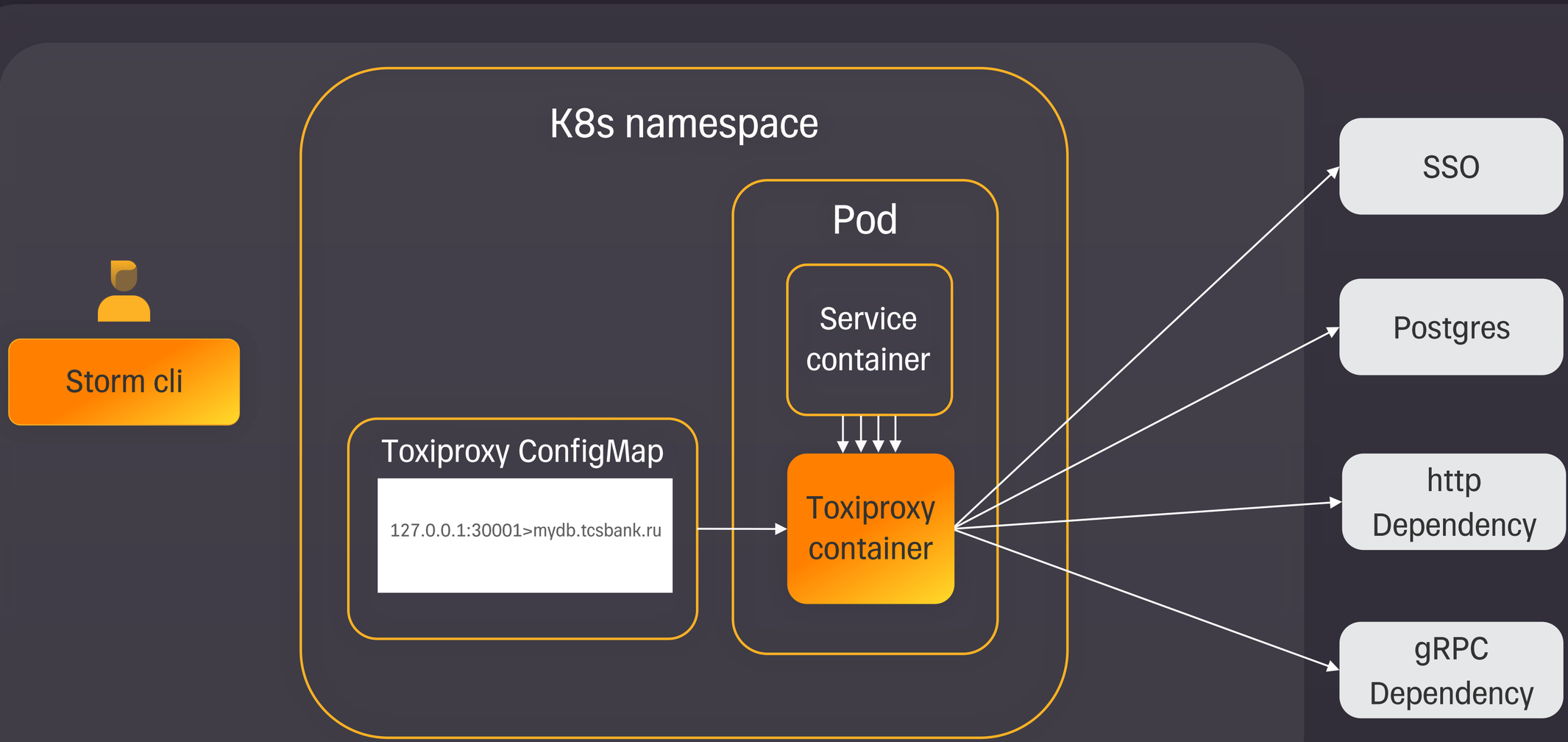
Добавляем sidecar прокси



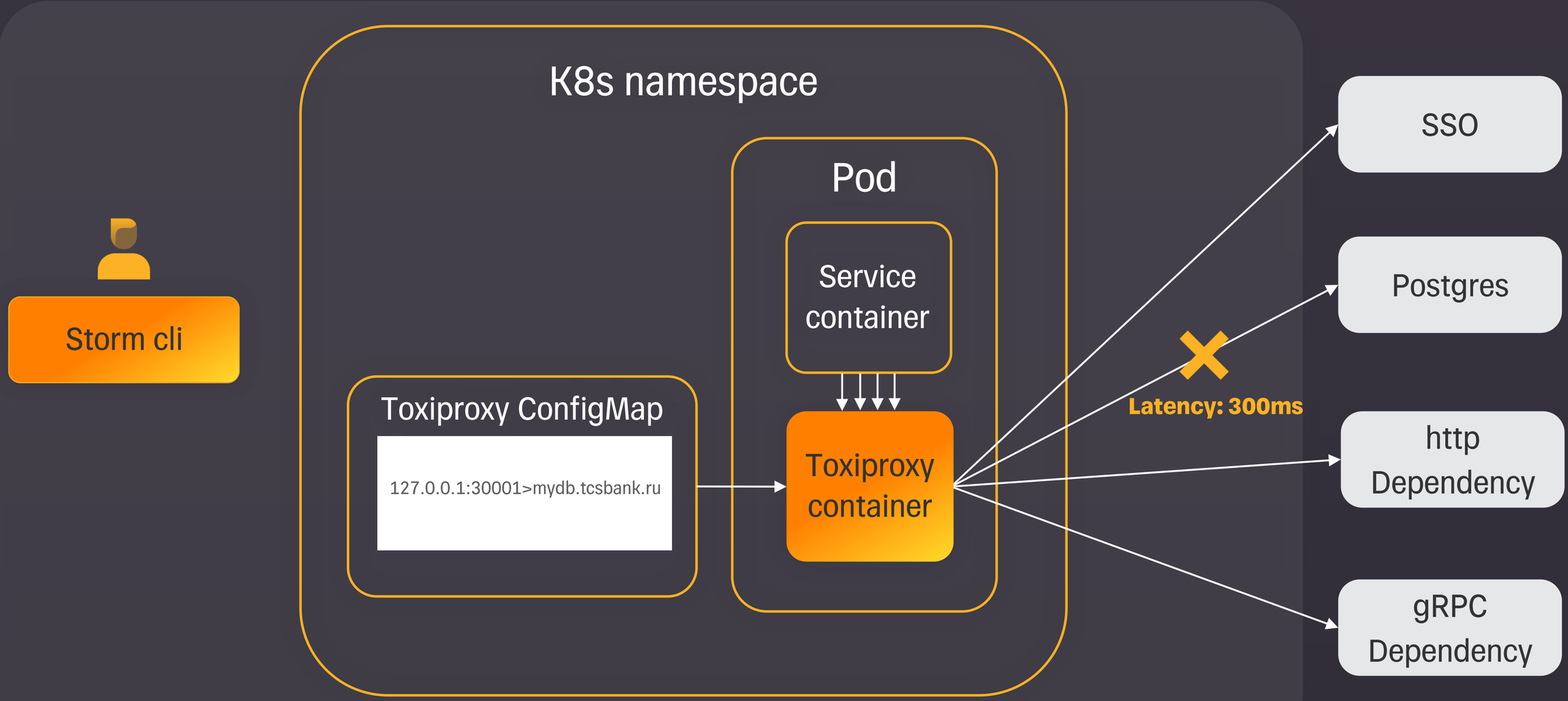
Добавляем sidecar прокси



Добавляем sidecar прокси



Добавляем sidecar прокси

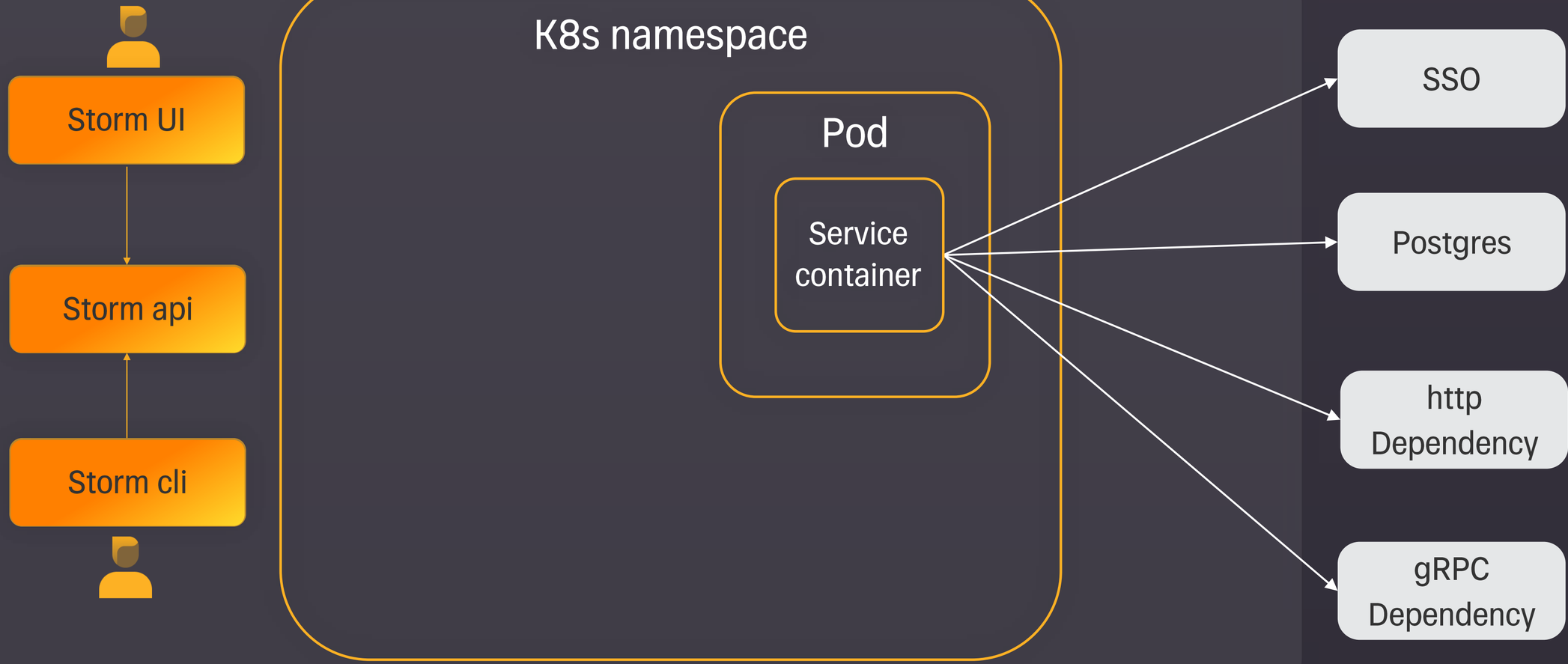


От MVP к продукту

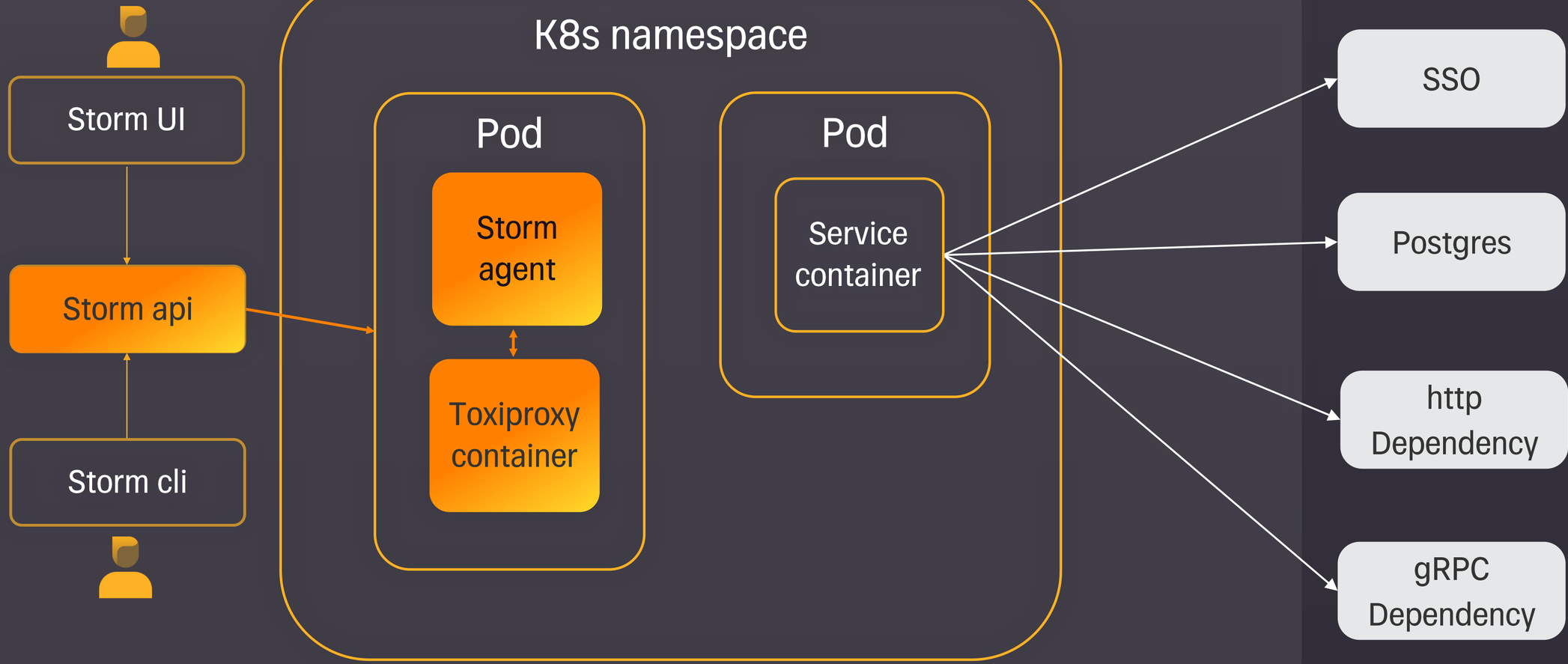
CLI на питоне - не продакшн решение

Обновление ConfigMap слишком долго обрабатывается в k8s

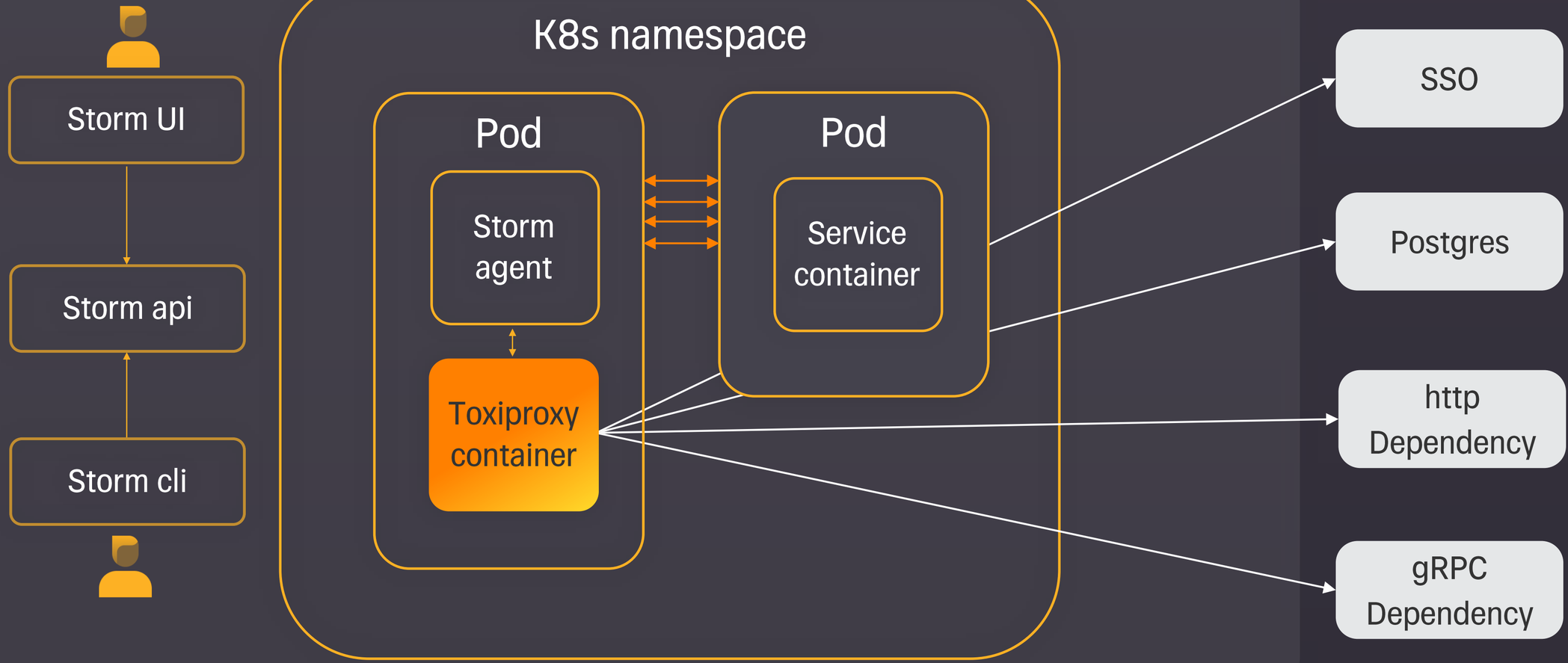
Добавляем STORM API



Добавляем STORM API



Добавляем STORM API



Перехватываем трафик через DNS aliases

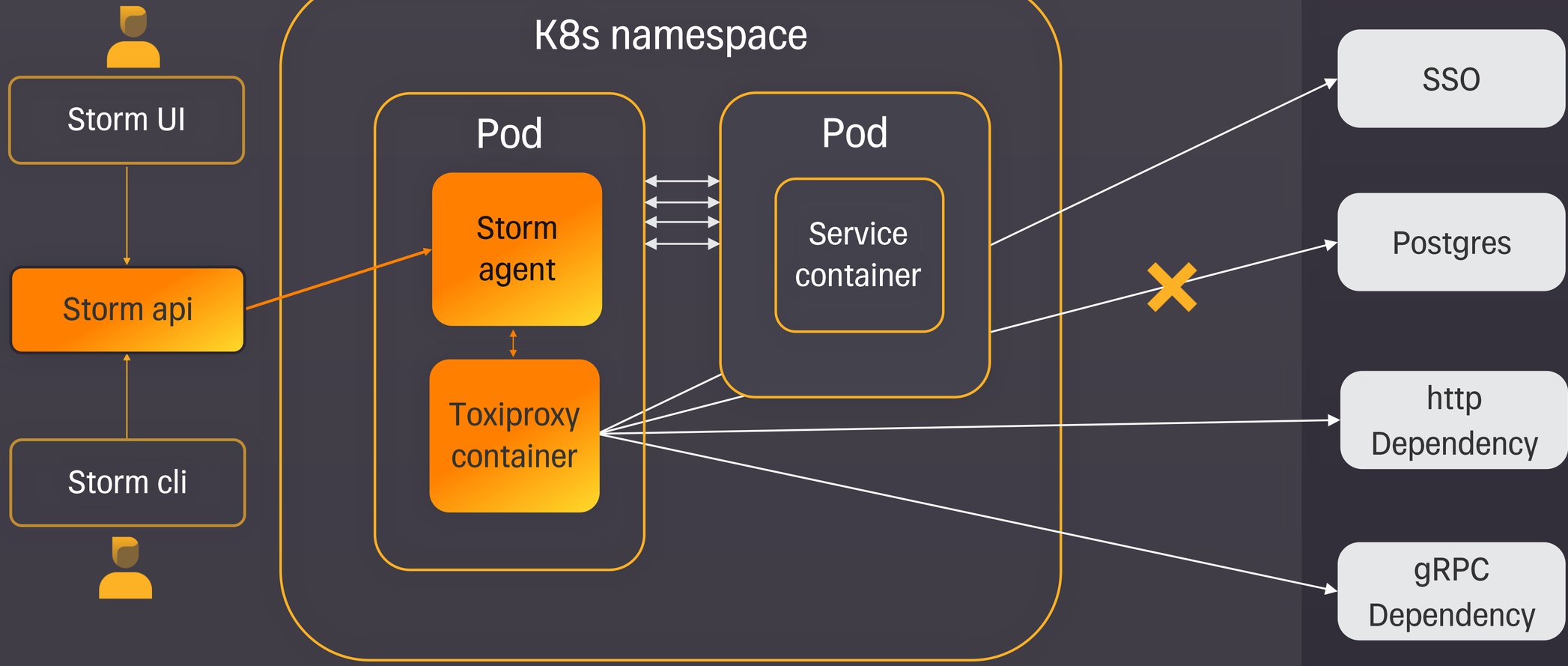
Original × Changed

```
apiVersion: apps/v1
kind: Deployment
spec:
 template:
  metadata:
  spec:
   dnsPolicy: ClusterFirst
   serviceAccount: default
   containers:
    - image: my-service:0.0.1
...
```

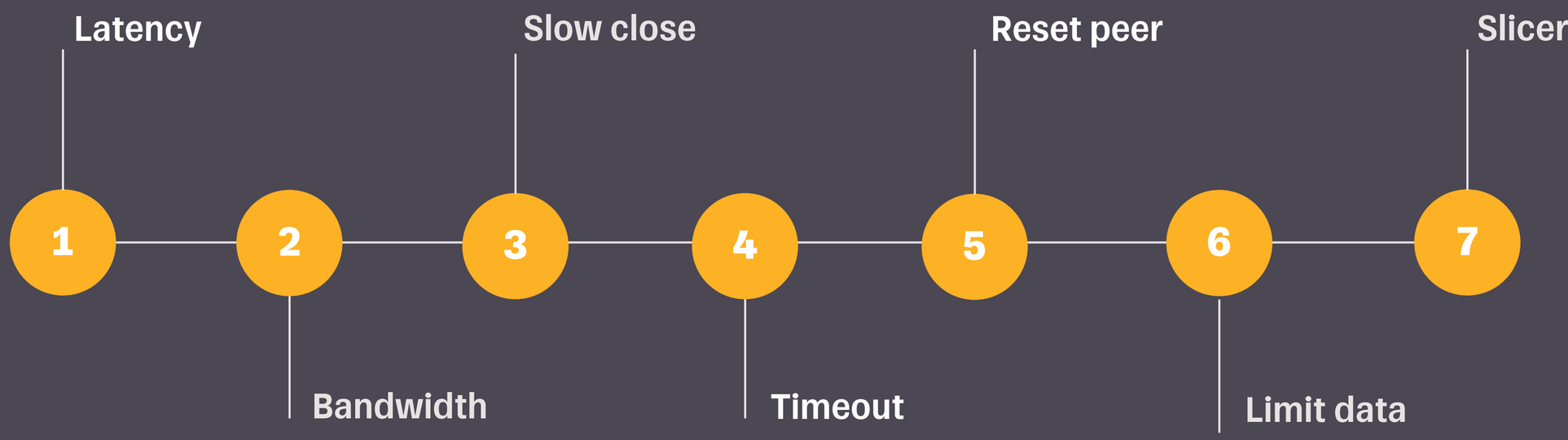

Original Changed ×

```
apiVersion: apps/v1
kind: Deployment
spec:
 template:
  metadata:
  spec:
   dnsPolicy: ClusterFirst
   serviceAccount: default
   hostAliases:
   - hostnames:
    - mydb.tcsbank.ru
    ip: 10.110.170.3
   - hostnames:
    - http-dependency.tcsbank.ru
    ip: 10.110.159.16
   containers:
    - image: my-service:0.0.1
...
```

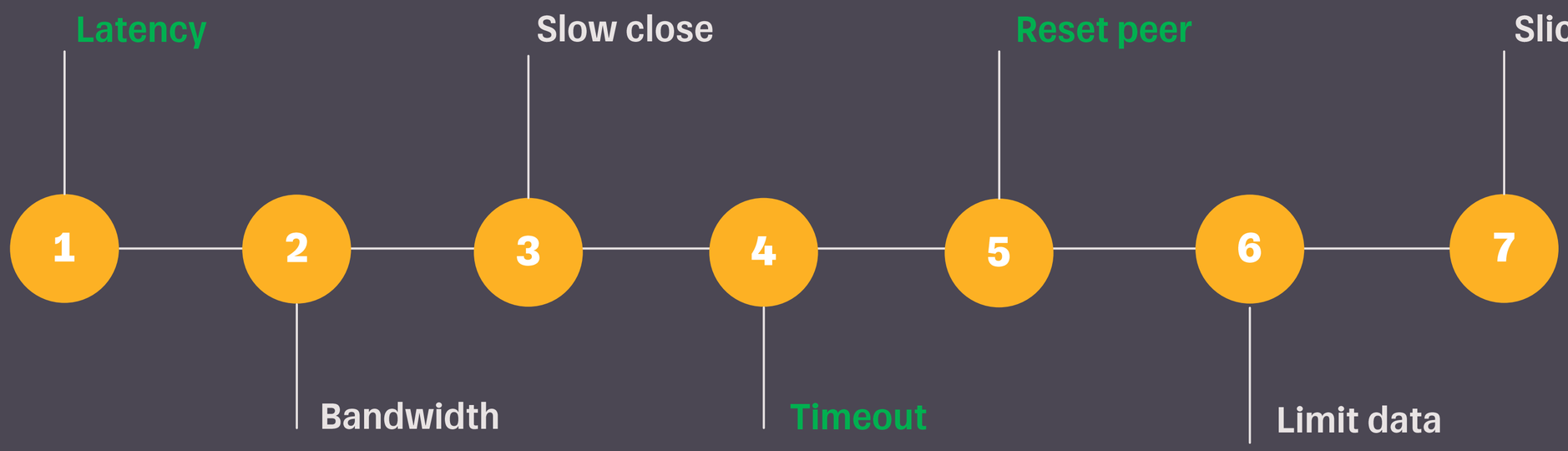
Добавляем STORM API



Виды аффектов



Виды аффектов



Описание аффекта

У каждого аффекта есть свои параметры

```
toxics_config:
  test_dataabase_ru_443_my_db:
    type: tcp
    bandwidth:
      attributes:
        rate: 10
      toxicity: 1
      enabled: false
    latency:
      attributes:
        jitter: 20
        latency: 100
      toxicity: 1
      enabled: false
    limit_data:
      attributes:
        bytes: 10140
      toxicity: 1
      enabled: false
    reset_peer:
      attributes:
        timeout: 1
      toxicity: 1
      enabled: false
    slicer:
      attributes:
        average_size: 100
        delay: 500
        size_variation: 10
      toxicity: 1
      enabled: false
    slow_close:
      attributes:
        delay: 2000
      toxicity: 1
      enabled: false
    timeout:
      attributes:
        timeout: 5000
      toxicity: 1
      enabled: false
```

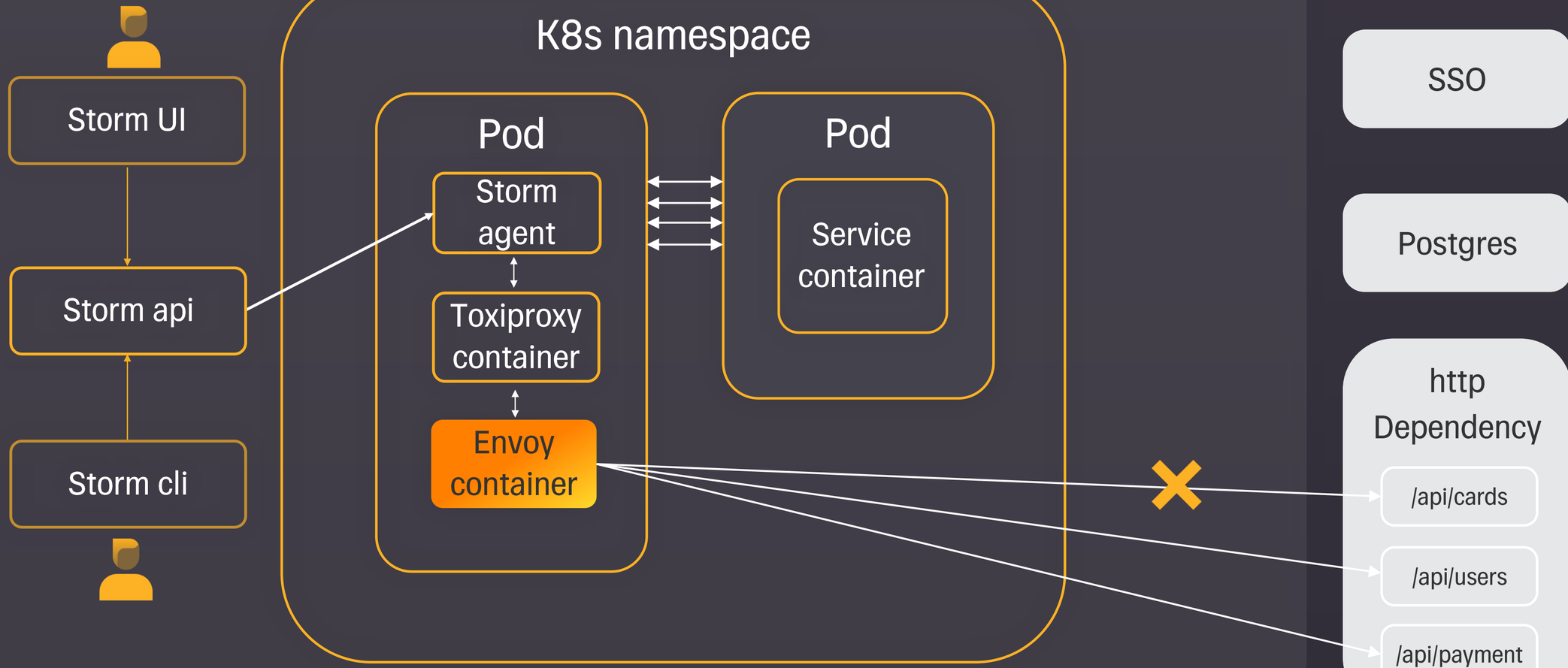
Toxicity	Реальный % срабатывания
0	0%
0.5	44%
0.8	81%
0.99	98%
1	100%

Envoy

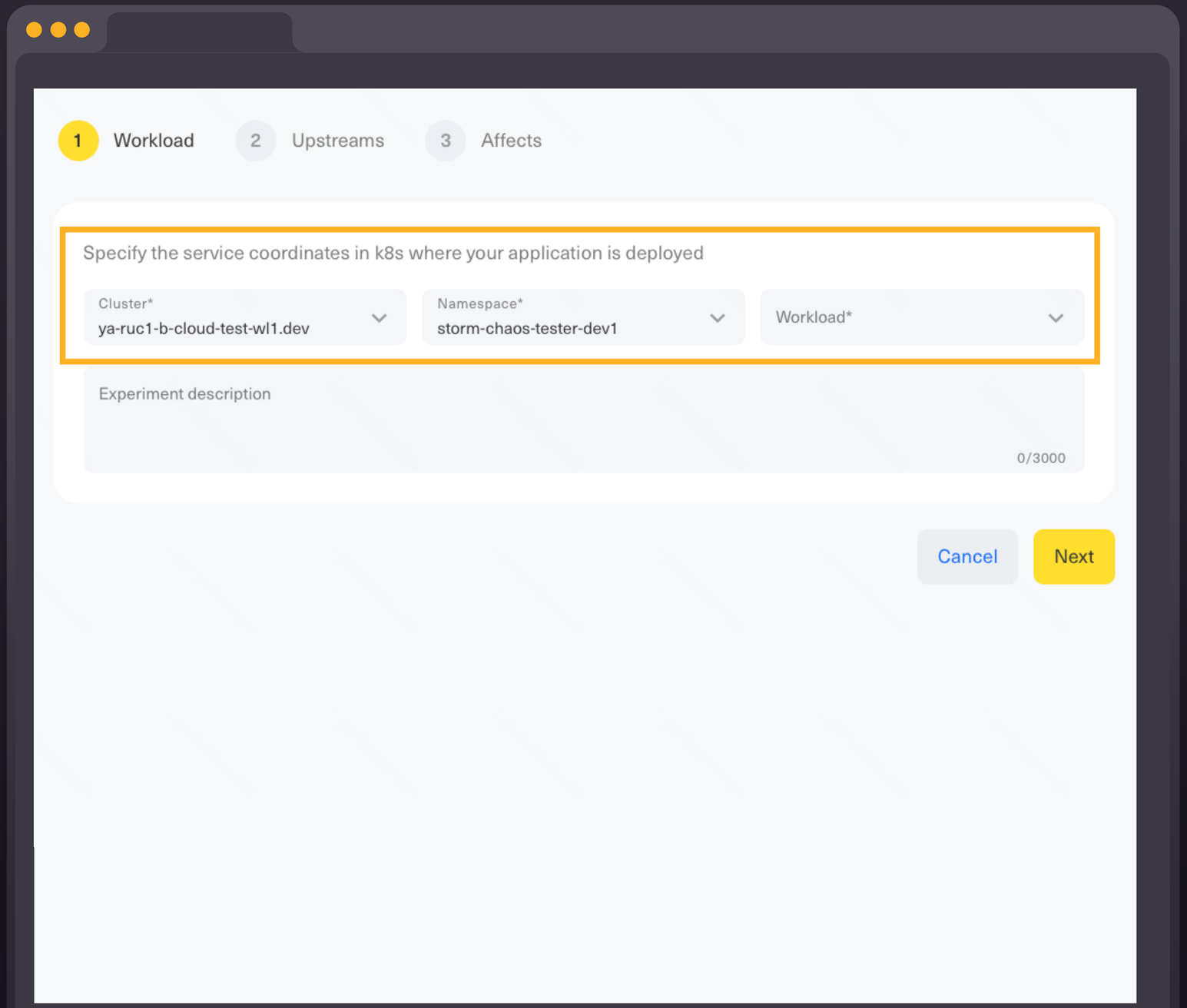
Хотим отдавать различные http коды ответа и оказывать хаос влияние на конкретные методы приложений

QA инженеры хотят влиять на мокированные интеграции (конкретные методы), которые работают на одном хосте

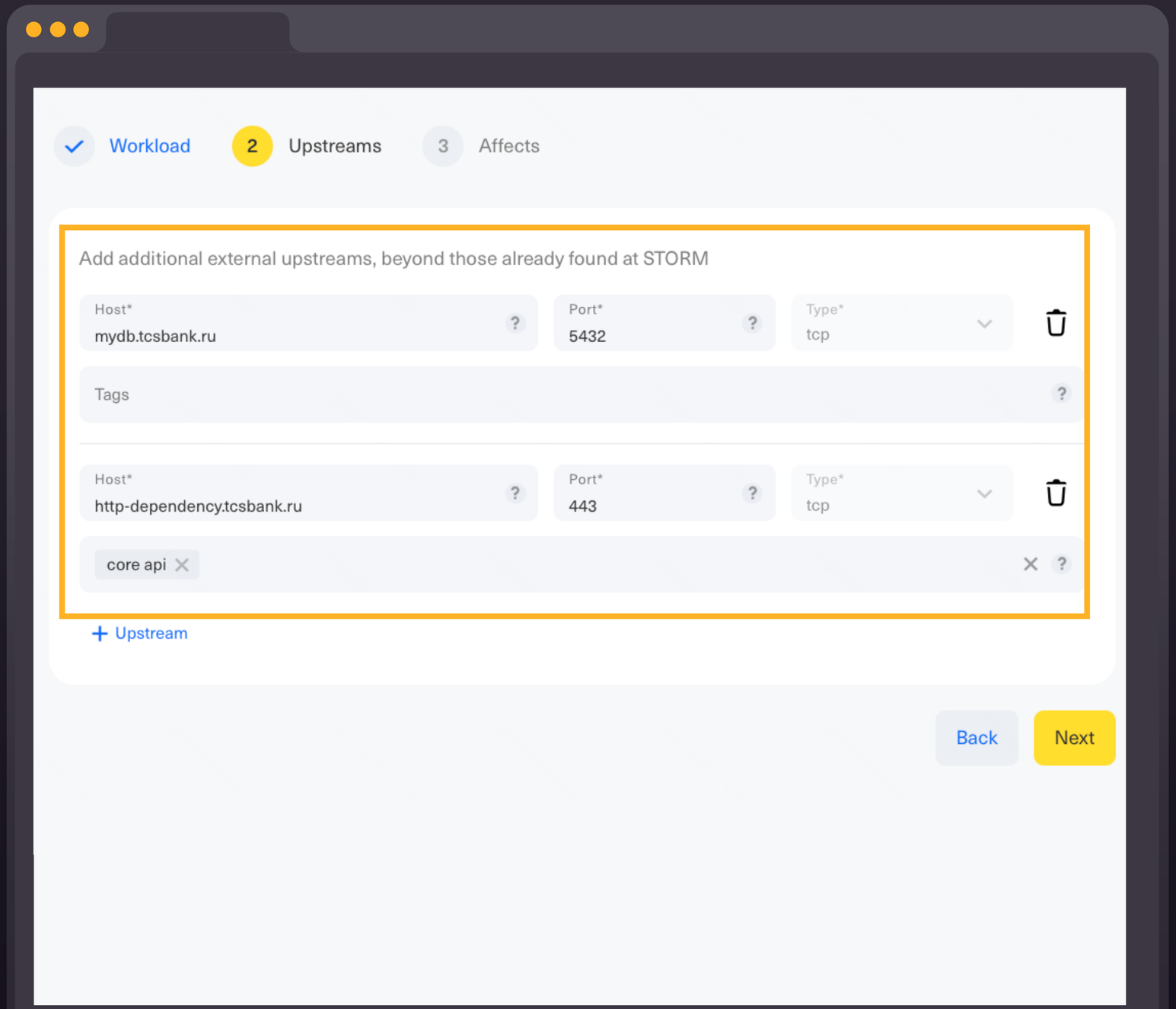
Добавляем Envoy прокси



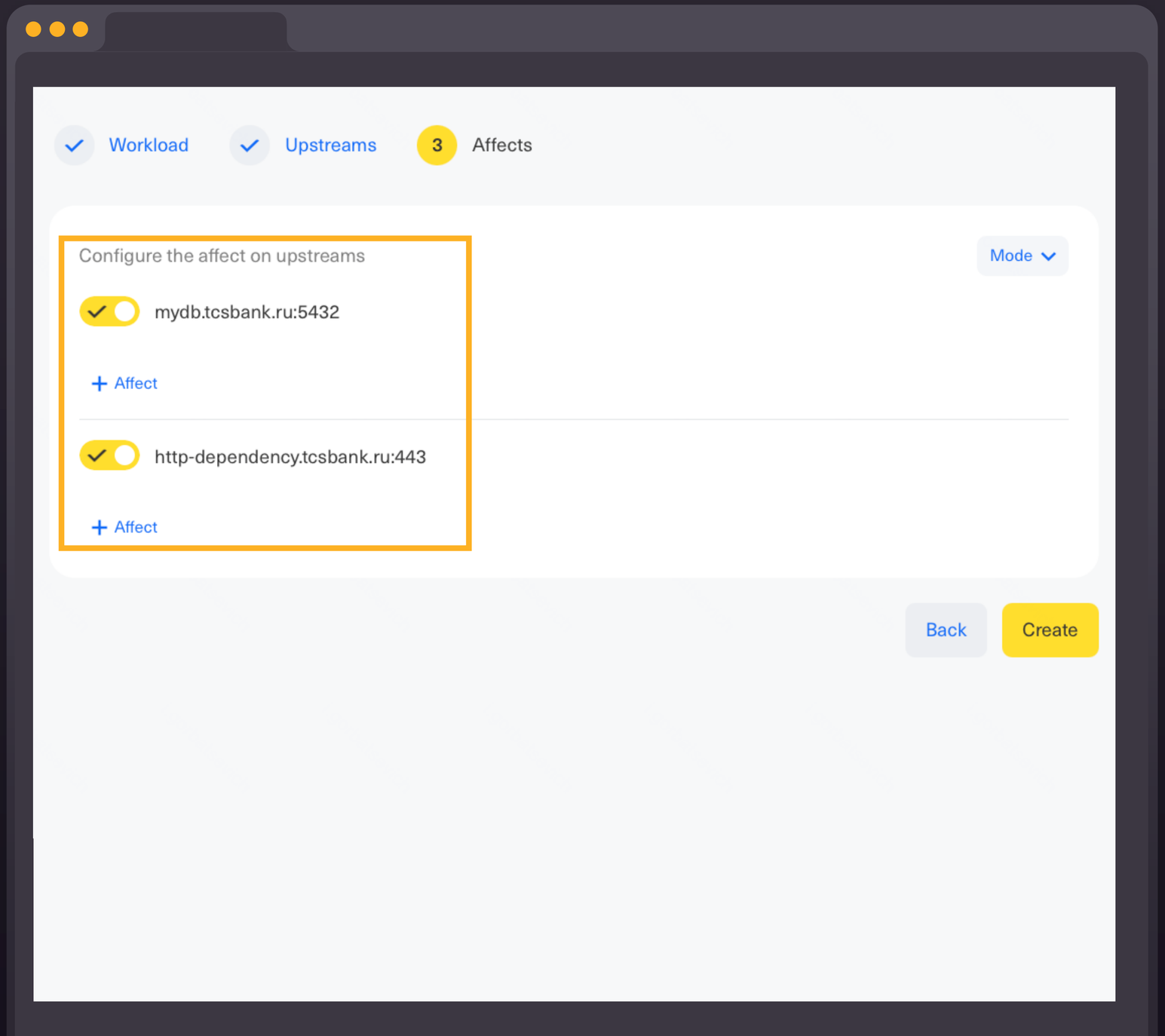
Запуск эксперимента в UI



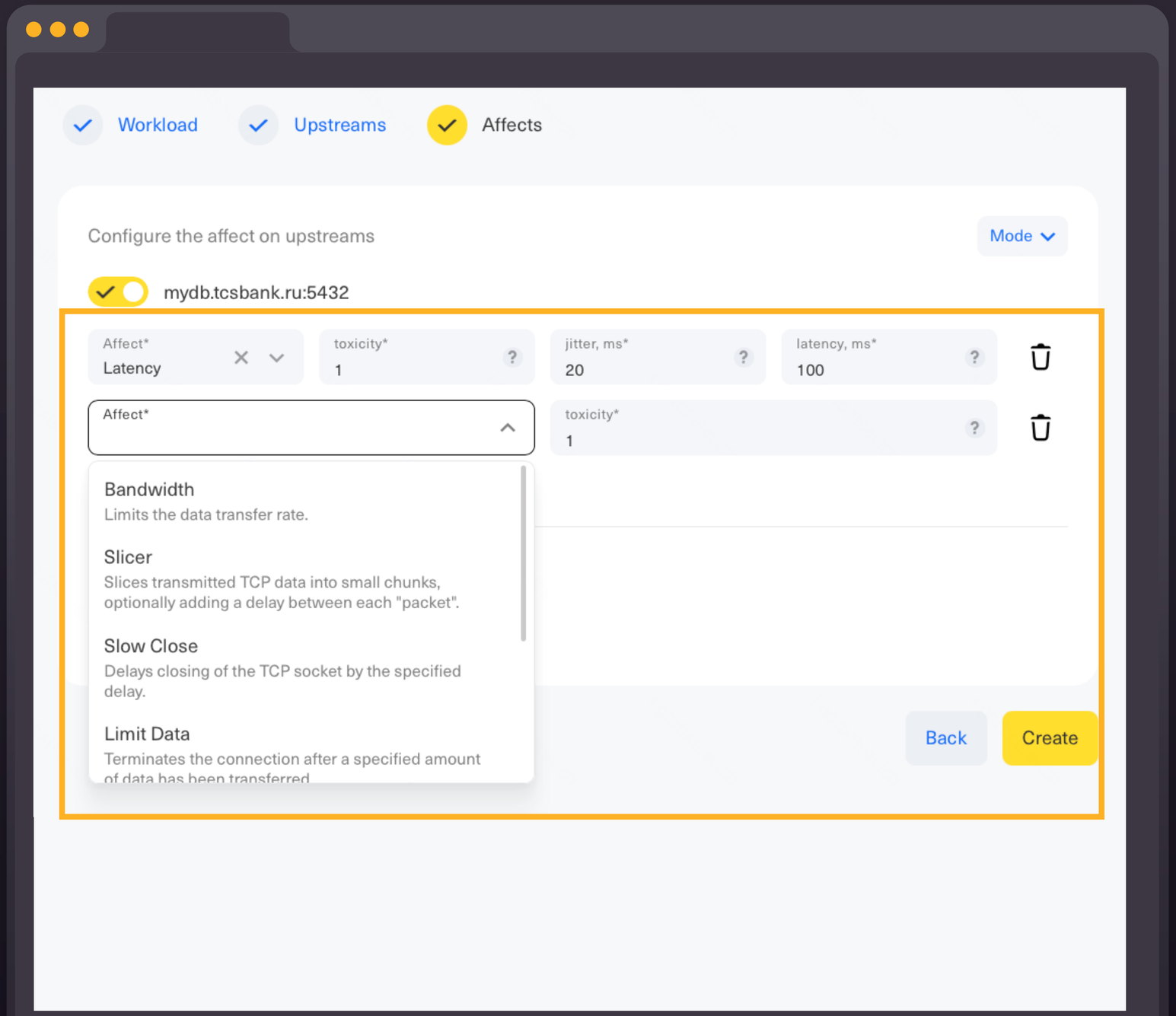
Запуск эксперимента в UI



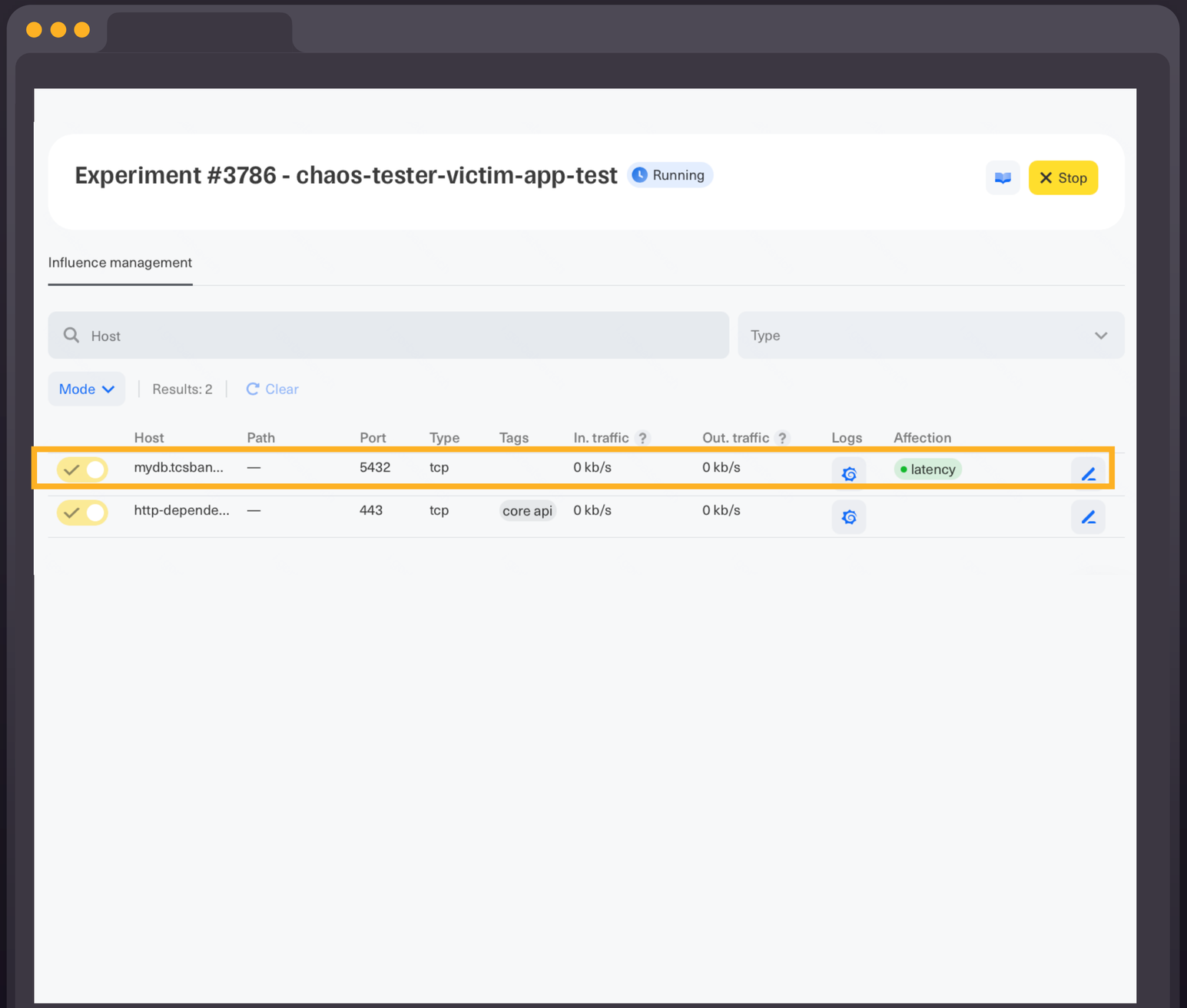
Запуск эксперимента в UI



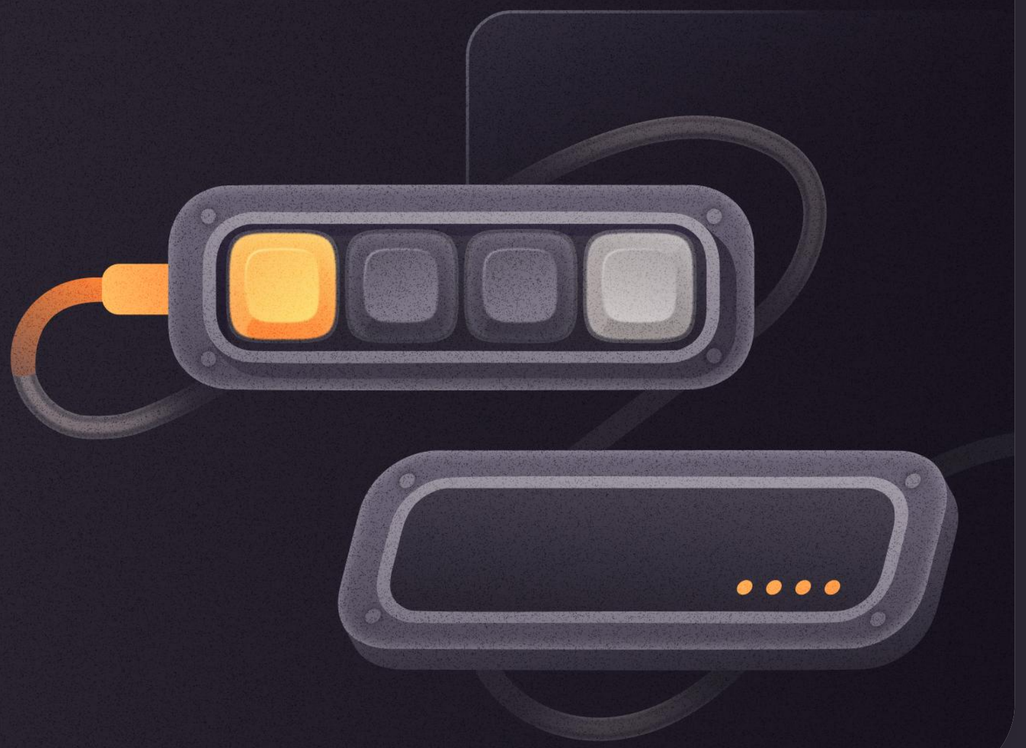
Запуск эксперимента в UI



Запуск эксперимента в UI



Что дает хаос на тесте?



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

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

Результаты и рекомендации

Результаты

324 сервиса протестировано

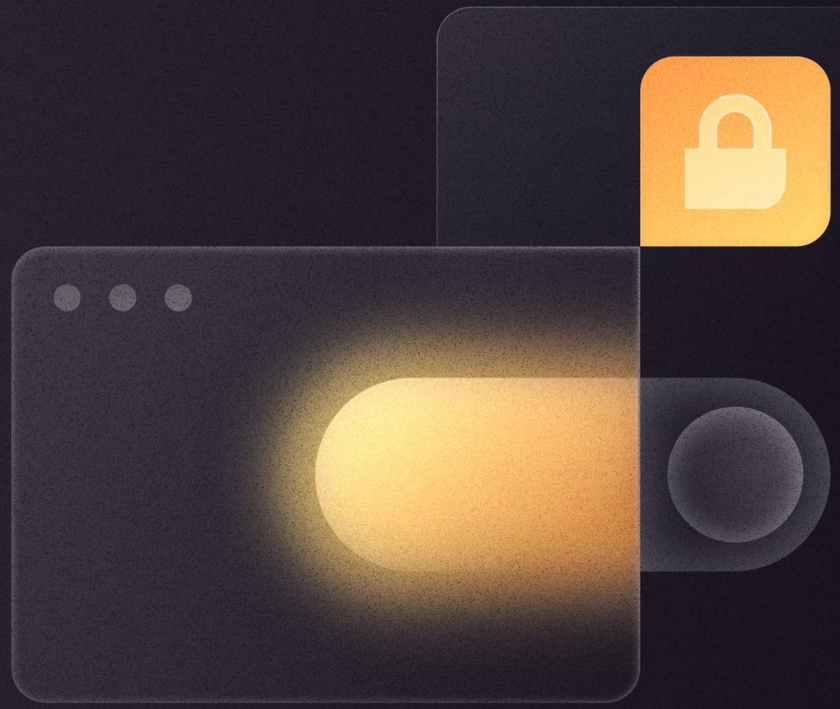
60 сервисов протестировано в продакшене

5427 экспериментов всего

370 экспериментов в месяц

В 11% экспериментов удалось найти проблемы

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**

Кому нужно применять практику?

Частые сбои на продакшене

Много интеграций

Хотите двигать тестирование вправо

Рекомендации к применению

Понять, нужна ли вам практика

Выделить ресурсы на создание инструмента

Создание инструмента

Разработка MVP – 3 месяца

Разработка полноценного инструмента – 6 месяцев

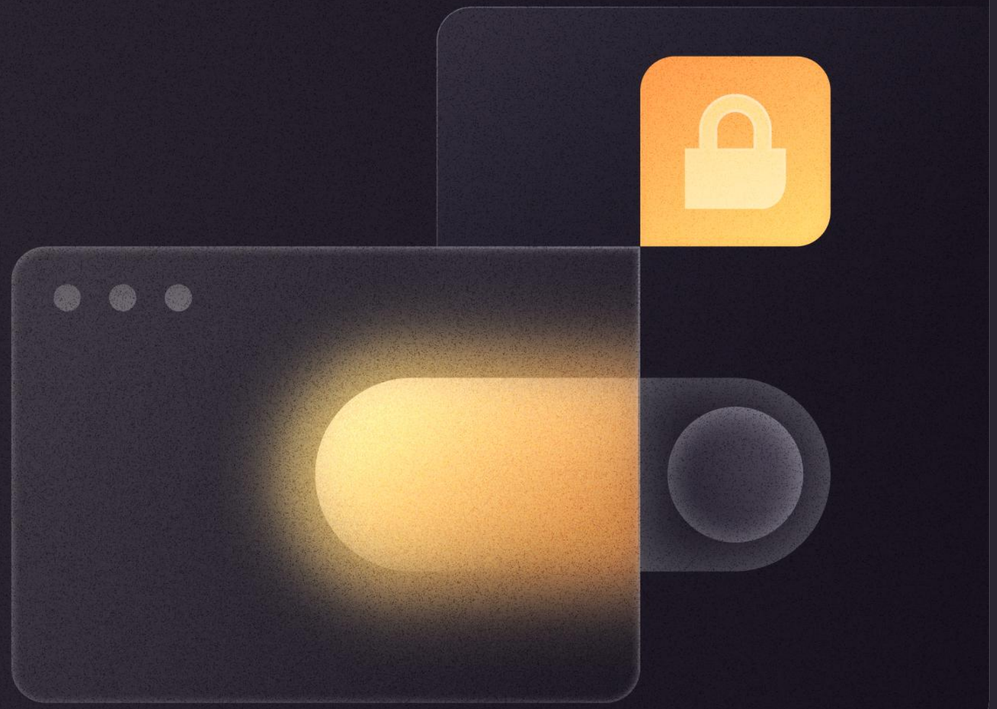
Рекомендации к применению

Понять, нужна ли вам практика

Выделить ресурсы на создание инструмента

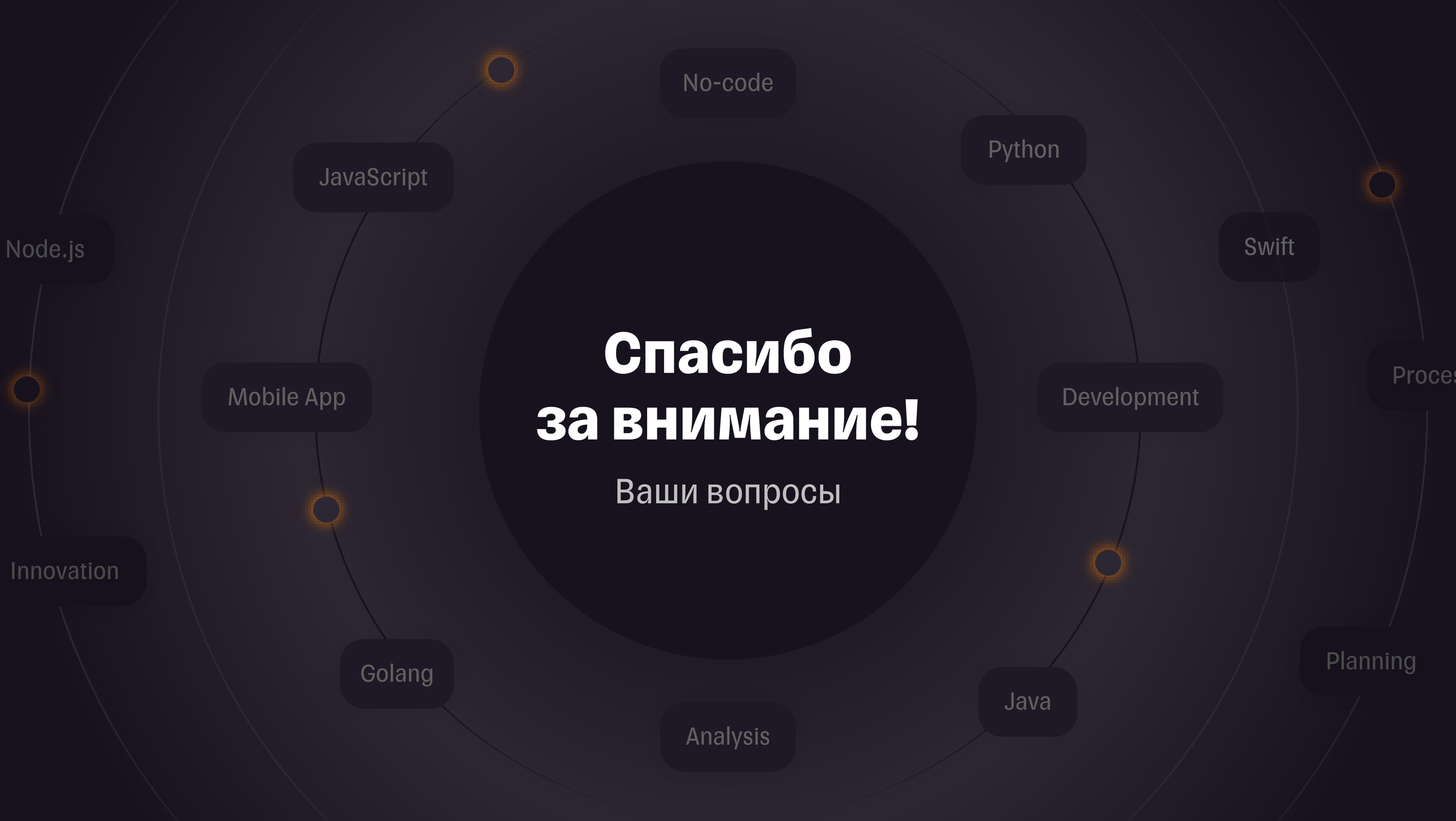
Обучить практике и внедрить в команды

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