



# Health Checking

Павел Селиванов  
SRE

# Health Checks



# Health Check в Kubernetes

- Проверка работоспособности сервиса
- **Liveness probe**: работает
- **Readiness probe**: готов принимать трафик



# Health Check в Kubernetes

- **HTTP GET**: опрашивает URL, ждет ответ 200
- **TCP Socket**: подключается по TCP, ждет соединение
- **Exec**: выполняет команду, ждет код возврата 0





Жив ли наш сервис?



# Что значит «жив»?

Есть соединение с сервером = жив?





# Что значит «жив»?

Есть ответ от произвольного API-запроса = жив?







# Что значит «жив»?

Есть ответ от пустого эндпоинта = жив?

Fifty  
Shades  
of NO





Хороший health check  
отражает состояние сервиса  
и зависимостей, без которых  
работа **НЕВОЗМОЖНА**.

# Подводные камни



# readinessProbe != livenessProbe

- Проверка БД при **readinessProbe**:  
под не начинает принимать трафик без БД
- Проверка БД при **livenessProbe**:  
при перезагрузке БД каскадом валятся все поды



# Exec probes

- Exec-проверки с Kubernetes + Docker могут приводить к появлению зомби-процессов
- Зомби-процессы приводят к невозможности чистой остановки контейнера

# initialDelaySeconds

- Проваленный **livenessProbe** перезагружает под
- Под большой нагрузкой поды могут инициализироваться дольше
- => под будет вечно перезагружаться
- **initialDelaySeconds** = p99 startup time

# Интересные паттерны

# Secondary Health Port

- Дополнительный management-порт
- Требуется рефакторинг приложения
- *Всегда* должен проверять основной порт
- **Пример:** сервис с gRPC на основном порту



# Sidecar Health Server

- Дополнительный контейнер внутри пода
- Опрашивает основное приложение
- **Пример:** health-server опрашивает dovecot

# Headless Probe

- CLI-клиент в контейнере с основным сервисом
- Без изменений кода или доп. контейнера
- **Пример:** redis-cli опрашивает redis



# Hardware Probe

- Контейнер, опрашивающий оборудование
- Для сервисов / задач, которые требуют hardware-ресурсов
- **Пример:** убирать поды с нод, где перегревается GPU

