Engineer

Простій

проміжок часу, коли сервіс нікому не відповідає

Що це таке

Простій — це час, коли сервіс не працює для тих, хто від нього залежить. Його ще називають збоєм, відмовою або недоступністю. Протилежне поняття — час безвідмовної роботи (uptime), а частка часу, коли сервіс працює, — це його доступність, яку зазвичай подають у відсотках за місяць або за рік.

Найпростіший випадок — повна зупинка: сайт не завантажується, API відхиляє з'єднання, база даних лежить. Звичайний випадок заплутаніший. Оформлення замовлення обривається за тайм-аутом, хоча каталог працює; сторінка входу відкривається, але відкидає кожен пароль; API відповідає на один запит із трьох. Для кожного, кому потрібен саме цей шлях, це простій, навіть якщо перевірка справності, спрямована на головну сторінку, світиться зеленим. Корисне визначення рахує те, що можуть зробити користувачі, а не те, чи працює процес.

Простій буває двох видів. Неплановий — це збій, якого ніхто не обирав: диск, невдалий деплой, прострочений сертифікат, залежність, що впала першою. Плановий простій обирають заздалегідь, зазвичай у вікні обслуговування, щоб оновити, мігрувати чи перенести щось, чого не можна змінювати, поки воно обслуговує трафік. З плановим простоєм легше жити, бо користувачів можна попередити, але він однаково зменшує доступність, якщо угода прямо його не виключає.

Для самої цієї миті HTTP має власний словник. RFC 9110 відводить клас кодів стану 5xx для сервера, який знає, що помилився або не здатен виконати запитане. Три з цих кодів під час збою означають різне. 502 Bad Gateway повідомляє, що проксі отримав некоректну відповідь від сервера за ним. 504 Gateway Timeout — що проксі надто довго чекав на відповідь. А 503 Service

Unavailable надсилають свідомо: специфікація пов’язує його з тимчасовим перевантаженням або запланованим обслуговуванням і дозволяє серверу сказати, коли повертатися:

```
HTTP/1.1 503 Service Unavailable
Retry-After: 120
```

Так браузери, API-клієнти й пошукові роботи дізнаються, що стан тимчасовий, і отримують пораду зачекати дві хвилини. Це зовсім інше повідомлення, ніж зламана сторінка, віддана з кодом успіху.

Застосуйте поняття до рішення

Застосуйте «Що таке простій?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Моніторинг і сповіщення, які попереджають, поки збій ще малий, і цифри, що показують, як часто так і залишається.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Запустила перший дашборд спостережуваності — перший огляд власних систем компанії

Запустила перший дашборд спостережуваності Зв'язатися з нами →

Чому це важливо

Цілі доступності записують дев’ятками, а дев’ятки можна перевести в час. Для року з 365 днів, тобто 8 760 годин, кожна ціль дозволяє стільки простою:

- 99%: 87,6 години на рік, близько трьох з половиною діб;
- 99,9%: 8,76 години на рік, або 43,2 хвилини за 30-денний місяць;
- 99,95%: 4,38 години на рік, або 21,6 хвилини за 30-денний місяць;
- 99,99%: 52,56 хвилини на рік;
- 99,999%: близько 5,26 хвилини на рік.

Кожна додаткова дев’ятка зменшує запас удесятеро, а ціна її досягнення стрімко зростає. За доступності 99,9% людина встигає прокинутися, увійти в систему, знайти проблему й виправити її в межах місячного бюджету. За 99,99% на весь рік припадає менше години, чого вистачить хіба що на один інцидент, тож відновлення мусить бути автоматичним. За 99,999% навіть звичайний перезапуск має минати непомітно для користувачів.

Залежності перемножуються. Запит, якому потрібні балансувальник навантаження, застосунок і база даних, кожен із яких доступний 99,9% часу й відмовляє незалежно від інших, обслуговується лише тоді, коли працюють усі три: $0{,}999^3$, тобто близько 99,7%, або приблизно 26 годин простою на рік замість 8,76. Сервіс не може бути доступнішим за ланцюжок того, від чого він залежить, хіба що прибере ланки з ланцюжка або продублює їх.

Планування починається з бізнесу. Посібник NIST із планування безперервності для федеральних систем США, SP 800-34, називає три числа, які варто узгодити до першої аварії. Максимально допустимий простій (maximum tolerable downtime, MTD) — це загальна тривалість збою, яку готовий прийняти власник системи. Цільовий час відновлення (recovery time objective, RTO) — як довго система може лишатися недоступною, перш ніж наслідки стануть неприйнятними; він має бути коротшим за MTD. Цільова точка відновлення (recovery point objective, RPO) — момент у минулому, до якого мають бути відновлені дані з останньої резервної копії, тож вона вимірює допустиму втрату даних, а не тривалість простою. Коли ці три числа записано, розклад резервного копіювання й схема перемикання на резерв стають відповіддю на запитання, а не здогадками.

Як це виглядає на практиці

На IPTV- та OTT-платформі з клієнтами в Китаї, США та Німеччині кожна година була для когось праймтаймом, тож тихої години для обслуговування просто не існувало. Цілодобова підтримка охоплювала близько тисячі серверів, а також чимало систем, що належали самим клієнтам. Значна частина роботи полягала в тому, щоб помітити відхилення до того, як воно переросте в збій, бо в прямій трансляції простій — це чийсь телевізор, що перестав показувати. Платформа лишалася безперервно доступною для глобальної аудиторії.

У компанії середнього розміру десять серверів баз даних PostgreSQL і SQL Server тримали 99,9% безвідмовної роботи, що за розрахунком вище означає запас 8,76 години на рік для кожного. За цим числом стояли автоматичне резервне копіювання через `pg_dump` і завдання SQL Server Agent із політиками збереження та офсайт-зберіганням, моніторинг у Prometheus і Grafana, а також задокументовані процедури аварійного відновлення, зокрема відновлення на конкретний момент часу й перемикання на резерв.

А на парку з тридцяти серверів Ubuntu, чиє налаштування радше накопичувалося, ніж проєктувалося, аварійне відновлення перебудували на резервних копіях і процедурах відновлення, які справді перевірялися. Мережеву конфігурацію звели до одного захищеного базового рівня на всіх інстансах. Ризик простою знизився, а шляхи відновлення були перевірені ще до того, як комусь знадобилися.

Де виникають проблеми

Моніторинг сидить усередині збою. Перевірка справності, що працює на тому самому сервері чи в тому самому дата-центрі, і далі світиться зеленим, поки користувачі натрапляють на прострочений сертифікат TLS, зламаний запис DNS або балансувальник, що вказує в порожнечу. Перевіряйте ззовні, з кількох місць, тим шляхом, яким іде справжній користувач, і надсилайте сповіщення за тим, що бачить саме ця перевірка.

Сторінка помилки відповідає кодом 200. Фреймворк, що показує привітну сторінку помилки з кодом успіху, ховає збій від моніторингу, від клієнтів, які могли б повторити запит, і від пошукових роботів, що можуть проіндексувати сторінку помилки замість справжньої. Помилки мають нести код помилки, а на час запланованого обслуговування сервер має відповідати 503 із заголовком Retry-After.

Резервну копію жодного разу не відновлювали. Резервна копія доводить лише те, що байти скопійовано. Доки відновлення не запустили, не заміряли й не звірили з даними, час відновлення залишається припущенням, і часто він значно довший за RTO на папері. Відновлюйте за розкладом, в окреме середовище, і записуйте, скільки це тривало. Те саме стосується перемикання: резервний сервер, який ще ніколи не перебирав навантаження, — це надія, а репетиція перемикання в спокійних умовах допомагає виявити проблеми ще до інциденту.

Релізи спричиняють ті самі збої, яким мали запобігти. Розгортання, що зупиняє стару версію, перш ніж нова готова, залишає прогалину на кожному релізі. Зміна схеми, що переписує велику таблицю під ексклюзивним блокуванням, блокує кожен запит до неї, доки не завершиться. Викочуйте нові інстанси за перевіркою готовності (readiness), перш ніж зупиняти старі; розбивайте зміни схеми на крок розширення (expand) і пізніший крок

звуження (contract), щоб у проміжку працював і старий, і новий код; а в Kubernetes задайте PodDisruptionBudget, щоб виведення вузла з роботи (drain) не могло зупинити всі репліки одночасно.

Що почитати далі

- RFC 9110: HTTP Semantics, section 15.6 — коди стану 5xx, зокрема 503 і його підказка Retry-After.
- NIST SP 800-34 Rev. 1: Contingency Planning Guide for Federal Information Systems — документ, де визначено максимально допустимий простій, RTO і RPO.
- PostgreSQL: High Availability, Load Balancing, and Replication — резервні сервери й перемикання так, як їх описує посібник самої бази даних.
- PostgreSQL: Continuous Archiving and Point-in-Time Recovery — базова резервна копія та архів WAL, на яких тримається відновлення на конкретний момент часу.
- Kubernetes: Disruptions — добровільні переривання та PodDisruptionBudget, що їх обмежують.

Словник

Обирайте за фактичними умовами

Опишіть шлях розгортання, експлуатаційні ризики та чергову команду. Ми можемо визначити зміну з перевірками, шляхом відновлення та чіткою відповідальністю.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке простій?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/downtime/