Engineer

Тривога

повідомлення, яке система шле, коли потрібна людина

Що це таке

Тривога — це автоматичний сигнал про те, що умова, за якою стежить система, трималася достатньо довго, щоб заради неї варто було відірвати людину від справ. В українських командах часто кажуть просто «алерт», а тривогу, через яку будять чергового серед ночі, англійською називають page.

За цим словом стоять два різні об’єкти, і Prometheus розводить їх настільки чітко, що різницю легко помітити. Правило тривоги в Prometheus породжує тривоги, по одній на кожен окремий набір міток, для якого вираз повертає результат. Далі вони надходять до окремої програми, Alertmanager, а та перетворює їх на сповіщення: лист, повідомлення в чаті, дзвінок через сервіс чергувань. Сам проєкт прямо пише, що правила тривог самі по собі ще не є повноцінною системою сповіщень. Розрізняти ці дві речі доводиться тому, що їхня кількість розходиться. П’ятдесят тривог можуть прийти одним листом, одна тривога може нагадувати про себе кожні кілька годин, а тривога, якій вимкнули звук, до людей не доходить, хоча й далі спрацьовує.

Життєвий цикл у тривоги короткий. Поки вираз нічого не повертає, тривога неактивна. Щойно він уперше дає результат, тривога переходить у стан очікування (pending) і лишається в ньому стільки, скільки вимагає параметр `for` у правилі. Якщо наприкінці цього проміжку умова все ще виконується, тривога спрацьовує (firing) і йде далі на доставку. Коли умова перестає виконуватися, тривога завершується (resolved). Новіший параметр `keep_firing_for` утримує тривогу в стані спрацювання ще деякий час після того, як умова зникла; документація пояснює його як захист від мерехтіння та хибних повідомлень про те, що все вже гаразд. Крім того, Prometheus записує стани очікування й спрацювання як часовий ряд `ALERTS`, тож до самих тривог можна робити запити й показувати їх на графіках.

Правило, яке чекає двадцять хвилин, перш ніж спрацювати, і ще десять, перш ніж відпустити, має такий вигляд:

```
groups:
    - name: exports
      rules:
          - alert: ExportQueueStalled
            expr: exports_waiting > 0 and rate(exports_done_total[15m]) == 0
            for: 20m
            keep_firing_for: 10m
            labels:
                severity: page
                team: billing
            annotations:
                summary: "{{ $value }} exports queued and none finishing"
```

Мітки й анотації схожі на вигляд, але виконують різну роботу. Мітки входять до того, що робить тривогу саме цією тривогою, і Alertmanager читає їх, щоб вирішити, куди її надіслати: `severity: page` може потрапити на телефон, а `severity: ticket` — у чергу заявок. Анотації несуть слова, потрібні людині, коли повідомлення вже надійшло: короткий підсумок, довший опис, посилання на інструкцію з першими кроками. Анотації можна складати з шаблонів, що підставляють мітки тривоги та значення, яке її викликало, і саме тому підсумок у прикладі вище здатен назвати число.

Тривога й інцидент — різні поняття. Тривога повідомляє, що умова справдилася, а чи є те, що відбувається, інцидентом, вирішує людина.

Застосуйте поняття до рішення

Застосуйте «Що таке тривога?» 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

Тривога — єдина частина моніторингу, яка не чекає, поки на неї подивляться. Метрика лежить у сховищі, дашборд висить на екрані, журнал чекає на пошуковий запит, а тривога вривається в чийсь вечір. Тож кожне правило тривоги — це невелика постійна заявка на чиюсь увагу, і вона справедлива лише тоді, коли людина може щось із цим зробити. Настанови Prometheus щодо тривог радять не будити людей, коли робити нічого, і тримати систему тривог нескладною.

Друге — звичайна арифметика. Одна несправність зазвичай зачіпає багато правил водночас. Якщо через розділення мережі половина екземплярів сервісу втрачає зв'язок із базою даних, кожен екземпляр може надіслати власну тривогу, і документація Alertmanager саме на цьому прикладі показує, як сотні тривог стають одним сповіщенням. Alertmanager збирає разом тривоги зі спільними мітками, переліченими в `group_by`, чекає `group_wait` (типово 30 секунд), щоб перше повідомлення вмістило всю групу, надсилає оновлення для групи не частіше ніж раз на `group_interval` (5 хвилин) і повторює повідомлення про тривогу, що досі спрацьовує, через `repeat_interval` (4 години). Пригнічення йде ще далі й приглушує одні тривоги, поки спрацьовує інша, тож тривога про недоступність усього кластера може заглушити тривоги окремих сервісів у ньому. Silence, як це називає Alertmanager, вимикає звук для тривог із певними мітками на визначений час, зазвичай на період планових робіт.

Третє — кінець історії. Більшість каналів сповіщення надсилає повідомлення про завершення тривоги, але для поштового каналу параметр `send_resolved` типово має значення false. Команда, яка отримує тривоги поштою, може побачити, як проблема почалася, і так і не дізнатися, коли вона скінчилася.

Як це виглядає на практиці

Власний хост компанії використовує тривогу за відсутності сигналу. Він має 464 МБ пам'яті й обслуговує вебсервер і git-сервер, а вичерпання пам'яті та заповнення диска — два важливі ризики відмови. Таймер на хості пінгує зовнішній сервіс лише тоді, коли перевірки доступної пам'яті й вільного місця на диску проходять за встановленими порогами кожні п'ятнадцять хвилин із випадковим зсувом, а зовнішній сервіс надсилає тривогу, коли очікувані пінги перестають надходити. Відмови, що не дають скрипту виконатися чи з'єднатися із цим сервісом, зокрема втрата мережі або невдале завантаження, спричиняють відсутність пінгу. Отриманий сигнал

підтверджує, що скрипт виконався, а перевірки доступної пам’яті й вільного місця на диску пройдено за встановленими порогами; він не засвідчує загальної справності хоста чи того, що сайт обслуговує запити. Моніторинг може виявляти тривожні ознаки, хоча час їх появи залежить від навантаження й умов спостереження.

Те саме обернення може захищати заплановані завдання. У реалізації резервного копіювання сигнали успіху для копіювання та відновлення розділено. Скрипти й модулі systemd підтримують зашифровані знімки поза хостом, перевірки цілісності та щомісячний тест відновлення відомого файла; сигнал успіху надсилається лише після успішного завершення відповідного процесу. Це схема сповіщення, а не доказ того, що вона була активною. У записі про реалізацію від 2026-10-27 зазначено, що таймери були вимкнені без налаштованого сховища, і не зафіксовано жодного завершеного відновлення в реальних умовах. Відсутність сигналу може викликати зовнішнє сповіщення після активації процесу та його моніторингу; сама конфігурація не доводить, що вони працюють.

Інвертовані тривоги мають власний спосіб зламатися, і він проявився, коли платформа мала вирости з одного хоста до двох. Якби обидва хости ділили один перемикач, це послабило б саме ту систему тривог, яку мали розширити: поки пінгує хоча б один хост, перевірка лишається зеленою для обох, навіть коли другий уже мертвий. Кожен перемикач став окремим значенням для свого хоста, а поля перемикача для наявного хоста свідомо залишили порожніми, тож конфігурація показує, що з’єднання передбачене, а значення ще треба вписати.

Де виникають проблеми

Виміряне значення потрапляє в мітку. Оскільки мітки визначають тривогу, мітка з поточним числом перетворює кожну зміну цього числа на нову тривогу й завершує попередню. Отримувач бачить потік пар «спрацювала — завершилася» через одну проблему. Залишайте мітки для ідентичності й маршрутизації, а числа виносьте в анотації.

Тривога мерехтить. Умова, що коливається біля порогу, спрацьовує, завершується й знову спрацьовує кожні кілька хвилин, і кожне повідомлення про завершення каже читачеві неправду. Подовжте `for`, щоб короткий сплеск минав раніше, ніж про нього хтось дізнається, і додайте `keep_firing_for`, щоб коротке падіння нижче порогу не закривало тривогу.

Silence живе довше за свою причину. Хтось вимикає звук тривоги на час робіт із надто широкою умовою та щедрим часом завершення, і справжній збій наступного тижня надходить беззвучно. Обмежуйте silence так вузько, як дозволяє робота, ставте кінець близько до того часу, коли роботу мають завершити, і пишіть у полі коментаря, хто його створив і навіщо.

Широкий маршрут поглинає вужчі. Alertmanager обходить дерево маршрутизації по черзі і, якщо маршрут не має `continue: true`, зупиняється на першому дочірньому маршруті, що підходить. Загальний маршрут, поставлений над маршрутом окремої команди, забирає тривоги цієї команди з собою. Ставте найвужчі маршрути першими й перевіряйте дерево на прикладах міток, перш ніж на нього покладатися.

Усе стає терміновим. Диск, заповнений на 70%, сертифікат, що спливає наступного місяця, і платіжний сервіс, який повертає помилки, дзвонять на той самий телефон, і незабаром на телефон перестають зважати. Будіть людей лише через умови, які вимагають дій негайно, решту надсилайте в чергу заявок чи щоденне зведення, а правило, яке місяцями спрацьовує без жодної реакції, видаляйте.

Що почитати далі

- Alerting overview — як правила тривог у Prometheus і Alertmanager ділять між собою роботу та з яких кроків складається їхнє налаштування.
- Alerting rules — синтаксис правил, зокрема `for`, `keep_firing_for`, мітки, анотації та шаблони.
- Alertmanager — групування, пригнічення, silences і робота кількох екземплярів Alertmanager для високої доступності.
- Alertmanager configuration — дерево маршрутизації, таймери групування, правила пригнічення й налаштування кожного отримувача, серед них `send_resolved`.
- Alerting practices — власні поради проєкту про те, заради чого варто будити людей, а що має йти іншими каналами.

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 тривога?» 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/alert/