Engineer

cron

планувальник Unix, що запускає команди за розкладом crontab

Що це таке

cron — це служба Unix, яка запускає команди в заданий час, і для цього нікому не треба входити в систему. Розклад кожного користувача зберігається в crontab: текстовій таблиці, де один рядок — одне завдання. Те саме слово називає команду `crontab`, яка встановлює, показує та видаляє цю таблицю.

POSIX стандартизує команду й формат таблиці, але не програму, яка ці таблиці читає. В обґрунтуванні стандарту прямо сказано, що всі згадки про демон cron і файли cron з нього прибрано. Який демон виконує завдання, вирішує кожна система. Сторінки довідки про cron, на які спирається ця стаття, взято з cronie: його crontab(5) називає автором Пола Вікси (Paul Vixie) і описує формат як стандарт V7 з низкою розширень, що зберігають сумісність.

Рядок має шість полів, розділених пробілами або табуляціями. Перші п'ять кажуть, коли: хвилина (від 0 до 59), година (від 0 до 23), день місяця (від 1 до 31), місяць (від 1 до 12) і день тижня (від 0 до 6, де 0 — неділя). Кожне поле — це зірочка, що означає будь-яке значення, число, діапазон через дефіс, як-от `1-5`, або список таких елементів через кому. Шосте поле — команда, яку виконує `sh`. На цьому стандарт закінчується. Кроки на кшталт `*/15`, трилітерні назви на кшталт `mon`, `7` для неділі, стовпець з ім'ям користувача в системних таблицях і скорочення на кшталт `@daily` — це розширення cronie; таблиця, яка ними користується, вже не переносна.

`crontab -e` відкриває вашу таблицю в редакторі, `crontab -l` виводить її, `crontab -r` видаляє, а `crontab файл` замінює її вмістом файлу. Крім того, cronie має системні таблиці в `/etc/crontab` і `/etc/cron.d/`, де між часом і

командою стоїть додаткове поле з ім’ям користувача, від якого запускається завдання. POSIX це передбачав: в обґрунтуванні зазначено, що системи, похідні від BSD, найімовірніше додадуть поле користувача.

Застосуйте поняття до рішення

Застосуйте «Що таке cron?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Щоденний догляд за серверами, поштою, обліковими записами й оновленнями, щоб те, на що всі покладаються, працювало тихо й без сюрпризів.

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. Підтримувала 10 серверів PostgreSQL і MS SQL — 10 серверів баз даних розгорнуто й підтримувано в справному стані

Підтримувала 10 серверів PostgreSQL і MS SQL Зв'язатися з нами →

Чому це важливо

Чимало обслуговування сервера відбувається, коли ніхто не дивиться: старі файли прибирають, звіти розсилають поштою, бази даних вивантажують у дамп, роблять резервні копії. cron — стандартний спосіб сказати «запусти це о третій ночі», і ця утиліта є в стандарті ще з Issue 2, як засвідчує історія змін її сторінки в POSIX. Таблиця, що обмежується полями POSIX, читається однаково на будь-якій системі, яка відповідає стандарту.

Слабкість закладено в саму ідею. Заплановане завдання не має глядачів, тож коли воно ламається, жоден екран не червоніє. POSIX каже, що вивід і помилки, які команда не перенаправила, надсилаються користувачеві поштою, а як саме, вирішує кожна реалізація. На хості, де локальна пошта нікуди не доходить, туди ж зникає й помилка. Більшість проблем із cron зводиться до цієї тиші.

Тепер цю роботу ділять двоє сусідів. У Linux-системах під керуванням systemd те саме завдання може запускати таймер-юніт, і він додає те, чого бракує cron: уміє надолужити запуск, пропущений, поки машина була вимкнена, розносить час старту випадковим чином і ніколи не запускає другу копію служби, яка ще працює. Усередині PostgreSQL розширення `pg_cron` читає ті самі розклади з п’ятьма полями, а фоновий процес у базі даних відстежує завдання. Наш проєкт із чергою завдань River — приклад: у жовтні 2026 року його сторінка налічує 20 завдань `pg_cron` під чергою, які чистять партиції, змінюють солі та оновлюють агрегати, тобто обслуговування, що, за словами сторінки, має жити в базі даних.

Як це виглядає на практиці

Візьмімо два завдання: прибирання в будні й нічний дамп бази даних. Спершу таблиця задає власне середовище, бо завдання не успадковує налаштувань вашої оболонки входу. Рядки `NAME=value` і змінна `MAILTO` — це можливості cronie, а не POSIX.

```
SHELL=/bin/sh
PATH=/usr/local/bin:/usr/bin:/bin
MAILTO=ops@example.com
# хвилина година день місяць день-тижня команда
15 3 * * 1-5 find "$HOME" -name core -exec rm -f {} +
30 2 * * * flock -n /var/lock/dump-db dump-db >> "$HOME/dump.log" 2>&1
0 * * * * date +\%s >> "$HOME/hourly.log"
```

Перше завдання читається як «о 03:15, з понеділка по п’ятницю». Друге запускається щоночі о 02:30, пише свій вивід і помилки в лог-файл і загорнуте у `flock -n`, яке одразу здається замість чекати, якщо блокування вже взято, тож дамп, що затягнувся, ніколи не отримає двійника. Третє показує екранований знак відсотка: без зворотної скісної риски cron обрізав би команду на `%`.

Тримайте таблицю у файлі під контролем версій і встановлюйте її цілком, а не редагуйте наживо:

```
crontab jobs.cron
crontab -l
```

Друга команда показує, що саме читатиме cron. Опція `-T` у cronie перевіряє синтаксис файлу ще до встановлення, а сам файл має закінчуватися символом нового рядка.

Той самий нічний дамп у вигляді таймера systemd потребує двох юнітів. Служба каже, що запускати, а таймер — коли:

```
# /etc/systemd/system/dump-db.service
[Service]
Type=oneshot
ExecStart=/usr/local/bin/dump-db
```

```
# /etc/systemd/system/dump-db.timer
[Timer]
OnCalendar=*-*-* 02:30:00
Persistent=true
RandomizedDelaySec=10min

[Install]
WantedBy=timers.target
```

`Persistent=true` зберігає на диску час останнього запуску й запускає пропущений, щойно таймер знову активується. `RandomizedDelaySec=` відкладає старт на випадковий проміжок до десяти хвилин, тож парк машин з однаковим файлом не вдарить по серверу бази даних в одну й ту саму секунду. `systemd-analyze calendar '*-*-* 02:30:00'` виводить нормалізований вираз і час наступного спрацювання; це найшвидший спосіб перевірити розклад, перш ніж йому довіряти.

Розклад каже, коли завдання стартує, а не чи виконало воно свою роботу. На одному з наших хостів із 464 МБ dead man’s switch перевертає це: таймер чотири рази на годину, щоразу з випадковим відступом у часі, запускає скрипт, а той надсилає пінг сервісу ззовні лише тоді, коли ні пам’ять, ні диск не опустилися нижче своєї межі. Коли пінги припиняються, спрацьовує тривога.

Де виникають проблеми

Команда працює у вашій оболонці, але не в cron. POSIX гарантує завданню лише `HOME`, `LOGNAME`, `PATH` і `SHELL`, а `PATH` має знаходити тільки стандартні утиліти. Нічого з вашого профілю не завантажується. Вихід — абсолютні шляхи або `PATH`, заданий на початку таблиці, і перевірка завдання в такому самому скупому середовищі.

Помилки зникають. Неперенаправлений вивід надсилається поштою. cronie шле його власникові таблиці, на адресу з `MAILTO` або нікуди, якщо `MAILTO=""`. Вихід — перенаправляти вивід у лог, який хтось читає, а для важливих завдань ще й повідомляти про успіх тому, хто підніме тривогу, коли повідомлення припиняться, як це робить dead man’s switch вище.

Знак відсотка розрізає команду. У полі команди `%` стає символом нового рядка, виконується лише перший рядок, а решта йде йому на стандартний ввід, тож `date +%s` ламається. Вихід — екранувати знак як `\%` або винести команду в скрипт і викликати його.

Два поля дня означають «або», а не «і». Коли обмежено і день місяця, і день тижня, завдання запускається в будь-який день, що відповідає хоча б одному з них. Власний приклад POSIX, `0 0 1,15 * 1`, спрацьовує першого й п'ятнадцятого числа, а ще щопонеділка. Вихід — обмежувати лише одне з двох полів або взяти календар systemd, де `Mon *-*-01..07 09:00` збігається лише тоді, коли підходять і день тижня, і дата: це перший понеділок місяця.

Кроки починаються заново щогодини. cronie обчислює крок лише в межах його власного поля, тож `*/35` у хвилинах спрацьовує на нульовій і тридцять п'ятій хвилині кожної години, а не кожні 35 хвилин, а `*/23` у годинах — опівночі та о 23:00. Вихід — крок, на який діапазон ділиться без остачі, або таймер systemd з `OnUnitActiveSec=`, який рахує від останнього запуску служби.

Годинник переводять. cronie документує, що час, який зникає під час переходу на літній час, ніколи не збігається, тож ці завдання не виконуються, а час, що повторюється, коли годинник переводять назад, змушує завдання запуститися двічі. Вихід — планувати завдання поза годинами, коли ваш часовий пояс зсувається, або задати `CRON_TZ` з поясом без літнього часу.

Останній рядок не має символу нового рядка. cronie очікує, що кожен запис закінчується символом нового рядка; якщо останній запис без нього, таблицю вважають щонайменше частково пошкодженою, а в syslog іде попередження. Вихід — завершальний символ нового рядка, який гарантує налаштування редактора або перевірка під час рев'ю.

Запуски накладаються. Формат crontab не має поля для того, що робити, коли попередній запуск ще триває. Таймер systemd не чіпає службу, яка ще працює. Якщо ж наступний час спрацювання настав, поки служба ще працювала, і `DeferReactivation=` не задано, таймер запускає її знову, щойно вона завершиться; `pg_cron` змушує другий запуск того самого завдання чекати, доки завершиться перший. Вихід для звичайного cron — блокування, як-от `flock -n` у прикладі вище.

Машина була вимкнена. Рядок crontab називає момент, і формат не вміє сказати «запусти пізніше, якщо цей момент минув». Вихід — `Persistent=true` на таймері з `OnCalendar=` , якщо машиною керує systemd.

Усі машини водночас. Таблиця, скопійована на сотню хостів, запускає сотню завдань в одну хвилину. Вихід — `RandomizedDelaySec=` на таймері systemd, а в cronie — змінна `RANDOM_DELAY` або діапазон із тильдою, як-от `6~15` , що вибирає одну випадкову хвилину під час читання таблиці.

Голий `crontab` стирає таблицю. Без аргументу команда чекає нову таблицю на стандартному вводі, і кінець файлу (зазвичай Ctrl-D) замінює поточну порожньою. Вихід — натиснути натомість клавішу переривання і тримати таблицю у файлі, який ви встановлюєте через `crontab файл` .

Що почитати далі

- crontab, POSIX.1-2024 — шість полів стандарту, типове середовище, правило пошти й обґрунтування.
- crontab(5), cronie — кроки, `MAILTO` , `CRON_TZ` , `RANDOM_DELAY` , літній час і застереження.
- systemd.timer(5) — `OnCalendar=` , `Persistent=` , `RandomizedDelaySec=` і запущені служби.
- systemd.time(7) — синтаксис календарних подій, скорочення і як їх перевірити.
- flock(1) — блокування навколо команди, з `-n` , щоб здатися, коли його вже взято.
- pg_cron — розклади cron для SQL-завдань усередині PostgreSQL як розширення.

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 cron?» 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/cron/