Engineer

файлова система

структура, що перетворює диск на іменовані файли й каталоги

Що це таке

Файлова система — це частина комп'ютера, яка перетворює диск, що сам по собі зберігає лише пронумеровані блоки байтів, на іменовані файли, впорядковані в каталоги. Англійською її пишуть і двома словами, file system, і одним, filesystem; друге написання здебільшого вживає документація ядра Linux. POSIX, стандарт, якого дотримуються Unix-подібні системи, у редакції 2024 року визначає файлову систему як сукупність файлів і певних їхніх атрибутів. Те саме визначення додає, що файлова система задає простір імен для порядкових номерів файлів (file serial numbers), тобто номерів, які однозначно ідентифікують кожен файл у межах файлової системи, де він лежить.

Решта випливає з кількох інших визначень стандарту. Файл — це об'єкт, у який можна записувати, з якого можна читати або і те, й інше, з атрибутами на кшталт прав доступу і типу: звичайний файл, каталог, символьне посилання, FIFO, сокет, символьний або блоковий спеціальний файл. Каталог — це файл, що складається із записів каталогу, і жодні два записи в одному каталозі не мають однакового імені. Запис каталогу, який POSIX також називає жорстким посиланням (hard link), пов'язує ім'я файлу з файлом, і на один файл можуть вказувати кілька записів. Отже, ім'я файлу і сам файл — різні речі. У Linux запис про сам файл називають індексним дескриптором, inode; посібник GNU до df описує його як місце, де зберігаються власник файлу, права доступу, часові позначки й розташування у файловій системі, тоді як імена живуть у каталогах, що на нього вказують.

Unix-подібна система показує всі свої файлові системи як одне дерево імен. Перша дає кореневий каталог, а кожну наступну приєднують до якогось каталогу дерева, її точки монтування (mount point), після чого шляхи під цим каталогом належать приєднаній файловій системі. Усередині ядра Linux між програмами й численними файловими системами, які воно підтримує, стоїть

віртуальна файлова система, VFS. Документація ядра описує її (вона ж Virtual Filesystem Switch) як шар, що дає програмам інтерфейс до файлових систем і дозволяє різним реалізаціям співіснувати. Програма викликає `open(2)`, `stat(2)`, `read(2)` чи `write(2)`, а VFS розбирає шлях і передає роботу тій файловій системі, де лежить файл: скажімо, ext4 на розділі диска або псевдофайловій системі, чиї inode існують лише в пам'яті. Типи, які зареєструвало запущене ядро, перелічено в `/proc/filesystems`.

Застосуйте поняття до рішення

Застосуйте «Що таке файлова система?» 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

Майже все, що сервер має пам'ятати після наступного перезавантаження, лежить у файловій системі: файли даних його баз даних, логи, завантажені документи, конфігурація. Чи переживуть вони вимкнення живлення, заповнений диск або накопичувач, що відмовляє, значною мірою вирішують рішення, ухвалені під час створення й монтування файлової системи, і деякі з них згодом уже не змінити.

Створення фіксує місткість у двох валютах. Блоки зберігають уміст файлів, а inode — самі файли. mke2fs, інструмент, що створює файлові системи ext2, ext3 і ext4, створює один inode на кожні стільки-то байтів простору, і його посібник попереджає, що це співвідношення байтів на inode неможливо змінити, коли файлова система вже існує. Тому том може вичерпати місце для нових файлів, поки для їхнього вмісту місце ще є, і це заскочує кожного, хто стежить лише за відсотком зайнятого простору.

Монтування вирішує, чого коштуватиме збій. ext4 веде журнал, і його документація описує три режими даних. Типовий, `data=ordered`, примусово записує дані файлів в основну файлову систему, перш ніж метадані, що на них посилаються, буде зафіксовано в журналі. `data=writeback` відмовляється від цього порядку, і документація попереджає, що після некоректного

вимкнення в нещодавно записаних файлах можуть опинитися застарілі дані, що в деяких випадках становить ризик для безпеки. `data=journal` пропускає через журнал і самі дані; цей режим застарілий (deprecated), а його підтримку планують прибрати в січні 2028 року. Параметр `errors=` визначає, що робить файлова система, коли знаходить помилку в собі: перемонтовується лише для читання, працює далі або викликає паніку ядра й зупиняє машину.

Від цього залежить і відновлення. Коли файлову систему пошкоджено, уміст файлів може ще лежати на диску, тоді як структура, що дає їм імена, зламана, і саме тут працює відновлення даних. Запис у портфоліо про відновлення даних з SD-карт, HDD, SSD і RAID розповідає, що кожен носій виходить з ладу й віддає дані по-своєму, а в Mac ще й файлова система зі своїми примхами. Тому найбільше важило вміння дібрати метод саме під той носій, що лежить на столі, а не якась одна техніка.

Як це виглядає на практиці

Візьмімо новий розділ, `/dev/sdb1`, на якому мають лежати дані сервісу в `/srv/data`. Щоб створити на ньому файлову систему ext4 і змонтувати її вручну, достатньо трьох команд:

```
mke2fs -t ext4 -L data /dev/sdb1
mkdir -p /srv/data
mount -t ext4 /dev/sdb1 /srv/data
```

`-t ext4` задає тип, як у власній документації ядра про ext4, а `-L` дає файловій системі мітку (label), яку mke2fs обмежує 16 байтами. Решта береться зі значень за замовчуванням; у конфігурації, яку постачає e2fsprogs, кожен inode, наприклад, займає 256 байтів. Дві команди `df` після цього показують обидва види місткості:

```
df -hT /srv/data
df -i /srv/data
```

Перша виводить тип файлової системи та її зайняті й вільні блоки в одиницях, побудованих на степенях 1 024 (`-H` використав би степені 1 000). Друга рахує inode замість блоків: скільки їх усього, скільки зайнято й вільно

та який відсоток використано.

Монтування вручну діє лише до наступного перезавантаження. Щоб зробити його постійним, файловій системі додають рядок у `/etc/fstab`, файл, який mount, umount і fsck читають послідовно:

```
LABEL=data  /srv/data  ext4  defaults,nofail  0  2
```

Перше поле вказує файлову систему за міткою, а не як `/dev/sdb1`. Посібник fstab радить там `LABEL=` або `UUID=`, бо імена пристроїв часто залежать від того, в якому порядку обладнання випадково виявили, і можуть змінитися, коли диски додають чи прибирають. `defaults` зазвичай означає читання й запис плюс інші стандартні параметри ядра, а `nofail` каже mount не повідомляти про помилку, якщо пристрою немає. П'яте поле читає dump(8), і 0 залишає файлову систему поза ним. Шосте задає fsck порядок перевірок під час завантаження: 1 для кореневої файлової системи, 2 для решти й 0, щоб не перевіряти зовсім. У системі під керуванням systemd посібник радить після правки виконати `systemctl daemon-reload`. Потім `mount -a` монтує всі записи без позначки `noauto`, тож помилка в новому рядку виявиться, поки хтось ще сидить за клавіатурою.

Супровід файлової системи — це ще й стеження за місцем. Запис у портфоліо про моніторинг, що реагує на мовчання хоста називає заповнений диск одним із двох реалістичних способів, якими може впасти хост із 464 МБ пам'яті, де працюють git-сервер і вебсервер; другий — нестача пам'яті, і жоден з них сам себе не видає. Відповідь там — невеликий скрипт на таймері: раз на п'ятнадцять хвилин, з випадковим зсувом, він вимірює скільки на диску лишилося місця і скільки пам'яті доступно, і пінгує зовнішній сервіс, лише поки обидва значення тримаються вище своїх порогів, тож сигналом тривоги стає тиша. Перевірка такого типу може читати стовпчик inode з `df -i` так само, як стовпчик місця, бо будь-який із них може першим дійти до межі.

Де виникають проблеми

Більшість проблем із файловими системами з'являється як заповнений диск, машина, що не монтує того, що вчора монтувала, або дані, яких бракує після збою, і чимало з них зводиться до параметра, на який ніхто не подивився, коли файлову систему створювали.

Диск заповнений, хоча місце ще є. Кожному файлу потрібен inode, а mke2fs створює їх фіксовану кількість, коли створює файлову систему. Том із мільйонами дрібних файлів може вичерпати inode, поки `df -h` ще показує вільне місце, і відтоді на ньому не створити жодного нового файлу. Вихід — стежити за `df -i` поряд із `df -h`, а для тому, призначеного для великої кількості дрібних файлів, задавати співвідношення ключем `-i` або кількість ключем `-N` під час запуску mke2fs, бо співвідношення потім не змінити. Зміна розміру файлової системи зберігає співвідношення, і кількість inode змінюється відповідно.

Перевірку запускають на змонтованій файловій системі. Посібник e2fsck каже, що запускати його на змонтованій файловій системі загалом небезпечно. Єдиний виняток — `-n`, з яким файлову систему відкрито лише для читання, а на всі запитання дано відповідь «ні», і навіть тоді результати, які він виводить, недійсні, поки файлова система змонтована. Коли e2fsck питає, чи продовжувати на змонтованій файловій системі, посібник відповідає «ні». Вихід — спершу відмонтувати її, а для кореневої файлової системи дати перевірці відбутися під час завантаження, про що й дбає шосте поле fstab.

«Лише для читання» пише на диск. Монтування ext4 з `ro` не зупиняє записів: документація зазначає, що журнал буде відтворено, а отже, на розділ потраплять записи навіть при монтуванні лише для читання. Найбільше це важить для диска, який рятують, де будь-який запис може влучити в щось ще потрібне. Параметри `ro,noload` запобігають записам, але та сама сторінка прямо називає ціну: після некоректного відмонтування пропущене відтворення залишає файлову систему з неузгодженостями. Вихід для диска, чий уміст важливіший за його безперервну роботу, — змонтувати його з `ro,noload`, скопіювати потрібне й вважати прочитане можливо неузгодженим.

Диск змінює ім’я. Рядок, що монтує `/dev/sdb1`, монтує те, що ядро сьогодні називає `/dev/sdb1`, і посібник fstab попереджає, що це може змінитися, коли диски додають чи прибирають. Тоді новий диск може опинитися змонтованим там, де має бути старий, або старого не виявиться під час завантаження. Вихід — `LABEL=` або `UUID=` у першому полі, з мітками, заданими через `mke2fs -L` або пізніше інструментом на кшталт `e2label`.

Живлення зникає між записом і диском. За замовчуванням ext4 фіксує журнал кожні п’ять секунд, тож вимкнення живлення може коштувати останніх п’яти секунд змін метаданих, а журнал при цьому зберігає саму файлову систему неушкодженою. Дані файлів можуть загубитися й за довший час: за відкладеного виділення місця (delayed allocation) запис на диск починається лише після інтервалу, заданого в `/proc/sys/vm/dirty_expire_centisecs`. Документація називає програми, що замінюють файли без `fsync()`, зламаними, а типовий для ext4 параметр `auto_da_alloc` розпізнає два поширені шаблони, запис нового файлу з перейменуванням поверх старого та обрізання файлу з перезаписом на місці, щоб після збою файли не поверталися порожніми. Вихід — у програмі, яка пише файл: викликати `fsync()`, перш ніж повідомляти, що дані збережено, а не покладатися на цю страховку.

Часові позначки зупиняються у 2038 році. Посібник mke2fs каже, що файлові системи з inode розміром 128 байтів не підтримують часових позначок після 19 січня 2038 року, тоді як inode розміром 256 байтів і більше підтримують розширені позначки. Нові файлові системи за замовчуванням отримують 256 байтів, але стара файлова система або створена з `-I 128` досі може мати малий розмір. Вихід — знайти такі файлові системи раніше, ніж це зробить дата; документація ext4 показує, як `tune2fs -I 256` перетворює файлову систему, створену з inode по 128 байтів.

Що почитати далі

- Definitions, POSIX.1-2024 (The Open Group Base Specifications Issue 8) — визначення файлової системи, файлу, каталогу, жорсткого посилання й точки монтування.
- Overview of the Linux Virtual File System — як VFS ядра розбирає шлях і передає роботу файловій системі.
- ext4 General Information — створення й монтування ext4, режими даних, `commit=` і `ro` з журналом.
- mke2fs(8) — співвідношення байтів на inode, розмір inode, межа 2038 року й мітки томів.
- e2fsck(8) — перевірка ext2, ext3 і ext4 і чому ніколи на змонтованій файловій системі.
- fstab(5) — шість полів, `LABEL=` і `UUID=`, `nofail` і порядок перевірок під час завантаження.

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 файлова система?» 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/file-system/