Engineer

filsystem

den struktur, der gør en disk til navngivne filer og mapper

Hvad det er

Et filsystem er den del af en computer, der gør en disk, som i sig selv kun gemmer nummererede blokke af bytes, til navngivne filer ordnet i mapper. På engelsk skrives det både file system i to ord og filesystem i ét, og det er den sidste form, Linux-kernens dokumentation mest bruger. POSIX, den standard som Unix-lignende systemer følger, definerer i udgaven fra 2024 et filsystem som en samling filer og visse af deres attributter. Samme definition tilføjer, at filsystemet giver et navnerum for filernes serienumre, de numre, der entydigt identificerer hver fil inden for det filsystem, den ligger i.

Resten følger af et par andre definitioner i standarden. En fil er et objekt, der kan skrives til, læses fra eller begge dele, og som har attributter som adgangsrettigheder og type: almindelig fil, mappe, symbolsk link, FIFO, socket eller tegn- eller blokspecialfil. En mappe er en fil, der består af mappeindgange, og to indgange i samme mappe kan ikke have samme navn. En mappeindgang, som POSIX også kalder et hardlink, knytter et filnavn til en fil, og flere indgange kan pege på den samme fil. Navnet og selve filen er altså to forskellige ting. På Linux hedder posten for selve filen en inode, som GNU-manualen til df beskriver som det sted, hvor filens ejer, adgangsrettigheder, tidsstempler og placering i filsystemet står, mens navnene bor i de mapper, der peger på den.

Et Unix-lignende system viser alle sine filsystemer som ét samlet træ af navne. Det første leverer rodmappen, og hvert af de andre hænges på en mappe i træet, dets monteringspunkt (mount point), hvorefter stierne under den mappe hører til det filsystem, der er monteret dér. Inde i Linux-kernen ligger Virtual File System, VFS, mellem programmerne og de mange filsystemer, kernen understøtter. Kernens dokumentation beskriver det, også kendt som Virtual Filesystem Switch, som det lag, der giver programmerne deres grænseflade til filsystemerne og lader forskellige implementeringer eksistere side om side. Et program kalder `open(2)`, `stat(2)`, `read(2)` eller `write(2)`, og VFS slår stien op

og sender arbejdet videre til det filsystem, der har filen: ext4 på en diskpartition, for eksempel, eller et pseudofilsystem, hvis inodes kun findes i hukommelsen. De typer, den kørende kerne har registreret, står i `/proc/filesystems` .

Brug begrebet i en beslutning

Brug »Hvad er et filsystem?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Daglig drift af servere, mail, konti og opdateringer, så det, alle er afhængige af, forbliver kedeligt.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har drevet 10 PostgreSQL- og MS SQL-servere — 10 databaseservere udrullet og holdt i god drift

Har drevet 10 PostgreSQL- og MS SQL-servere Kontakt os →

Hvorfor det betyder noget

Næsten alt, hvad en maskine skal huske efter næste genstart, ligger i et filsystem: databasernes datafiler, logfilerne, uploadede dokumenter, konfigurationen. Om de overlever en strømafbrydelse, en fuld disk eller et drev, der er ved at give op, afgøres i høj grad af valg, der blev truffet, da filsystemet blev oprettet og monteret, og nogle af dem kan ikke laves om bagefter.

Oprettelsen fastlægger kapaciteten i to valutaer. Blokke rummer filernes indhold, og inodes rummer selve filerne. mke2fs, værktøjet der opretter ext2-, ext3- og ext4-filsystemer, laver én inode for hver så og så mange bytes plads, og manualen advarer om, at dette forhold mellem bytes og inodes ikke kan ændres, når filsystemet først findes. Et volumen kan derfor løbe tør for plads til nye filer, mens der stadig er plads til deres indhold, og det kommer bag på enhver, der kun holder øje med, hvor mange procent af pladsen der er brugt.

Monteringen afgør, hvad et nedbrud koster. ext4 fører en journal, og dokumentationen nævner tre datatilstande. Standarden, `data=ordered` , tvinger fildata ud i selve filsystemet, før de metadata, der henviser til dem, lægges i journalen. `data=writeback` dropper den rækkefølge, og dokumentationen advarer om, at den efter en uren nedlukning kan efterlade gamle data synlige i filer, der for nylig er skrevet. `data=journal` sender også dataene gennem journalen; den tilstand er udfaset (deprecated), og understøttelsen skal efter planen fjernes i januar 2028. Indstillingen `errors=` bestemmer, hvad filsystemet gør, når det finder en fejl i sig selv: monterer sig igen skrivebeskyttet, fortsætter eller går i panik og standser maskinen.

Gendannelse hænger også på det. Når et filsystem er beskadiget, kan filernes indhold stadig ligge på disken, mens den struktur, der giver dem navne, er gået i stykker, og det er dér, datagendannelse arbejder. Porteføljesiden om datagendannelse fra SD-kort, harddiske, SSD'er og RAID fortæller, at hvert medie svigter på sin egen måde og også skal have sine data lokket ud på sin egen måde, og at et Mac-filsystem har sine egne særheder. Det afgørende var derfor at vælge den rigtige fremgangsmåde til netop det medie, der lå på bordet, frem for at stole på én bestemt teknik.

Sådan ser det ud i praksis

Tag en ny partition, `/dev/sdb1`, der skal rumme en tjenestes data under `/srv/data`. At oprette et ext4-filsystem på den og montere det i hånden kræver tre kommandoer:

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

`-t ext4` angiver typen, som i kernens egen ext4-dokumentation, og `-L` giver filsystemet en etiket (label), som mke2fs begrænser til 16 bytes. Resten kommer fra standardværdierne; i den konfiguration, e2fsprogs leverer, fylder hver inode for eksempel 256 bytes. To `df`-kommandoer viser derefter de to slags kapacitet:

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

Den første viser filsystemets type og dets brugte og ledige blokke i enheder bygget på potenser af 1.024 (`-H` ville bruge potenser af 1.000). Den anden opgør inodes i stedet for blokke: hvor mange der er, hvor mange der er brugt og ledige, og hvor mange procent der er i brug.

En montering i hånden holder kun til næste genstart. For at gøre den permanent får filsystemet en linje i `/etc/fstab`, filen som mount, umount og fsck læser i rækkefølge:

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

Det første felt angiver filsystemet ved dets etiket i stedet for som `/dev/sdb1`. Manualen til fstab anbefaler `LABEL=` eller `UUID=` dér, fordi enhedsnavne ofte afhænger af, i hvilken rækkefølge hardwaren tilfældigvis blev fundet, og kan skifte, når der kommer diske til, eller nogle bliver fjernet. `defaults` betyder som regel læse og skrive plus kernens øvrige standardindstillinger, og `nofail` får mount til ikke at melde fejl, hvis enheden ikke findes. Det femte felt læses af dump(8), og 0 holder filsystemet udenfor. Det sjette giver fsck rækkefølgen for tjek ved opstart: 1 for rodfilsystemet, 2 for de øvrige og 0 for ingen. På et system, der kører systemd, anbefaler manualen `systemctl daemon-reload` efter en ændring. `mount -a` monterer derefter alle linjer, der ikke er mærket `noauto`, så en fejl i den nye linje viser sig, mens der stadig sidder nogen ved tastaturet.

Drift af et filsystem betyder også at holde øje med pladsen. Porteføljesiden om værtsovervågning efter dødmandsknap-princippet nævner en disk, der løber fuld, som den ene af to realistiske måder, en maskine på 464 MB med en forge og en webserver kan dø på. Den anden er mangel på hukommelse, og ingen af dem melder sig selv. Svaret dér er et lille script, som en timer starter fire gange i timen på et lidt forskudt tidspunkt, og som måler ledig diskplads og tilgængelig hukommelse og kun giver lyd fra sig udadtil, så længe begge holder sig over deres grænser, så stilheden er alarmen. Et tjek af den slags kan lige så godt læse inode-kolonnen fra `df -i` som pladskolonnen, for begge kan ramme loftet først.

Hvor det går galt

De fleste problemer med filsystemer viser sig som en fuld disk, en maskine, der ikke vil montere det, den monterede i går, eller data, der mangler efter et nedbrud, og meget af det kan føres tilbage til en indstilling, ingen så på, da filsystemet blev lavet.

Disken er fuld, selv om der er plads. Hver fil kræver en inode, og mke2fs laver et fast antal af dem, når filsystemet oprettes. Et volumen med millioner af små filer kan bruge alle sine inodes op, mens `df -h` stadig viser ledig plads, og derefter kan der ikke oprettes en eneste ny fil. Løsningen er at holde øje med `df -i` ved siden af `df -h` og, for et volumen beregnet til mange små filer, at sætte

forholdet med `-i` eller antallet med `-N`, når mke2fs køres, for forholdet kan ikke ændres bagefter. Ændres filsystemets størrelse, bevares forholdet, og antallet af inodes følger med.

Et tjek kører på et monteret filsystem. Manualen til e2fsck siger, at det generelt ikke er sikkert at køre den på et monteret filsystem. Den eneste undtagelse er `-n`, som åbner filsystemet skrivebeskyttet og svarer nej til alle spørgsmål, og selv da er de resultater, den skriver, ikke gyldige, mens filsystemet er monteret. Spørger e2fsck, om den skal fortsætte på et monteret filsystem, er manualens svar nej. Løsningen er at afmontere først eller, for rodfilsystemet, at lade tjekket køre ved opstart, som det sjette felt i fstab sørger for.

»Skrivebeskyttet« skriver til disken. At montere ext4 med `ro` stopper ikke skrivningerne: dokumentationen gør opmærksom på, at journalen afspilles igen, og at der dermed skrives til partitionen, også når den er monteret skrivebeskyttet. Det betyder mest på en disk, der skal reddes, hvor hver skrivning kan ramme noget, der stadig skal bruges. Indstillingerne `ro,noload` forhindrer skrivningerne, men samme side gør prisen klar: efter en uren afmontering står filsystemet tilbage med uoverensstemmelser, når afspilningen springes over. Løsningen for en disk, hvis indhold betyder mere end dens oppetid, er at montere den med `ro,noload`, kopiere det nødvendige væk og behandle det, der læses, som muligvis inkonsistent.

En disk skifter navn. En linje, der monterer `/dev/sdb1`, monterer det, som kernen kalder `/dev/sdb1` i dag, og fstab-manualen advarer om, at det kan ændre sig, når der kommer diske til, eller nogle bliver fjernet. Så kan en ny disk ende med at blive monteret dér, hvor den gamle hører hjemme, eller den gamle kan mangle ved opstart. Løsningen er `LABEL=` eller `UUID=` i første felt, med etiketter sat af `mke2fs -L` eller senere af et værktøj som `e2label`.

Strømmen går mellem skrivning og disk. Som standard skriver ext4 sin journal hvert femte sekund, så et strømsvigt kan koste de seneste fem sekunders ændringer i metadata, mens journalen holder selve filsystemet uskadt. Fildata kan gå længere tilbage: med forsinket allokering begynder udskrivningen først efter det interval, der er sat i `/proc/sys/vm/dirty_expire_centisecs`. Dokumentationen kalder programmer, der erstatter filer uden `fsync()`, for defekte, og ext4's standardindstilling `auto_da_alloc` genkender de to almindelige mønstre, at skrive en ny fil og omdøbe den hen over den gamle eller at afkorte en fil og skrive den forfra, for at undgå filer, der kommer tomme

tilbage efter et nedbrud. Løsningen er i det program, der skriver filen: kald `fsync()`, før nogen får at vide, at dataene er gemt, i stedet for at læne sig op ad det sikkerhedsnet.

Tidsstemplerne stopper i 2038. Manualen til mke2fs siger, at filsystemer med inodes på 128 bytes ikke understøtter tidsstempler efter 19. januar 2038, mens inodes på 256 bytes eller mere understøtter udvidede tidsstempler. Nye filsystemer får 256 bytes som standard, men et gammelt filsystem eller et, der er lavet med `-I 128`, kan stadig have den lille størrelse. Løsningen er at finde de filsystemer, før datoen gør det; ext4-dokumentationen viser `tune2fs -I 256` konvertere et filsystem, der er oprettet med inodes på 128 bytes.

Læs mere

- Definitions, POSIX.1-2024 (The Open Group Base Specifications Issue 8) — definitionerne af filsystem, fil, mappe, hardlink og monteringspunkt.
- Overview of the Linux Virtual File System — hvordan kernens VFS slår en sti op og overlader arbejdet til et filsystem.
- ext4 General Information — oprettelse og montering af ext4, datatilstandene, `commit=` og `ro` med journalen.
- mke2fs(8) — forholdet mellem bytes og inodes, inodestørrelsen, grænsen i 2038 og etiketter.
- e2fsck(8) — tjek af ext2, ext3 og ext4, og hvorfor aldrig på et monteret filsystem.
- fstab(5) — de seks felter, `LABEL=` og `UUID=`, `nofail` og rækkefølgen for tjek ved opstart.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er et filsystem?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

welcome@engineer.company +45 50 10 61 84 København, Danmark

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/file-system/