21. Soubory a systémy souborů v Unixu

Obsah

- Vztah adresářové struktury a fyzického systému souborů
- Typy fyzických systémů souborů
- RAID
- Základní vlastnosti ZFS
- Typy souborů
- Přístupová práva

Vztah adresářové struktury a fyzického systému souborů

Logický vs. fyzický systém souborů

- **Logický systém souborů** je abstraktní koncept, který vidí uživatel
  - V každém operačním systému máme pouze jeden logický systém
  - Tento logický systém typicky obsahuje **několik fyzických systémů souborů**
- **Fyzický systém souborů** je konkrétní implementace uložení dat na disku
  - Definuje, jak jsou data fyzicky organizována na paměťovém médiu

Implementace v Unixu

V unixových systémech je adresářová struktura implementována pomocí:

1. **Adresářů** - speciální typ souboru obsahující tabulku s dvojicemi:
   - Jméno souboru
   - Odkaz na i-node (informační uzel)
2. **i-node (Information Node)**
   - Tabulka obsahující všechny informace o souboru:
     - Metadata (vlastník, skupina, přístupová práva, časové značky)
     - Velikost souboru
     - Typ souboru
     - Počet pevných odkazů na soubor
     - Odkazy na datové bloky obsahující skutečná data souboru
3. **Datových bloků**
   - Obsahují skutečná data souboru
   - Velikost bloku je typicky 4KB nebo 8KB

Typy odkazů

Pevný odkaz (Hard Link)

- Více odkazů (jmen) na stejný i-node
- Všechny odkazy jsou rovnocenné, neexistuje "originál"
- Omezení:
  - Nelze vytvářet mezi různými fyzickými systémy souborů
  - Nelze vytvářet na adresáře (s výjimkou "." a "..")
  - Počet pevných odkazů se uchovává v i-node a kontroluje jej systém

```
# Vytvoření pevného odkazu
ln soubor.txt novy_odkaz.txt
```

Symbolický odkaz (Soft Link)

- Speciální typ souboru obsahující textovou cestu k cílovému souboru
- Funguje jako ukazatel na jiný soubor
- Výhody:
  - Může odkazovat na jakýkoliv soubor včetně adresářů
  - Může odkazovat na soubory v jiných souborových systémech

- Může odkazovat i na neexistující soubory (konzistenci nekontroluje systém)

```
# Vytvoření symbolického odkazu
ln -s soubor.txt sym_odkaz.txt
```

Typy fyzických systémů souborů

Historický vývoj

1. **UFS (Unix File System)**
   - Původní souborový systém pro Unix
   - Neefektivní pro velké soubory
   - Limitovaná podpora pro velké disky
2. **VxFS (Veritas File System)**
   - Optimalizován pro velké soubory (např. databáze)
   - Založen na konceptu velkých souborů, kterých však není mnoho
3. **Ext2/Ext3/Ext4 (Extended File System)**
   - Ext4 je aktuálně nejpoužívanější v Linuxu
   - Podporuje velké soubory a souborové systémy
   - Journaling (v Ext3 a Ext4) pro ochranu integrity dat
4. **ZFS (Zettabyte File System)**
   - Vyvinutý Sun/Oracle pro Solaris
   - Pokročilé funkce včetně deduplikace dat a ochrany integrity
5. **Btrfs (B-tree File System)**
   - Moderní souborový systém pro Linux
   - Podobné funkce jako ZFS
6. **Jiné systémy**
   - Windows: **NTFS**, **exFAT** (dříve FAT32)
   - MacOS: **APFS** (dříve HFS+)

Kategorie systémů souborů

Podle umístění

1. **Diskové systémy souborů**
   - Uložené na fyzickém disku
   - Mohou být na celém disku nebo ve vymezeném prostoru (oddílu)
   - Mohou být také v souboru (např. ISO obrazy, virtuální disky)
2. **Síťové systémy souborů**
   - Data jsou dostupná přes síť
   - Příklady: NFS, SMB/CIFS, AFS
   - Požadavky pro přístup k datům se posílají přes síť
3. **Virtuální systémy souborů**
   - Nemají trvalé fyzické úložiště
   - Existují pouze v RAM paměti
   - Rychlé, ale drahé
   - Příklady: tmpfs, ramfs

Podle vlastností

1. **Journaling systémy souborů**
   - Udržují záznamy o změnách před jejich aplikací
   - Chrání před poškozením dat při výpadku napájení
   - Příklady: Ext3, Ext4, XFS, JFS
2. **Copy-on-write systémy souborů**
   - Místo přepisování dat vytváří nové kopie
   - Umožňují snapshoty a ochranu proti chybám
   - Příklady: ZFS, Btrfs

RAID (Redundant Array of Independent Disks)

RAID je technologie, která kombinuje více fyzických disků do jedné logické jednotky pro zvýšení výkonu, kapacity nebo redundance.

Základní úrovně RAID

RAID 0 (Striping - Proužkování) - Data jsou rozdělena na bloky a zapisována střídavě na všechny disky (Disky v řadě) - Výhody: Zvýšení rychlosti čtení/zápisu, využití plné kapacity všech disků - Nevýhody: Žádná redundance - při selhání jednoho disku jsou ztracena všechna data - Minimální počet disků: 2

(Concat) - Skládání disku za sebe - Virtuálně větší disk

RAID 1 (Mirroring - Zrcadlení)

- Data jsou duplikována na dva nebo více disků
- Výhody: Plná redundance dat, vyšší rychlost čtení
- Nevýhody: Využitelná kapacita je pouze 50% z celkové kapacity disků
- Minimální počet disků: 2

RAID 5 (Striping with Parity)

- Data jsou rozdělena na bloky a zapisována střídavě na všechny disky
- Paritní informace jsou rovnoměrně rozloženy na všech discích
- Výhody: Dobrý poměr výkonu a redundance, přežije výpadek jednoho disku
- Nevýhody: Pomalejší zápis kvůli výpočtu parity
- Kapacita - (n-1) nejmenšího disku
- Minimální počet disků: 3

RAID 6 (Striping with Double Parity)

- Podobný RAID 5, ale s dvojitou paritou
- Výhody: Přežije výpadek až dvou disků současně
- Nevýhody: Ještě pomalejší zápisy než RAID 5
- Minimální počet disků: 4 (výhodný až od 6, protože na 4 je lepší mirroing - proč dva paritní na 4?)

Kombinované RAID úrovně

RAID 10 (RAID 1+0)

- Nejprve RAID 1 (zrcadlení), pak RAID 0 (proužkování)
- Výhody: Vysoký výkon i redundance
- Nevýhody: Vysoká cena (využitelná kapacita pouze 50%)
- Minimální počet disků: 4

RAID 01 (RAID 0+1)

- Nejprve RAID 0 (proužkování), pak RAID 1 (zrcadlení)
- Méně používaný než RAID 10
- Minimální počet disků: 4

Další úrovně RAID

- **RAID 2**: Striping na úrovni bitů s dedikovanými paritními disky (historické)
- **RAID 3**: Striping na úrovni bajtů s jedním paritním diskem
- **RAID 4**: Striping na úrovni bloků s jedním paritním diskem (problém s přetížením paritního disku)

Základní vlastnosti ZFS

ZFS (Zettabyte File System) je pokročilý souborový systém původně vyvinutý společností Sun Microsystems pro operační systém Solaris, nyní dostupný i pro jiné operační systémy.

Klíčové vlastnosti ZFS

1. **Ochrana dat a integrity**
   - Kontrolní součty všech dat a metadat
   - Automatická detekce a oprava tichého poškození dat
   - Atomické operace - zabraňuje výskytu poškozených dat při náhlém vypnutí
2. **Správa úložiště**
   - Pooling úložišť - disky jsou agregovány do "poolů" úložiště
   - Dynamické přidávání úložiště bez nutnosti přeformátování
   - Podpora pro obrovské kapacity (až 256 kvadrilionů ZB)
3. **Pokročilé funkce**
   - **Copy-on-write** - nikdy nepřepisuje existující data
   - **Snapshoty** - okamžité kopie stavu souborového systému
   - **Klony** - zapisovatelné kopie snapshotu
   - **Deduplikace dat** - automatické odstranění duplicitních dat
   - **Komprese dat** - transparentní komprese pro úsporu místa
4. **RAID a redundance**
   - Integrovaný RAID (nazývaný RAID-Z1, RAID-Z2, RAID-Z3)
   - Samoopravné funkce pomocí redundantních dat
5. **Výkon**
   - Adaptivní vyrovnávací paměť ARC (Adaptive Replacement Cache)
   - Předvídatelné čtení a prefetching
   - Rozvrhování I/O operací

Typy souborů

V unixových systémech rozlišujeme několik základních typů souborů, které lze identifikovat pomocí příkazu `ls -l`:

Podle obsahu

1. **Textové soubory**
   - Obsahují čitelný text
   - Lze je editovat v textových editorech
   - Často používané pro konfigurační soubory, skripty, zdrojový kód
2. **Binární soubory**
   - Obsahují data v binárním formátu
   - Nelze je přímo číst nebo editovat v textovém editoru
   - Příklady: spustitelné soubory, knihovny, obrázky

Podle typu v systému

Při výpisu `ls -l` je typ souboru zobrazen jako první znak v právech:

1. **Obyčejný soubor (`-`)**
   - Běžný soubor obsahující data
   - Nejčastější typ souboru
2. **Adresář (`d`)**
   - Speciální typ souboru obsahující odkazy na jiné soubory
   - Používá se pro organizaci souborového systému
3. **Symbolický odkaz (`l`)**
   - Odkaz na jiný soubor nebo adresář
   - Obsahuje cestu k cílovému souboru
4. **Speciální blokový soubor (`b`)**
   - Reprezentuje blokové zařízení (např. disk)
   - Přístup k datům po blocích určité velikosti
5. **Speciální znakový soubor (`c`)**
   - Reprezentuje znakové zařízení (např. terminál, myš)
   - Přístup k datům po jednotlivých znacích

6. **Pojmenovaná roura (`p`)**
   - Speciální soubor pro komunikaci mezi procesy (FIFO)
   - Teoreticky má neomezenou paměť (technické a i-node omezení)
   - Také nazýván "pipe" nebo "pajpa"
7. **Socket (`s`)**
   - Speciální soubor pro síťovou komunikaci mezi procesy

Příklad výpisu `ls -l`:

```
drwxr-xr-x 2 user group 4096 May 4 17:42 Documents
-rw-r--r-- 1 user group 1234 May 4 17:43 hello.txt
lrwxrwxr-x 1 user group 4 May 4 17:44 link.sh -> hello.txt
```

Přístupová práva

Unixové systémy používají jednoduchý, ale efektivní systém přístupových práv pro řízení, kdo může číst, zapisovat nebo spouštět soubory.

Základní přístupová práva

Každý soubor nebo adresář má přístupová práva definovaná pro tři kategorie uživatelů: 1. **Vlastník** (`u` - user) 2. **Skupina** (`g` - group) 3. **Ostatní** (`o` - others)

Pro každou kategorii existují tři základní práva: 1. **Čtení** (`r` - read) - hodnota 4 - Pro soubor: možnost číst obsah - Pro adresář: možnost zobrazit obsah (výpis souborů)

2. **Zápis** (`w` - write) - hodnota 2
   - Pro soubor: možnost měnit obsah
   - Pro adresář: možnost vytvářet, přejmenovávat a mazat soubory
3. **Spuštění** (`x` - execute) - hodnota 1
   - Pro soubor: možnost spustit jako program
   - Pro adresář: možnost vstoupit do adresáře a přistupovat k souborům

Zobrazení a interpretace přístupových práv

Příkaz `ls -l` zobrazuje přístupová práva jako řetězec 10 znaků:

```
-rw-r--r--
```

- První znak označuje typ souboru
- Následující 3 znaky jsou práva vlastníka
- Další 3 znaky jsou práva skupiny
- Poslední 3 znaky jsou práva ostatních

Změna přístupových práv

Příkaz `chmod` Změnu přístupových práv lze provést dvěma způsoby:

1. **Symbolický režim**
   - Specifikace kategorie: `u` (vlastník), `g` (skupina), `o` (ostatní), `a` (všichni)
   - Operace: `+` (přidat), `-` (odebrat), `=` (nastavit)
   - Práva: `r` (čtení), `w` (zápis), `x` (spuštění)

```
chmod u+x soubor           # Přidá vlastníkovi právo na spuštění
chmod g-w soubor           # Odebere skupině právo na zápis
chmod a=r soubor           # Nastaví všem pouze právo čtení
chmod u+rwx,g+rx,o= soubor # Komplexní nastavení
```

2. **Oktalový (číselný) režim**
   - Každé právo má číselnou hodnotu: `r=4`, `w=2`, `x=1`
   - Sečtením hodnot pro každou kategorii vznikne třímístné oktalové číslo

```
chmod 755 soubor  # rwxr-xr-x (vlastník všechna práva, ostatní čtení a spuštění)
chmod 644 soubor  # rw-r--r-- (vlastník čtení a zápis, ostatní jen čtení)
chmod 700 soubor  # rwx------ (pouze vlastník má všechna práva)
```

Speciální bity Kromě základních přístupových práv existují tři speciální bity:

1. **Setuid bit (4000)**
   - Pokud je nastaven na spustitelném souboru, proces se spustí s právy vlastníka souboru
   - Zobrazuje se jako `s` místo `x` v právech vlastníka
2. **Setgid bit (2000)**
   - Pokud je nastaven na spustitelném souboru, proces se spustí s právy skupiny souboru
   - Pokud je nastaven na adresáři, nově vytvořené soubory zdědí skupinu adresáře
   - Zobrazuje se jako `s` místo `x` v právech skupiny
3. **Sticky bit (1000)**
   - Pokud je nastaven na adresáři, soubory v něm může mazat pouze vlastník (používá se např. pro /tmp)
   - Zobrazuje se jako `t` místo `x` v právech ostatních

```
chmod 4755 soubor  # Nastaví setuid bit
chmod 2755 soubor  # Nastaví setgid bit
chmod 1777 adresar # Nastaví sticky bit (typicky pro /tmp)
```

Změna vlastníka a skupiny

Příkaz `chown`

```
chown novy_vlastnik soubor          # Změna vlastníka
chown novy_vlastnik:nova_skupina soubor  # Změna vlastníka a skupiny
```

Příkaz `chgrp`

```
chgrp nova_skupina soubor  # Změna skupiny
```

Příklady skriptů pro práci se soubory

Skript pro nastavení práv 700 pro všechny soubory v adresáři

```
#!/bin/bash
# nastav_700_souboru.sh [adresar]

if [ $# -ne 1 ]; then
  echo "Špatný počet parametrů. Použití: $0 adresar" >&2
  exit 1
fi

if ! [ -d "$1" ]; then
  echo "$1 není adresář" >&2
  exit 1
fi

cd "$1"
for S in *; do
  chmod 700 "$S"
done
```

Skript pro určení počtu dní v měsíci

```
#!/bin/bash
# pocet_dni.sh [mesic]

if [ $# -ne 1 ]; then
  echo "Špatný počet parametrů. Použití: $0 mesic" >&2
  exit 1
fi
```

```
case $1 in
  leden|brezen|kveten|cervenec|srpen|rijen|prosinec) echo "31 dní";;
  unor) echo "28 nebo 29 dní";;
  duben|cerven|zari|listopad) echo "$1 má 30 dní";;
  *) echo "$1 není měsíc";;
esac
```

Skript pro hromadné nahrazení textu v souboru

```
#!/bin/bash
# zmen_soubor.sh [soubor] [stare_jmeno] [nove_jmeno]

if [ $# -ne 3 ]; then
  echo "Špatný počet parametrů. Použití: $0 soubor stare_jmeno nove_jmeno" >&2
  exit 1
fi

if ! [ -f "$1" ]; then
  echo "$1 není soubor" >&2
  exit 1
fi

sed "s/$2/$3/g" "$1" > "$1.new"
mv "$1.new" "$1"
```