Management systému a správa procesů

Datum zpracování: 19.9.2023

Zpracovali: Kevin Daněk

Zadání

- free - statický výpis použité paměti
- df - volné místo v připojených souborových systémech
- vmstat - informace o paměti a discích
- ps - process status - statický výpis procesů
- top - dynamický výpis procesů a paměti
- htop - manažer procesů
- nice - přidělování priority procesu při spuštění
- renice - změna priority běžícího procesu
- kill - ukončení procesu (na základě PID)
- bg - spuštění pozastaveného procesu na pozadí
- fg - spuštění pozastaveného, nebo na pozadí běžícího procesu na popředí
- jobs - výpis procesů spuštěných na pozadí

úkoly

1. zjistěte parametry paměti a její vytížení
2. zjistětě rozložení diskových oddílů a body jejich připojení
3. vpište všechny běžící procesy v systému a včetně jejich vlastníků
4. vypište všechny své běžící procesy a zobrazte jejich vzájemné vazby
5. spusťe déle běžící proces a pak jej z jiného terminálu ukončete pomocí příkazu kill
6. spusťte déle běžící proces (např. cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 100)
    1. suspendujte jej příkazem kill
    2. spusťte jej na pozadí
    3. vypište běžící procesy
    4. vraťte jej do běhu na popředí

Veškeré postupy dokumentujte pomocí screenshotů nebo kopírujte stav terminálu do elaborátu.

Úkol zpracujte individuálně!

Elaborát do e-learningu nahrávejte ve formátu PDF.

Zjistěte parametry paměti a její využití

Parametry paměti můžeme zjistit pomocí příkazu vmstat, který nám na standardní výstup vrátí tabulku, kterou můžete vidět na obrázku níže. Můžeme se z ní dočíst kolik má paměť volného místa, kolik je rezervováno na vyrovnávací paměť a kolik na cache.

```
kevin.danek@A0320:/$ vmstat
procs -----------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
 0  0      0 13832900  55720 1415576    0    0    43     8  101  680  1  1 98  0  0
```

Pomocí příkazu `free` můžeme zjistit využití paměti, kde nás zajímá právě sloupeček `free` na řádku `mem`.

```
kevin.danek@A0322:/$ free
               total        used        free      shared  buff/cache   available
Mem:        16228972      501480    14452548       89512     1274944    15341716
Swap:        2097148           0     2097148
```

Zjistěte rozložení diskových oddílů a body jejich připojení

Informace o diskových oddílech můžeme získat pomocí příkazu `df`, který nám vrátí informace o připojených souborových systémech. Z disků nás zajímají řádky `/dev/sda1` a `/dev/sda2`. Bod připojení (mount) zjistíme ve sloupci `Mounted on`.

```
kevin.danek@A0322:/$ df
Filesystem     1K-blocks      Used Available Use% Mounted on
tmpfs            1622900      1948   1620952   1% /run
/dev/sda2      244506940 136816332  95197552  59% /
tmpfs            8114484         0   8114484   0% /dev/shm
tmpfs               5120         4      5116   1% /run/lock
tmpfs            8114484         0   8114484   0% /run/qemu
/dev/sda1         523244      6216    517028   2% /boot/efi
tmpfs            1622896       132   1622764   1% /run/user/115
tmpfs            1622896       124   1622772   1%
```

Vypište všechny běžící procesy v systému a včetně jejich vlastníků

Výpis běžících procesů lze získat pomocí příkazu `ps`. K tomu byly doplněny následující možnosti:

- `e` pro výpis všech procesů
- `f` pro formátování argumentů, se kterými byly programy spuštěny (sloupec CMD)
- `l` pro dlouhý formát

Vlastníky procesů vyčteme ze sloupce `UID`, ve kterém jsou názvy vlastníků.

```
kevin.danek@A0322:/$ ps -efl
F S UID          PID    PPID  C PRI  NI ADDR SZ WCHAN  STIME TTY          TIME CMD
4 S root           1       0  0  80   0 - 42073 -      zář18 ?      00:00:08 /sbin/init splash
1 S root           2       0  0  80   0 -     0 -      zář18 ?      00:00:00 [kthreadd]
1 I root           3       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [rcu_gp]
1 I root           4       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [rcu_par_gp]
1 I root           5       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [slub_flushwq]
1 I root           6       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [netns]
1 I root           8       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [kworker/0:0H-events_highpri]
1 I root          10       2  0  60 -20 -     0 -      zář18 ?      00:00:00 [mm_percpu_wq]
1 I root          11       2  0  80   0 -     0 -      zář18 ?      00:00:00 [rcu_tasks_kthread]
1 I root          12       2  0  80   0 -     0 -      zář18 ?      00:00:00 [rcu_tasks_rude_kthread]
```

Vypište všechny své běžící procesy a zobrazte jejich vzájemné vazby

Vazby mezi běžícími procesy si můžeme zobrazit pomocí příkazu `pstree`, který nám seznam běžících procesů vykreslí ve struktuře připomínající stromovou strukturu. Lze z ní vyčíst, jakého procesu je jiný proces potomek, či naopak má proces nějaké podprocesy.

S tím, že je struktura vykreslována do řádku, je velmi užitečná s příkazem `grep`.

```
kevin.danek@A0322:/$ pstree
systemd─┬─ModemManager───2*[{ModemManager}]
        ├─NetworkManager───2*[{NetworkManager}]
        ├─accounts-daemon───2*[{accounts-daemon}]
        ├─acpid
        ├─agetty
        ├─atd
        ├─automount───3*[{automount}]
        ├─avahi-daemon───avahi-daemon
        ├─blkmapd
        ├─bluetoothd
        ├─colord───2*[{colord}]
        ├─cron
        ├─cups-browsed───2*[{cups-browsed}]
        ├─cupsd───dbus
```

Spusťte déle běžící proces a pak jej z jiného terminálu ukončete pomocí příkazu kill

V tomto případě jsem pustil v jednom okně terminálu ukázkový příklad, který je ze zadání. Následně jsem v druhém okně terminálu pomocí příkazu `htop` zjistit PID (Process Identifier) běžícího příkazu a pomocí příkazu `kill`, který přijímá jako poziční argument PID procesu, ho ukončil.

```
kevin.danek@A0322:/$ cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w
100
#Z jiného terminálu
kevin.danek@A0322:/$ htop
kevin.danek@A0322:/$ kill 14887
```

Spusťte déle běžící proces

suspendujte jej příkazem kill

Pozastavení procesu jsem provedl příkazem kill, s argumentem -STOP, který procesu pomocí signálu sdělil, že se má zastavit.

spusťte jej na pozadí

Pozastavený proces jsem následně spustil na pozadí pomocí příkazu bg. Tento příkaz ovšem nepřijímá PID, ale jobspec, neboli “pořadové číslo” úlohy.

vypište běžící procesy

Vzhledem k tomu, že v tomto případě běžící procesy odpovídají úlohám, použil jsem příklad jobs.

vraťte jej do běhu na popředí

Pro běh úlohy v popředí jsem použil příkaz analogický k bg, neboli fg, který opět přijímá jako argument jobspec.

```
kevin.danek@A0322:/$ sleep 10000 &
[1] 16325
kevin.danek@A0322:/$ kill -STOP 16325
kevin.danek@A0322:/$ jobs
[1]+  Stopped                 sleep 10000
kevin.danek@A0322:/$ bg 1
[1]+ sleep 10000 &
kevin.danek@A0322:/$ jobs
[1]+  Running                 sleep 10000 &
kevin.danek@A0322:/$ fg 1
sleep 10000
```

Závěr

Průvodní 4 úlohy byly příjemným a trochu i potřebným osvěžením některých utilit pro správu a management systému. Pro úlohu 5 a 6 nastal problém se správou procesu běžící v jiné instanci terminálu, kdy příkazem `ps` vylistovat nešel, ovšem `htop` se postaral o získání správného PIDu pro jeho ukončení. V úloze 6 mi chvíli trvalo pochopit zadání, co je po mě vlastně chtělo, ale doufám, že jsem se dostal do tíženého konce.