TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Management systému a správa procesů

Datum zpracování: 26.09.2023

Zpracovali: Knespl Daniel

Zadání

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

Postup

*1. Pro zjištění parametrů a vytížení paměti, lze využít příkazu **free***

```
daniel.knespl@A0320:~$ free
               total        used        free      shared  buff/cache   available
Mem:        16228968      547772    12931988       85336     2749208    15284540
Swap:        2097148           0     2097148
```

Obrázek 1: výstup free

*Ten nám dá informace o velikosti paměti a swapovacího prostoru, a jejich využití v okamžiku spuštění příkazu. Pro dynamické pozorování lze využít příkazů **top** a **htop**, ty jsou ale lépe využity pro dynamické pozorování procesů.*

*2. Pro zjištění rozložení diskových oddílů se využívá příkaz **df**. Ten nám mimo jiné také dává informace o tom, na jakém místě je připojen (sloupec Mounted on).*

```
daniel.knespl@A0320:~$ df
Filesystem                     1K-blocks        Used   Available Use% Mounted on
tmpfs                            1622900        2012     1620888   1% /run
/dev/sda2                      244506940   137710800    94303084  60% /
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
none                             8114484      185140     7929344   3% /tmp/guest-50hzez
tmpfs                            1622896         176     1622720   1% /run/user/996
147.230.77.233:/data/NFS/home 45095012352 22554125824 22540886528  51% /home
tmpfs                            1622896         128     1622768   1% /run/user/28288
```

Obrázek 2: výstup df

*3. Pro výpis všech procesů lze využít příkazů **ps**, **top**, **htop** a pravděpodobně nespočet jiných.*
*V případě **ps** je potřeba využít přepínačů **fe**. Přepínač **f** pro zobrazení uživatelů, kterým procesy patří viz. sloupec UID. Přepínač **e** zajišťuje zobrazení všech procesů. Příkazy **top** a **htop** jsou na rozdíl od **ps** dynamické a nevyžadují žádné další přepínače. Informace o vlastníkovi se nachází ve sloupci USER v obou výpisech.*

```
daniel.knespl@A0320:~$ ps -fe
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  0 zář21 ?        00:00:26 /sbin/init splash
root           2       0  0 zář21 ?        00:00:00 [kthreadd]
root           3       2  0 zář21 ?        00:00:00 [rcu_gp]
root           4       2  0 zář21 ?        00:00:00 [rcu_par_gp]
root           5       2  0 zář21 ?        00:00:00 [slub_flushwq]
root           6       2  0 zář21 ?        00:00:00 [netns]
root           8       2  0 zář21 ?        00:00:00 [kworker/0:0H-events_highpri]
root          10       2  0 zář21 ?        00:00:00 [mm_percpu_wq]
root          11       2  0 zář21 ?        00:00:00 [rcu_tasks_kthread]
root          12       2  0 zář21 ?        00:00:00 [rcu_tasks_rude_kthread]
root          13       2  0 zář21 ?        00:00:00 [rcu_tasks_trace_kthread]
root          14       2  0 zář21 ?        00:00:00 [ksoftirqd/0]
root          15       2  0 zář21 ?        00:01:00 [rcu_preempt]
root          16       2  0 zář21 ?        00:00:02 [migration/0]
root          17       2  0 zář21 ?        00:00:00 [idle_inject/0]
root          19       2  0 zář21 ?        00:00:00 [cpuhp/0]
root          20       2  0 zář21 ?        00:00:00 [cpuhp/1]
root          21       2  0 zář21 ?        00:00:00 [idle_inject/1]
root          22       2  0 zář21 ?        00:00:02 [migration/1]
root          23       2  0 zář21 ?        00:00:05 [ksoftirqd/1]
```

Obrázek 3: ps -fe (prvních několik záznamů - seřazeny podle PID)

*4. Pro zobrazení vazeb mezi procesy, můžeme využít příkazu **pstree**, který ve stromové struktuře naznačuje tyto vazby. Samotný **pstree** vypisuje strom procesů s kořenem v procesu init, je ale možné místo něj doplnit uživatele, pro kterého hledáme strom procesů.*

```
daniel.knespl@A0320:~$ ps -u daniel.knespl
    PID TTY          TIME CMD
  79682 ?        00:00:00 systemd
  79683 ?        00:00:00 (sd-pam)
  79690 ?        00:00:00 pipewire
  79691 ?        00:00:00 pipewire-media-
  79692 ?        00:00:00 pulseaudio
  79734 ?        00:00:00 dbus-daemon
  79750 ?        00:00:00 sshd
  79752 ?        00:00:00 xdg-document-po
  79755 ?        00:00:00 xdg-permission-
  79775 pts/0    00:00:00 bash
  79903 pts/0    00:00:00 ps
daniel.knespl@A0320:~$ pstree daniel.knespl
sshd---bash---pstree

systemd---(sd-pam)
        |-dbus-daemon
        |-pipewire---{pipewire}
        |-pipewire-media-+---{pipewire-media-}
        |-pulseaudio---2*[{pulseaudio}]
        |-xdg-document-po-+-fusermount3
        |                 `-5*[{xdg-document-po}]
        `-xdg-permission-+---2*[{xdg-permission-}]
```

Obrázek 4: všechny procesy uživatele - ps, vazby mezi procesy - pstree

*5. Pro ukončení procesu je nutné, znát jeho PID, ten lze nalézt pomocí **ps -u**. V jednom okně jsem tedy spustil proces **sleep 100** – dělání ničeho na 100s. V druhém okně jsem si zjistil jeho PID a následně ukončil příkazem **kill**.*

```
daniel.knespl@A0320:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
daniel.+   80923  0.0  0.0  19576  5376 pts/0    Ss   13:29   0:00 -bash
daniel.+   80973  0.0  0.0  19576  5504 pts/1    Ss   13:29   0:00 -bash
daniel.+   81032  0.0  0.0  19748  3456 pts/1    S+   13:34   0:00 sleep 100
daniel.+   81033  0.0  0.0  24392  4864 pts/0    R+   13:34   0:00 ps -u
daniel.knespl@A0320:~$ kill 81032
daniel.knespl@A0320:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
daniel.+   80923  0.0  0.0  19576  5376 pts/0    Ss   13:29   0:00 -bash
daniel.+   80973  0.0  0.0  19576  5504 pts/1    Ss+  13:29   0:00 -bash
daniel.+   81035  0.0  0.0  24392  4864 pts/0    R+   13:34   0:00 ps -u
```

*6. Pro poslední úlohu jsem spouštěl proces **sleep 1000**, jelikož mi na zpracování úlohy 16,6 minuty na rozdíl od **sleep 100**, které mi dává 1,6 minuty.*

*Po spuštění **sleep**u v jednom z terminálů, jsem v druhém zjistil přes **ps** PID procesu. Ten jsem následně přes **kill -s STOP** suspendoval.*

```
daniel.knespl@A0320:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
daniel.+   81960  0.0  0.0  19576  5504 pts/0    Ss   14:34   0:00 -bash
daniel.+   82265  0.0  0.0  19576  5248 pts/1    Ss   15:02   0:00 -bash
daniel.+   82322  0.0  0.0  19748  3456 pts/1    S+   15:14   0:00 sleep 1000
daniel.+   82323  0.0  0.0  24392  4864 pts/0    R+   15:14   0:00 ps -u
daniel.knespl@A0320:~$ kill -s STOP 82322
daniel.knespl@A0320:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
daniel.+   81960  0.0  0.0  19576  5504 pts/0    Ss   14:34   0:00 -bash
daniel.+   82265  0.0  0.0  19576  5248 pts/1    Ss+  15:02   0:00 -bash
daniel.+   82322  0.0  0.0  19748  3456 pts/1    T    15:14   0:00 sleep 1000
daniel.+   82330  0.0  0.0  24392  4864 pts/0    R+   15:14   0:00 ps -u
```

Obrázek 5: ps terminál

```
[1]+  Stopped                 sleep 1000
```

Obrázek 6: sleep terminál

*Pro obnovení běhu v pozadí je potřeba v terminálu, ve kterém původně sleep běžel použít příkaz **bg** s ID suspendovaných procesů. Tato ID lze najít využitím příkazu **jobs***

```
daniel.knespl@A0320:~$ bg 1
[1]+ sleep 1000 &
```

Obrázek 7: suspendovaný sleep terminál

K výpisu procesů můžeme opět využít druhý terminál.

```
daniel.knespl@A0320:~$ ps -u
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
daniel.+   81960  0.0  0.0  19576  5504 pts/0    Ss   14:34   0:00 -bash
daniel.+   82265  0.0  0.0  19576  5248 pts/1    Ss+  15:02   0:00 -bash
daniel.+   82322  0.0  0.0  19748  3456 pts/1    S    15:14   0:00 sleep 1000
daniel.+   82376  0.0  0.0  24392  4864 pts/0    R+   15:22   0:00 ps -u
```

Obrázek 8: ps terminál se sleepem v pozadí sleep terminálu

*Pro přesun do běhu v popředí se využívá příkaz **fb**. Pokud ho využijeme v terminálu, ve kterém na pozadí běží **sleep**, přesune onen **sleep** do popředí.*

```
daniel.knespl@A0320:~$ fg
sleep 1000
```

Obrázek 9: sleep terminál s přesunutím do popředí

*V druhém terminálu nyní můžeme stejně jako v předchozí úloze využít **kill** pro ukončení procesu.*
*Pokud bychom porovnali jednotlivé výpisy **ps**, tak můžeme vidět, že jediná informace, která se měnila, byla STAT.*

Závěr

Celkově se nejednalo o příliš těžké zadání (převážně díky informacím v zadání o různých příkazech). Jednalo se spíše o připomenutí existence využitých příkazů. Napříč úkoly jsem nenarazil na žádné problémy, které nebyly zmíněny v postupové části tohoto elaborátu.