Operacijski sistemi

Naloga 2

Rok za oddajo: ponedeljek, 2. maj 2022, 23.59

Svetovno znano podjetje iz Silicijieve doline z raziskovalno-razvojno podružnico v Sloveniji išče študente, ki obvladajo programski jezik C in sistemsko porgramiranje. V ta namen je razpisalo natečaj `SEGFAULT-HUNT` za najboljše sistemsko orodje, ki bo vključeno v njihovo najnovejšo distribucijo operacijskega sistema GNU/Linux. Novo varnostno orodje imenovano `Naloga2` mora izpisovati sistemske informacije s pomočjo sistemskih klicev in datotečnega sistema `/proc`, prav tako želijo svojim strankam ponuditi drevesni generator procesov. Preliminarno testiranje, izbor in nagrajevanje je podjetje poverilo asistentom na FRI. Nagrada: kratkotrajna slava in čast, programerske sreče ter prgišče točk.

Navodila

Napišite program v programskem jeziku C, ki ga uporabljamo na naslednji način:

- `./Naloga2` *akcija parametri*

Prvi argument je torej *akcija* in predstavlja opravilo, ki naj se izvede. Argumenti od drugega naprej pa so *parametri* akcije. Program naj podpira spodaj naštete akcije. Ob uspešnem izvajanju naj bo izhodni status enak 0, razen če akcija ne zahteva drugače.

Več akcij izpisuje različne informacije o sistemu oz. o procesih, zato je pri njih prvi parameter akcije pot do imenika (`proc` sistem), ki vsebuje informacije o jedru (privzeta vrednost parametra je `/proc`). V spodnjih primerih mu podamo pot do imenika `proc-demo`, katerega najdete v arhivski datoteki proc-demo.tgz. Testni `proc` imenik ne bodo vseboval vseh podatkov iz pravega `proc` sistema, le nekatere pomembnejše datoteke, ki jih potrebujete za izvedbo naloge.

Pri implementaciji smete uporabljati ovojne funkcije sistemskih klicev, kot so npr. `fork`, `wait`, `exec`, `pause`, `kill`, `signal`, `sleep`, `getpid`, `getppid`, `opendir`, `readdir`, `closedir`, itd. Od "višje nivojskih" funkcij so dovoljene funkcije za delo z nizi, npr. `strlen`, `strcmp`, `strcasecmp`, `atoi`, itd., za delo s pomnilnikom, npr. `malloc`, `free`, `memcpy`, itd., za branje in pisanje (izpis), npr. `fopen`, `fclose`, `fscanf`, `scanf`, `printf`, `fprintf`, `sprintf` itd. Izvajanje ukazov, npr. `ps`, `uname`, iz okolja **ni** dovoljeno - vse informacije o sistemu preberite iz sistema `proc`. Prav tako **ni** dovoljeno uporabiti funkcij, ki združujejo več operacij, npr. `scandir`, `popen`.

V primeru nejasnosti se posvetujte z najbližjim asistentom ali profesorjem.

Akcija `sys`

Izpiše osnovne informacije o sistemu:

- različica Linux jedra,
- različica prevajalnika gcc za prevajanje jedra.

Izpis naj bo formatiran kot je razvidno iz primera. Primer:

```
$ ./Naloga2 sys proc-demo
Linux: 3.10.0-327.36.2.el7.x86_64
gcc: 4.8.5
```

Akcija `sysext`

Gre za nadgradnjo akcije `sys`. Izpiše osnovne informacije o sistemu:

- različica Linux jedra,
- različica prevajalnika gcc za prevajanje jedra,
- prva swap particija,
- število jedrnih modulov.

Izpis naj bo formatiran kot je razvidno iz primera. Primer:

```
$ ./Naloga2 sysext proc-demo
Linux: 3.10.0-327.36.2.el7.x86_64
gcc: 4.8.5
Swap: /dev/dm-1
Modules: 59
```

Izhodni status procesa naj bo enak številu modulov.

Akcija `me`

S pomočjo **sistemskih klicev** (glej tudi tale spisek) izpiše osnovne informacije o procesu in sistemu:

- Uid, EUid, Gid, EGid, Cwd in prioriteto procesa,
- pot do imenika v `proc` datotečnem sistemu z informacijami o procesu in dostopnost tega imenika,
- ime, izvedbo in verzijo operacijskega sistema,
- informacije o računalniku in njegovo ime,
- informacijo o časovnem pasu,
- največji možen procesorski čas za izvajanje procesa.

Izpis naj bo formatiran kot je razvidno iz primera. Primer:

```
$ ./Naloga2 me
Uid: 1005
EUid: 1005
Gid: 1006
EGid: 1006
Cwd: /home/test/src
Priority: 0
Process proc path: /proc/32/
Process proc access: yes
OS name: Linux
OS release: 3.10.0-1160.6.1.el7.x86_64
OS version: #1 SMP Tue Nov 17 13:59:11 UTC 2020
Machine: x86_64
Node name: stroj
Timezone: 0
CPU limit: 3
```

Akcija `pids`

Naraščajoče urejeno izpiše številke vse procesov (PID), vsak PID naj bo v svoji vrstici. Primer izpisa:

```
$ ./Naloga2 pids proc-demo
1
10
11
12
14
15
16
20
```

Akcija `names`

Izpiše PIDe in imena vseh procesov nepadajoče urejeno po imenih: vsako ime v svoji vrstici, v primeru enakih imen se upošteva urejenost po PIDu. Primer izpisa:

```
$ ./Naloga2 names proc-demo
1 bash
10 bash
20 copyproc.sh
14 dash
```

```
15 dash
16 dash
11 sleep
12 sleep
```

Akcija `ps`

Izpiše naslednje informacije o procesih: PID, PPID, stanje in ime procesa. Izpis naj bo urejen po PIDih. Primer izpisa:

```
$ ./Naloga2 ps proc-demo
  PID  PPID STANJE IME
    1     0      S bash
   10     1      S bash
   11    10      S sleep
   12    10      S sleep
   14     1      S dash
   15    14      T dash
   16    14      T dash
   20    14      S copyproc.sh
```

Za formatiranje izpisa lahko uporabite naslednji format:

- `printf("%5s %5s %6s %s\n", "PID", "PPID", "STANJE", "IME");`

Če je podan še dodatni argument - število, le-ta predstavlja PID. V tem primeru izpišite zgolj procese, katerih **prednik** (ne nujno neposredni in vključno z danim procesom) je proces s podanim PID. Primer izpisa:

```
$ ./Naloga2 ps proc-demo 10
  PID  PPID STANJE IME
   10     1      S bash
   11    10      S sleep
   12    10      S sleep
```

Akcija `psext`

Ta akcija je nadgradnja akcije `ps`. Dodatno izpišite še število niti in število odprtih datotek procesa. Za formatiranje izpisa uporabite naslednji format:

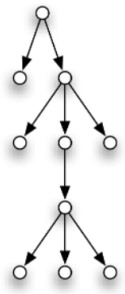
- `printf("%5s %5s %6s %6s %6s %s\n", "PID", "PPID", "STANJE", "#NITI", "#DAT.", "IME");`

Prav tako naj akcija podpira dodatni argument - PID prednika. Primer izpisa:

```
$ ./Naloga2 psext proc-demo 14
  PID  PPID STANJE  #NITI  #DAT. IME
   14     1      S      1      4 dash
   15    14      T      1      4 dash
   16    14      T      1      4 dash
   20    14      S      1      4 copyproc.sh
```

Akcija `forktree`

Ustvarjanje podane hierarhije procesov in njen izpis s klicem ukaza `pstree`. Na standardnem vhodu bo podano drevo kot zaporedje stopenj vozlišč po posameznih nivojih drevesa. Začetni proces naj sproži ustvarjanje podanega drevesa s klici `fork`. Po stvaritvi drevesa naj začetni proces s kombinacijo klicev `fork` in `exec` zažene ukaz `pstree` tako, da se bo izpisala ustvarjena hierarhija procesov vključno z začetnim procesom (koren tega dresa je torej začetni proces). Pri uporabi ukaza `pstree` vklopite stikalo `-c`. V vašem programu ne izvajajte nobenega izpisa, saj bo edini izpis na standardni izhod opravil `pstree`.
Kodiranje drevesa. Ponazorimo kodiranje drevesa z zaporedjem stopenj vozlišč s primerom.



Zgornje drevo predstavimo s kodo 2 0 3 0 1 0 3. Koren ima stopnjo 2, njegov levi otrok stopnjo 0, desni otrok pa stopnjo 3. Na naslednjem nivoju ima prvo vozlišče (vedno gledano od leve proti desni) stopnjo 0, nato stopnjo 1 in zadnje stopnjo 0. Na predzadnjem nivoju je samo eno vozlišče, ki ima stopnjo 3. Tu se vhodno zaporedje zaključi, kar pomeni, da imajo vsa vozlišča, katerih stopnje nismo podali, stopnjo 0.

Primer z razlago.

Izpis po zagonu ukaza `./Naloga2 forktree <<< "1 5 0 3"`:

```
$ ./Naloga2 forktree <<< "1 5 0 3"
Naloga2─┬─Naloga2─┬─Naloga2
        │         │         ├─Naloga2─┬─Naloga2
        │         │         │         └─Naloga2
        │         │         └─Naloga2
        │         ├─Naloga2
        │         ├─Naloga2
        │         └─Naloga2
        └─pstree
```

Začetni proces `Naloga2` in `pstree` sta vedno prisotna. Začetni proces je koren drevesa kodiranega z "`1 5 0 3`". Torej ima koren 1 otroka. Ta ima nato 5 otrok - vnuki. Prvi vnuk nima otrok, drugi vnuk pa ima 3. Vsi sledeči procesi nimajo otrok.

Prevajanje

Vaš program se bo prevedel s prevajalnikom gcc verzije 5.4. Torej lahko uporabljate razširitve jezika, kot da bi bilo vklopljeno stikalo `-std=gnu11`.

Če uporabljate sistemski klic `wait()`, ne pozabite vključiti zaglavja `wait.h`, sicer se znajo dogajati čudne stvari. (Če testirate na macOS, potem pri oddaji (v Linux) pazite pri `#include`, da bo brez `sys/`.)

Stanje oddaje prispevka

Stanje oddaje prispevka	Pri tej nalogi vam ni treba oddati ničesar
Stanje ocen	Neocenjeno
Preostali čas	14 dni 8 ure
Zadnja sprememba	-
Komentar oddane naloge	▸ Komentarji (0)