

Systemy komputerowe

Lista zadań A

Karolina Szlęk 300411

Zadanie 1

Zadanie 1. Wyjaśnij różnice między **powłoką** (ang. *shell*), **systemem operacyjnym** i **jądrem systemu operacyjnego** (ang. *kernel*). W tym celu dobierz kilka przykładów powszechnie wykorzystywanego oprogramowania. Jakie są główne zadania systemu operacyjnego z punktu widzenia programisty?

Powłoka (ang. *shell*) - na przykład bash, Finder. Program komputerowy sprawujący rolę pośrednika pomiędzy systemem operacyjnym a użytkownikiem lub aplikacjami a użytkownikiem, przyjmujący jego polecenia i wypisujący wynik działania programów. W zasadzie jest to interfejs użytkownika pozwalający na dostęp do usług systemu operacyjnego. Powłoki systemu operacyjnego używają interfejsu wiersza poleceń (CLI)- powłoki tekstowe lub graficznego interfejsu użytkownika (GUI)- powłoki graficzne, w zależności od roli komputera, operacji. Jako że jest najbardziej zewnętrzną warstwą wokół systemu operacyjnego, to nazywa się ją powłoką.

- Powłoki tekstowe - najczęściej realizowane jako interpreter poleceń w trybie tekstowym
- Powłoki graficzne -w postaci menedżera plików kontrolowanego przy pomocy myszy. Dzięki nim możemy w prosty sposób wykonywać najczęstsze operacje.

System operacyjny – oprogramowanie, które zarządza sprzętem komputerowym oraz tworzy środowisko do uruchamiania i kontroli zadań- programów. Działa jako interfejs między użytkownikiem a sprzętem komputerowym, jak również steruje wykonywaniem wszelkiego rodzaju programów. Przykładami są *Windows*, *Ubuntu*

Jądro systemu operacyjnego (ang. *kernel*) , np. Linux kernel, Windows NT - jądro systemu operacyjnego jest podstawową częścią systemu operacyjnego. Jest niezwykle ważne. To program komputerowy w rdzeniu systemu operacyjnego komputera. Program ten zapewnia pełną kontrolę nad wszystkim w systemie. Ma uprawnienia, do wykonywania poleceń w trybie zaufanym.

Głównymi zadaniami systemu operacyjnego są następujące:

- Obsługa wywołań systemowych
- Dostarczanie biblioteki standardowej oraz innych bibliotek dzielonych
- Nadzorowanie i kontrolowanie zasobów maszyny
- Dostarczanie środowiska potrzebnego do uruchomienia programów
- Zapewnianie mechanizmu do komunikacji między poszczególnymi zadaniami
- Zapewnianie mechanizmu do synchronizacji między zadaniami
- Nadzór nad wykonaniem programów
- Dostarczanie narzędzi systemowych
- Zarządzanie procesami, pamięcią operacyjną, plikami, nośnikami danych

Zadanie 2

Zadanie 2. Czym jest **zadanie** w **systemach wsadowych**? Jaką rolę pełni **monitor**? Na czym polega **planowanie zadań**? Zapoznaj się z rozdziałem „System Supervisor” dokumentu **IBM 7090/7094 IBSYS Operating System**⁵. Wyjaśnij pobieżnie znaczenie poleceń **języka kontroli zadań** (ang. *Job Control Language*) użytych na rysunku 3 na stronie 13. Do jakich zastosowań używa się dziś systemów wsadowych?

Wskazówka: Bardzo popularnym systemem realizującym szeregowanie zadań wsadowych jest **SLURM**⁶.

System wsadowy jest to taki system operacyjny, który na stałe znajduje się w pamięci operacyjnej. Po ukończeniu danego zadania system ten przekazuje dane wyjściowe kolejnemu zadaniu, gdzie dane te służą jako dane wejściowe. Może wykonywać tylko jedno zadanie.

W wsadowych systemach operacyjnych zadanie jest programem, do wykonania przez komputer (komputer ma wykonać ten program).

Monitor zaś ma za zadanie sterować kolejnością zdarzeń. Po rozpoczęciu zadania przekazuje mu (zadaniu) kontrolę nad zasobami jednostki sterującej.

Planowanie zadań, to całość procesu zarówno planowania zadań i wszystkiego co się z tym wiąże, a więc przydzielania im czasu procesora, decydowania o kolejności ich wykonania.

Przykładami systemów operacyjnych opartych na systemie wsadowym mogą być system płac czy wyciągi bankowe.

Język kontroli zadań jest nazwą określającą języki skryptowe używane w systemach.

Polecenia z rysunku 3 na stronie 13:

STOP - definiuje koniec wszystkich zadań.

IBEDT - system supervisor wywołuje system editor

IBSYS - system supervisor przejmuje kontrolę, czyta i przetwarza karty, dopóki nie wystąpi wywołanie zadania.

JOB – czyli rozpoczęcie zadania.

EXECUTE – to wykonanie zadania, co oznacza też nastąpienie przekazania kontroli

RELEASE – dla zadania, zwalnia przydzielone mu zasoby.

Zadanie 3

Zadanie 3. Jaka była motywacja do wprowadzenia **wieloprogramowych** systemów wsadowych? W jaki sposób wieloprogramowe systemy wsadowe wyewoluowały w systemy z **podziałem czasu** (ang. *time-sharing*)? Podaj przykład historycznego systemu **interaktywnego**, który nie jest wieloprogramowy.

Ściągnij ze strony przedmiotu archiwum „prog1.tar.gz”, następnie rozpakuj i skompiluj źródła poleceniem „make”.

Wieloprogramowe systemy wsadowe

Motywacja ich wprowadzenia: w komputerach 7094 bieżące zadanie było wstrzymywane do czasu zakończenia operacji wejścia-wyjścia. Procesor główny był bezczynny do momentu zakończenia tej operacji. Przy komercyjnym przetwarzaniu danych aż do 80-90% całkowitego czasu stanowił czas oczekiwania związany z operacjami wejścia-wyjścia.

Aby rozwiązać ten problem trzeba było podzielić pamięć na kilka części i umieścić w każdej z nich osobne zadanie. W czasie, gdy jedno zadanie oczekiwało na zakończenie operacji wejścia-wyjścia, inne mogło korzystać z procesora. Jeśli pamięć główna mogłaby pomieścić dostatecznie wiele zadań, to procesor główny mógłby być zajęty przez niemal 100% czasu.

Systemy z podziałem czasu (ang. *time-sharing*) - powstały z potrzeby szybkiej odpowiedzi. Technika *podziału czasu*, to odmiana systemów wieloprogramowych, gdzie każdy użytkownik posiadał terminal podłączony do komputera. Podczas debugowania programów najczęściej wydaje się krótkie polecenia (np. skompiluj iluś-stronnicową procedurę), a nie długie (np. sortuj plik z olbrzymią ilością rekordów np.999 999 999), więc komputer może zapewnić szybką, interaktywną obsługę wielu użytkownikom. Co więcej może też pracować nad złożonymi zadaniami wsadowymi w tle w tym samym czasie, kiedy w systemach bez podziału czasu komputer był bezczynny.

Interaktywny system- system interaktywny to system, który podczas działania systemu akceptuje dane wejściowe użytkownika.

Przykładem systemu interaktywnego, który nie jest wieloprogramowy jest *MS-DOS*.

Zadanie 4

Zadanie 4 (P). Uruchom program «1_ls» pod kontrolą narzędzia «ltrace -S». Na podstawie śladu wykonania programu zidentyfikuj, które z **wywołań systemowych** są używane przez procedury: «`opendir`», «`readdir`», «`printf`» i «`closedir`». Do czego służy wywołanie systemowe «`brk`»? Używając debuggera «`gdb`» i polecenia «`catch syscall brk`» zidentyfikuj, która funkcja używa «`brk`».

Uruchom program «1_ls» pod kontrolą narzędzia «ltrace -S».

Używamy polecenia `make`, a następnie polecenia `ltrace -S ./1_ls include/`

```
File Edit View Search Terminal Help
yus@yug ~$ sudo strace -o /tmp/strace.out ./ls include/
karolina@karolina-HP-ENVY-x360-Convertible-15-bp0xx:~/Pulpit/lista_1$ ltrace -S ./l_ls include/
SYS_brk(0) = 0x55dc3a557000
SYS_access("/etc/ld.so.nohwcap", 00) = -2
SYS_access("/etc/ld.so.preload", 04) = -2
SYS_openat(0xffffffffc, 0xf0783084428, 0x80000, 0) = 3
SYS_fstat(3, 0x7fffd6c40c6e0) = 0
SYS_mmap(0, 0xiad6a, 1, 2) = 0xf078326f000
SYS_close(3) = 0
SYS_access("/etc/ld.so.nohwcap", 00) = -2
SYS_openat(0xffffffffc, 0xf078328cd0, 0x80000, 0) = 3
SYS_read(3, "\177ELF\002\001\001\003", 832) = 832
SYS_fstat(3, 0x7fffd6c40cf40) = 0
SYS_mmap(0, 8192, 3, 34) = 0xf078326d000
SYS_mmap(0, 0x3f0ae0, 5, 2050) = 0xf0782c72000
SYS_mprotect(0xf0782e59000, 2097152, 0) = 0
SYS_mmap(0xf0783059000, 0x6000, 3, 2066) = 0xf0783059000
SYS_mmap(0xf078305f000, 0x3ae0, 3, 50) = 0xf078305f000
SYS_close(3) = 0
SYS_arch_prctl(4098, 0xf078326ee500, 0xf078326de30, 0xf078326d9b8) = 0
SYS_mprotect(0xf0783059000, 16384, 1) = 0
SYS_mprotect(0x55dc389c2000, 4096, 1) = 0
SYS_mprotect(0xf078328a000, 4096, 1) = 0
SYS_munmap(0xf078326f000, 109930) = 0
opendir("include/" <unfinished ...>
SYS_openat(0xffffffffc, 0x7fffd6c40e1cd, 0x90800, 0) = 3
SYS_fstat(3, 0x7fffd6c40cf0) = 0
SYS_brk(0) = 0x55dc3a557000
SYS_brk(0x55dc3a578000) = 0x55dc3a578000
<... opendir resumed> ) = 0x55dc3a557260
readdir(0x55dc3a557260 <unfinished ...>
SYS_getdents(3, 0x55dc3a557290, 0x8000, 0x55dc3a557010) = 80
<... readdir resumed> ) = 0x55dc3a557290
puts(".") <unfinished ...>
SYS_fstat(1, 0x7fffd6c40cf30) = 0
SYS_write(1, ".\n", 2. = 2
<... puts resumed> ) = 2
readdir(0x55dc3a557260) = 0x55dc3a5572a8
puts("..") <unfinished ...>
SYS_write(1, "..\n", 3.. = 3
<... puts resumed> ) = 3
readdir(0x55dc3a557260) = 0x55dc3a5572c0
puts("apue.h" <unfinished ...>
SYS_write(1, "apue.h\n", 7apue.h = 7
<... puts resumed> ) = 7
readdir(0x55dc3a557260 <unfinished ...>
SYS_getdents(3, 0x55dc3a557290, 0x8000, 0x55dc3a557010) = 0
<... readdir resumed> ) = 0
closedir(0x55dc3a557260 <unfinished ...>
SYS_close(3) = 0
<... closedir resumed> ) = 0
exit(0 <unfinished ...>
```

```

File Edit View Search Terminal Help
SYS_access("/etc/ld.so.preload", 04) = -2
SYS_openat(0xffffffff9c, 0x7f0783084428, 0x80000, 0) = 3
SYS_fstat(3, 0x7ffd6c40c6e0) = 0
SYS_mmap(0, 0x1ad6a, 1, 2) = 0x7f078326f000
SYS_close(3) = 0
SYS_access("/etc/ld.so.nohwcap", 00) = -2
SYS_openat(0xffffffff9c, 0x7f078328cdd0, 0x80000, 0) = 3
SYS_read(3, "\177ELF\002\001\001\003", 832) = 832
SYS_fstat(3, 0x7ffd6c40c740) = 0
SYS_mmap(0, 8192, 3, 34) = 0x7f078326d000
SYS_mmap(0, 0x3f0ae0, 5, 2050) = 0x7f0782c72000
SYS_mprotect(0x7f0782e59000, 2097152, 0) = 0
SYS_mmap(0x7f0783059000, 0x6000, 3, 2066) = 0x7f0783059000
SYS_mmap(0x7f078305f000, 0x3ae0, 3, 50) = 0x7f078305f000
SYS_close(3) = 0
SYS_arch_prctl(4098, 0x7f078326e500, 0x7f078326ee30, 0x7f078326d9b8) = 0
SYS_mprotect(0x7f0783059000, 16384, 1) = 0
SYS_mprotect(0x55dc389c2000, 4096, 1) = 0
SYS_mprotect(0x7f078328a000, 4096, 1) = 0
SYS_munmap(0x7f078326f000, 109930) = 0
opendir("include/" <unfinished ...>
SYS_openat(0xffffffff9c, 0x7ffd6c40e1cd, 0x90800, 0) = 3
SYS_fstat(3, 0x7ffd6c40cfe0) = 0
SYS_brk(0) = 0x55dc3a557000
SYS_brk(0x55dc3a578000) = 0x55dc3a578000
<... opendir resumed> ) = 0x55dc3a557260
readdir(0x55dc3a557260 <unfinished ...>
SYS_getdents(3, 0x55dc3a557290, 0x8000, 0x55dc3a557010) = 80
<... readdir resumed> ) = 0x55dc3a557290
puts(".") <unfinished ...>
SYS_fstat(1, 0x7ffd6c40cf30) = 0
SYS_write(1, ".\n", 2,
) = 2
<... puts resumed> ) = 2
readdir(0x55dc3a557260) = 0x55dc3a5572a8
puts("..") <unfinished ...>
SYS_write(1, "..\n", 3,
) = 3
<... puts resumed> ) = 3
readdir(0x55dc3a557260) = 0x55dc3a5572c0
puts("apue.h" <unfinished ...>
SYS_write(1, "apue.h\n", 7,
) = 7
<... puts resumed> ) = 7
readdir(0x55dc3a557260 <unfinished ...>
SYS_getdents(3, 0x55dc3a557290, 0x8000, 0x55dc3a557010) = 0
<... readdir resumed> ) = 0
closedir(0x55dc3a557260 <unfinished ...>
SYS_close(3) = 0
<... closedir resumed> ) = 0
exit(0 <unfinished ...>
SYS_exit_group(0 <no return ...>
+++ exited (status 0) +++
karolina@karolina-HP-ENVY-x360-Convertible-15-bp0xx:~/Pulpit/lista_1$

```

Na podstawie śladu wykonania programu zidentyfikuj, które z wywołań systemowych są używane przez procedury:

- <<opendir>>

```

SYS_munmap(0x7f078326f000, 109930) = 0
opendir("include/" <unfinished ...>
SYS_openat(0xffffffff9c, 0x7ffd6c40e1cd, 0x90800, 0) = 3
SYS_fstat(3, 0x7ffd6c40cfe0) = 0
SYS_brk(0) = 0x55dc3a557000
SYS_brk(0x55dc3a578000) = 0x55dc3a578000
<... opendir resumed> ) = 0x55dc3a557260

```

- <<readdir>>

```

<... opendir resumed> ) = 0x55dc3a557260
readdir(0x55dc3a557260 <unfinished ...>
SYS_getdents(3, 0x55dc3a557290, 0x8000, 0x55dc3a557010) = 80
<... readdir resumed> ) = 0x55dc3a557290

```

- <<printf>>


```

read(0x55dc3a557200) = 0x55dc3a557200
puts("apue.h" <unfinished ...>
SYS_write(1, "apue.h\n", 7apue.h
) = 7
<... puts resumed> ) = 7
read(0x55dc3a557260, <unfinished ...>

```

- <<closedir>>

```

<... read resumed> ) = 0
closedir(0x55dc3a557260 <unfinished ...>
SYS_close(3) = 0
<... closedir resumed> ) = 0
exit(0, <unfinished ...>

```

Do czego służy wywołanie systemowe «brk»?

W systemie linux wywołanie systemowe brk jest metodą na alokację pamięci, pozwala na zmiany końca segmentu pamięci.

Używając debuggera «gdb» i polecenia «catch syscall brk» zidentyfikuj, która funkcja używa «brk».

Komenda gdb 1_ls

Następnie wpisujemy catch syscall brk i run.

Teraz wystarczy, że wpiszemy bt

```

[5] * Stopped
karolina@karolina-HP-ENVY-x360-Convertible-15-bp0xx:~/Pulpit/lista_1$ gdb 1_ls
GNU gdb (Ubuntu 8.1-0ubuntu3.2) 8.1.0.20180409-git
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from 1_ls...done.
(gdb) catch syscall brk
Catchpoint 1 (syscall 'brk' [12])
(gdb) bt
(gdb) run.
Starting program: /home/karolina/Pulpit/lista_1/1_ls .

Catchpoint 1 (call to syscall brk), 0x00007ffff7df0ec9 in __brk (addr=addr@entry=0x0)
    at ../sysdeps/unix/sysv/linux/x86_64/brk.c:31
31      ../sysdeps/unix/sysv/linux/x86_64/brk.c: No such file or directory.
(gdb) bt
#0  0x00007ffff7df0ec9 in __brk (addr=addr@entry=0x0)
    at ../sysdeps/unix/sysv/linux/x86_64/brk.c:31
#1  0x00007ffff7defc71 in frob_brk () at ../sysdeps/unix/sysv/linux/dl-sysdep.c:35
#2  _dl_sysdep_start (start_argptr=start_argptr@entry=0x7ffff7ffde10,
    dl_main=dl_main@entry=0x7ffff7dd7660 <dl_main>) at ../elf/dl-sysdep.c:227
#3  0x00007ffff7dd7128 in _dl_start_final (arg=0x7ffff7ffde10) at rtld.c:414
#4  _dl_start (arg=0x7ffff7ffde10) at rtld.c:521
#5  0x00007ffff7dd6098 in _start () from /lib64/ld-linux-x86-64.so.2
#6  0x0000000000000002 in ?? ()
#7  0x00007ffff7ffe184 in ?? ()
#8  0x00007ffff7ffe1a7 in ?? ()
#9  0x0000000000000000 in ?? ()
(gdb) c

```

Nasamym początku programu uruchomił się start, który korzystał z brk. Jeżeli chcemy szukać kolejnych, to musimy teraz wpisać polecenie c, a następnie znowu bt i tak kilka razy, aż zauważymy zmianę. Widzimy, że opendir też korzysta z brk.

```
(gdb) bt
#0 0x00007ffff7afa4b9 in __brk (addr=addr@entry=0x0)
    at ../sysdeps/unix/sysv/linux/x86_64/brk.c:31
#1 0x00007ffff7afa548 in __GI__sbrk (increment=135168) at sbrk.c:41
#2 0x00007ffff7a7f199 in __GI__default_morecore (increment=<optimized out>)
    at morecore.c:47
#3 0x00007ffff7a77dac in sysmalloc (nb=nb@entry=592,
    av=av@entry=0x7ffff7dcfc40 <main_arena>) at malloc.c:2489
#4 0x00007ffff7a78ff0 in _int_malloc (av=av@entry=0x7ffff7dcfc40 <main_arena>,
    bytes=bytes@entry=576) at malloc.c:4125
#5 0x00007ffff7a7a4b5 in tcache_init () at malloc.c:2987
#6 0x00007ffff7a7abbb in tcache_init () at malloc.c:2983
#7 __GI__libc_malloc (bytes=32816) at malloc.c:3042
#8 malloc_hook_ini (sz=32816, caller=<optimized out>) at hooks.c:32
#9 0x00007ffff7ac39b6 in __alloc_dir (statp=0x7ffff7dc50, flags=0, close_fd=true, fd=3)
    at ../sysdeps/posix/opendir.c:216
#10 opendir_tail (fd=3) at ../sysdeps/posix/opendir.c:136
#11 __opendir (name=<optimized out>) at ../sysdeps/posix/opendir.c:190
#12 0x0000555555554af2 in main (argc=<optimized out>, argv=0x7ffff7fde18) at 1_ls.c:11
(gdb) c
```

Zadanie 6

Zadanie 6 (P). W systemach uniksowych wszystkie procesy są związane relacją **rodzic-dziecko**. Uruchom polecenie «ps -eo user,pid,ppid,pgid,tid,pri,stat,wchan,cmd». Na wydruku zidentyfikuj **identyfikator procesu**, **identyfikator grupy procesów**, **identyfikator rodzica** oraz **właściciela** procesu. Kto jest rodzicem procesu init? Wskaż, które z wyświetlonych zadań są **wątkami jądra**. Jakie jest znaczenie poszczególnych znaków w kolumnie STAT? Wyświetl drzewiastą reprezentację **hierarchii procesów** poleceniem pstree – które z zadań są wątkami?

Relacja rodzic-dziecko – proces, który tworzy inny proces jest nazywany rodzicem (stary proces), a tak powstałe procesy nazywane dziećmi (nowy proces). Zależność między takimi procesami to relacja rodzic-dziecko.

Identyfikator procesu – PID - unikatowy identyfikator procesu w wielozadaniowych systemach jest wyrażony jako liczba całkowita z określonego przedziału.

Identyfikator grupy procesów - PGID - identyfikator grupy procesów, do której ten proces należy. Na początku proces dziecko należy do tej samej grupy co proces rodzic, ale może założyć własną grupę procesów i mieć PID=PGID (zostaje “przywódcą”).

Identyfikator rodzica – PPID – proces może się “rozdwoić”, nowy proces nazywamy procesem dzieckiem, a stary procesem rodzicem.

Właściciel proces – USER

Podczas wykonywania tego zadania spotkamy się z powyższymi pojęciami. Ich wartości będą umieszczone w odpowiednich miejscach, oznaczone skrótami nazw.

Uruchom polecenie «ps -eo user,pid,ppid,pgid,tid,pri,stat,wchan,cmd».

Poniżej wstawione zrzuty ekranu przedstawiające wykonanie polecenia i jego efekt.

Wyświetlone zostało więcej informacji niż zmieściło się na 2 zrzutach ekranu, ale już na tych

2 zrzutach ekranu znajdują się potrzebne nam informacje. W razie potrzeby mogę dostać zrzuty ekranu przedstawiające resztę.


```

File Edit View Search Terminal Help
karolina@karolina-HP-ENVY-x360-Convertible-15-bp0xx:~/Pulpit/lista_1$ ps -eo user,pid,ppid,pgid
USER      PID  PPID  PGID   TID PRI  STAT  WCHAN  CMD
root         1     0     1     1  19  Ss    -      /sbin/init splash
root         2     0     0     2  19  S      -      [kthreadd]
root         4     2     0     4  39  I<     -      [kworker/0:0H]
root         6     2     0     6  39  I<     -      [mm_percpu_wq]
root         7     2     0     7  19  S      -      [ksoftirqd/0]
root         8     2     0     8  19  I      -      [rcu_sched]
root         9     2     0     9  19  I      -      [rcu_bh]
root        10     2     0    10 139  S      -      [migration/0]
root        11     2     0    11 139  S      -      [watchdog/0]
root        12     2     0    12  19  S      -      [cpuhp/0]
root        13     2     0    13  19  S      -      [cpuhp/1]
root        14     2     0    14 139  S      -      [watchdog/1]
root        15     2     0    15 139  S      -      [migration/1]
root        16     2     0    16  19  S      -      [ksoftirqd/1]
root        18     2     0    18  39  I<     -      [kworker/1:0H]
root        19     2     0    19  19  S      -      [cpuhp/2]
root        20     2     0    20 139  S      -      [watchdog/2]
root        21     2     0    21 139  S      -      [migration/2]
root        22     2     0    22  19  S      -      [ksoftirqd/2]
root        24     2     0    24  39  I<     -      [kworker/2:0H]
root        25     2     0    25  19  S      -      [cpuhp/3]
root        26     2     0    26 139  S      -      [watchdog/3]
root        27     2     0    27 139  S      -      [migration/3]
root        28     2     0    28  19  S      -      [ksoftirqd/3]
root        30     2     0    30  39  I<     -      [kworker/3:0H]
root        31     2     0    31  19  S      -      [kdevtmpfs]
root        32     2     0    32  39  I<     -      [netns]
root        33     2     0    33  19  S      -      [rcu_tasks_kthre]
root        34     2     0    34  19  S      -      [kauditd]
root        38     2     0    38  19  S      -      [khungtaskd]
root        39     2     0    39  19  S      -      [oom_reaper]
root        40     2     0    40  39  I<     -      [writeback]
root        41     2     0    41  19  S      -      [kcompactd0]
root        42     2     0    42  14  SN      -      [ksmd]
root        43     2     0    43   0  SN      -      [khugepaged]
root        44     2     0    44  39  I<     -      [crypto]
root        45     2     0    45  39  I<     -      [kintegrityd]
root        46     2     0    46  39  I<     -      [kblockd]
root        48     2     0    48  39  I<     -      [ata_sff]
root        49     2     0    49  39  I<     -      [md]
root        50     2     0    50  39  I<     -      [edac-poller]
root        51     2     0    51  39  I<     -      [devfreq_wq]
root        52     2     0    52  39  I<     -      [watchdogd]
root        55     2     0    55  19  S      -      [kswapd0]
root        57     2     0    57  19  S      -      [ecryptfs-kthrea]
root        99     2     0    99  39  I<     -      [kthrotld]
root       100     2     0   100  39  I<     -      [acpi_thermal_pm]
root       104     2     0   104  39  I<     -      [ipv6_addrconf]
root       113     2     0   113  39  I<     -      [kstrp]
root       131     2     0   131  39  I<     -      [charger_manager]
root       184     2     0   184  39  I<     -      [nvme-wq]
root       191     2     0   191  19  S      -      [scsi_eh_0]
root       192     2     0   192  39  I<     -      [scsi_tmf_0]

```

File	Edit	View	Search	Terminal	Help	
root	131	2	0	131	39	I< - [charger_manager]
root	184	2	0	184	39	I< - [nvme-wq]
root	191	2	0	191	19	S - [scsi_ah_0]
root	192	2	0	192	39	I< - [scsi_tm_0]
root	193	2	0	193	19	S - [scsi_ah_1]
root	194	2	0	194	39	I< - [scsi_tm_1]
root	195	2	0	195	19	S - [scsi_ah_2]
root	196	2	0	196	39	I< - [scsi_tm_2]
root	218	2	0	218	39	I< - [kworker/2:1H]
root	220	2	0	220	19	S - [jbd2/nvme0n1p6-]
root	221	2	0	221	39	I< - [ext4-rsv-conver]
root	259	1	259	259	20	S<s - /lib/systemd/systemd-journald
root	269	2	0	269	39	I< - [kworker/0:1H]
root	283	2	0	283	39	I< - [kworker/3:1H]
root	297	1	297	297	19	Ss - /lib/systemd/systemd-udev
root	315	2	0	315	39	I< - [kworker/1:1H]
root	368	2	0	368	90	S - [irq/142-ELAN073]
root	382	2	0	382	39	I< - [kworker/u9:2]
root	383	2	0	383	39	I< - [knemstick]
root	396	2	0	396	39	I< - [cfg80211]
root	402	2	0	402	90	S - [irq/145-iwlwifi]
root	436	2	0	436	39	S< - [loop0]
root	442	2	0	442	39	S< - [loop1]
root	452	2	0	452	39	S< - [loop2]
root	464	2	0	464	39	S< - [loop3]
root	527	2	0	527	41	S - [i915/signal:0]
root	528	2	0	528	41	S - [i915/signal:1]
root	529	2	0	529	41	S - [i915/signal:2]
root	530	2	0	530	41	S - [i915/signal:4]
systemd+	656	1	656	656	19	Ssl - /lib/systemd/systemd-timesyncd
systemd+	657	1	657	657	19	Ss - /lib/systemd/systemd-resolved
root	759	1	759	759	19	Ssl - /usr/sbin/ilo-sensor-proxy
root	760	1	760	760	19	Ss - /usr/sbin/acpid
root	763	1	763	763	19	Ssl - /usr/sbin/ModemManager --filter-policy=str
avahi	770	1	770	770	19	Ss - avahi-daemon: running [karolina-HP-ENVY-x3
root	772	1	772	772	19	Ss - /usr/sbin/cron -f
root	774	1	774	774	19	Ssl - /usr/bin/python3 /usr/bin/networkd-dispatch
avahi	778	770	770	778	19	S - avahi-daemon: chroot helper
root	779	1	779	779	19	Ss - /lib/systemd/systemd-logind
syslog	781	1	781	781	19	Ssl - /usr/sbin/rsyslogd -n
root	783	1	783	783	19	Ssl - /usr/sbin/thermald --no-daemon --dbus-enab
message+	787	1	787	787	19	Ss - /usr/bin/dbus-daemon --system --address=sh
root	804	1	804	804	19	Ss - /sbin/wpa_supplicant -u -s -O /run/wpa_sup
root	807	1	807	807	19	Ss - /usr/lib/bluetooth/bluetoothd
root	812	1	812	812	19	Ssl - /usr/lib/snapd/snapd
root	826	1	826	826	19	Ssl - /usr/lib/udisks2/udisksd
root	828	1	828	828	19	Ssl - /usr/sbin/NetworkManager --no-daemon
root	836	1	836	836	19	Ssl - /usr/sbin/irqbalance --foreground
root	846	1	846	846	19	Ssl - /usr/lib/accounts-service/accounts-daemon
postgres	910	1	882	910	19	S - /usr/lib/postgresql/10/bin/postgres -D /va
colord	919	1	919	919	19	Ssl - /usr/lib/colord/colord
root	924	1	924	924	19	Ssl - /usr/lib/policykit-1/polkitd --no-debug
postgres	950	910	950	950	19	Ss - postgres: 10/main: checkpoint process
postgres	951	910	951	951	19	Ss - postgres: 10/main: writer process
postgres	952	910	952	952	19	Ss - postgres: 10/main: wal writer process

Kto jest rodzicem procesu init?

Ponieważ proces init jest oznaczony jako rodzic każdego innego naszego procesu i jest wykonywany na początku, to zasadniczo nie ma on procesu rodzica, ale jeżeli już musielibyśmy jakiś znaleźć, określić, to można by powiedzieć, że jego rodzicem jest krenel.

Wskaż, które z wyświetlonych zadań są wątkami jądra.

Będą to wszystkie procesy, które są w kwadratowych nawiasach (na powyższych zrzutach ekranu). Można je też poznać po tym, że ich rodzicem jest kthreadd o PID równym 2, więc ich PPID wynosi 2.

Jakie jest znaczenie poszczególnych znaków w kolumnie STAT:

D oznacza nieprzerywalnie uśpiony (zazwyczaj wejście/wyjście)

R oznacza wykonywany lub możliwy do wykonania (w kolejce do wykonania)

S oznacza przerywalnie uśpiony (czekający na zakończenie zdarzenia)

- T oznacza zatrzymany albo przez sygnał kontroli zadań, albo z powodu śledzenia.
- W oznacza stronicowany (niepoprawne od wersji jądra 2.6.xx)
- X oznacza zmarły/niedziałający, ale to akurat nigdy nie powinno wystąpić
- Z oznacza proces niefunkcjonujący ("zombie"), zakończony, ale nie usunięty przez rodzica

Czasami mogą zostać wyświetlone dodatkowe znaki. Dzieje się tak dla formatów BSD, gdy użyto słowa kluczowego stat:

- < ma wysoki priorytet (niemiły dla innych użytkowników)
- N ma niski priorytet (miły dla innych użytkowników)
- L ma zablokowane strony w pamięci (dla czasu rzeczywistego lub IO)
- s jest liderem sesji
- l wielowątkowy
- + znajdujący się w pierwszoplanowej grupie procesów

Wyświetl drzewiastą reprezentację hierarchii procesów:

```

karollna@karollna-HP-ENVY-x360-Convertible-15-bp0xx:~/Pulpit/lista_1$ pstree
systemd├─ModemManager─2*[{ModemManager}]
      │
      └─NetworkManager─dhcpcd
            │
            └─2*[{NetworkManager}]
accounts-daemon─2*[{accounts-daemon}]
acpid
avahi-daemon─avahi-daemon
bluetoothd
boltd─2*[{boltd}]
colord─2*[{colord}]
cron
cups-browsed─2*[{cups-browsed}]
cupsd
dbus-daemon
firefox├─RDD Process─3*[{RDD Process}]
      │
      ├──3*[{Web Content─27*[{Web Content}]]
      │
      ├──2*[{Web Content─25*[{Web Content}]]
      │
      ├──Web Content─29*[{Web Content}]
      │
      ├──Web Content─32*[{Web Content}]
      │
      ├──Web Content─28*[{Web Content}]
      │
      ├──WebExtensions─23*[{WebExtensions}]
      │
      └─78*[{firefox}]
frontend─update-securebo
gdm3├─gdm-session-wor
      │
      └─gdm-wayland-ses
            │
            └─gnome-session-b
                  │
                  └─gnome-shell
                        │
                        ├──Xwayl+
                        │
                        └─ibus+
                              │
                              └─14*[{+}]
gdm3├─gdm-session-wor
      │
      └─gdm-wayland-ses
            │
            └─gnome-session-b
                  │
                  └─gnome-shell
                        │
                        ├──gsd-a11y-settin─3+
                        │
                        ├──gsd-clipboard─3*[{+}]
                        │
                        ├──gsd-color─4*[{gsd+}]
                        │
                        ├──gsd-datetime─2*[{+}]
                        │
                        ├──gsd-housekeepin─2+
                        │
                        ├──gsd-keyboard─4*[{+}]
                        │
                        ├──gsd-media-keys─4+
                        │
                        ├──gsd-mouse─2*[{gsd+}]
                        │
                        ├──gsd-power─5*[{gsd+}]
                        │
                        ├──gsd-print-notif─2+
                        │
                        ├──gsd-rfkill─2*[{gs+}]
                        │
                        ├──gsd-screensaver─2+
                        │
                        ├──gsd-sharing─3*[{g+}]
                        │
                        ├──gsd-smartcard─4*[{+}]
                        │
                        ├──gsd-sound─3*[{gsd+}]
                        │
                        ├──gsd-wacom─4*[{gsd+}]
                        │
                        ├──gsd-xsettings─4*[{+}]
                        │
                        └─3*[{gnome-session-b+}]
gdm3├─gdm-session-wor
      │
      └─gdm-wayland-ses
            │
            └─gnome-session-b
                  │
                  └─gnome-shell
                        │
                        ├──deja-dup-monito─3*[{+}]
                        │
                        └─gnome-shell
                              │
                              └─ibus-da+
                                    │
                                    └─13*[{gn+}]
gdm3├─gdm-session-wor
      │
      └─gdm-wayland-ses
            │
            └─gnome-session-b
                  │
                  └─gnome-shell
                        │
                        ├──gnome-software─3*[{+}]
                        │
                        ├──gsd-a11y-settin─3*[{+}]
                        │
                        ├──gsd-clipboard─2*[{g+}]
                        │
                        ├──gsd-color─3*[{gsd+c+}]
                        │
                        └─gsd-datetime─3*[{gsd+}]

```

