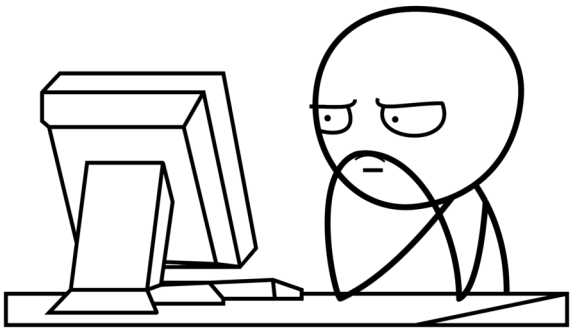
Операционни системи

ФМИ

СИ 2018

Процеси



Процеси

- Програма - някакъв изпълним код
- Процес - програма + памет (код, който се изпълнява в момента върху някаква памет)
- Нишки (threads)
  - Изпълнението на процеса може да се раздели на няколко нишки, които вървят “едновременно”
  - Паметта е обща за всички нишки
  - Всеки процес има поне една нишка
- Тъй като главно говорим за процеси с една нишка, често ще използваме **нишка** и **процес** с подобно значение.

Планиране

- Говорим за планиране на процесорното време (CPU Scheduling) - има и други
- Един процесор може да изпълнява само една нишка в даден момент
  - Ако имаме 42 нишки ни трябват 42 процесора, ако искаме да вървят едновременно
- Планиране (scheduling) - процесорното време се разпределя между нишките, като работата им се превключва достатъчно бързо
- Context Switch - смяна на текущо работещата нишка

Планиране - стратегии

Cooperative Scheduling

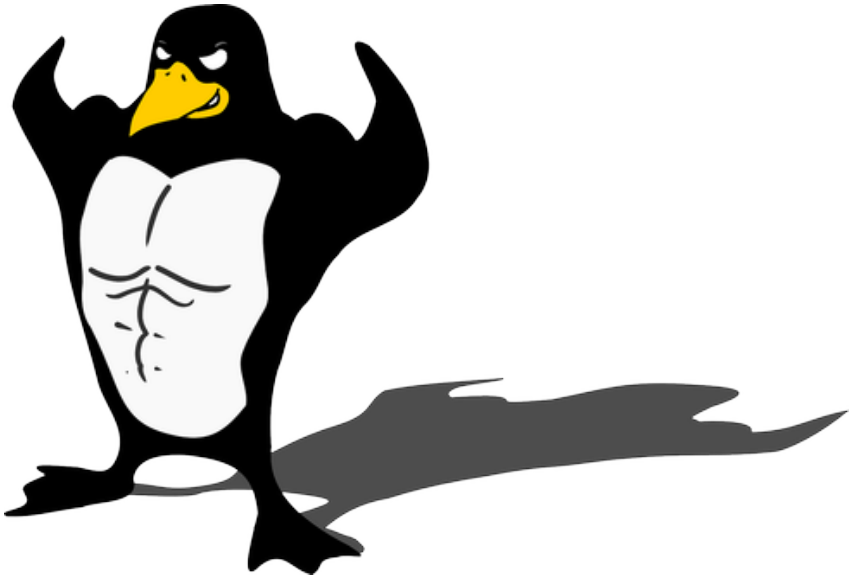
- Всяка нишка се грижи да работи “малко” и след това сама да предаде контрола на операционната система, която може да пусне друга нишка да се изпълнява
- Бързо и лесно за имплементация
- Ако една нишка зацикли, умираме
- Трудно обработваме събития от хардуера

Планиране - стратегии

Preemptive Scheduling

- Операционната система сама решава кога да прекъсне текущата нишка и да даде контрола на друга
- Една нишка не работи повече от някакво определено време (i.e. 200ms)
- При настъпване на събитие може веднага да се събуди съответната нишка (например, при мърдане на мишката се събужда процесът, който мести курсора)
- По-бавно е и ако преразпределянето се случва прекалено често, може да губим много време в него вместо в реална работа

А сега в Linux



Процес

- PID (Process ID)
- priority and niceness
- memory
- security context
- environment
- file descriptors

Йерархия на процесите

- init (PID 1) - (пра-)родител на всички процеси (нещо като Чингиз Хан)
- При стартиране, ядрото стартира точно един процес - init.
- Всеки процес може да има деца-процеси
- Всеки процес е дете на точно един друг процес (родител)
- `$ pstree` - показва дървото на процесите

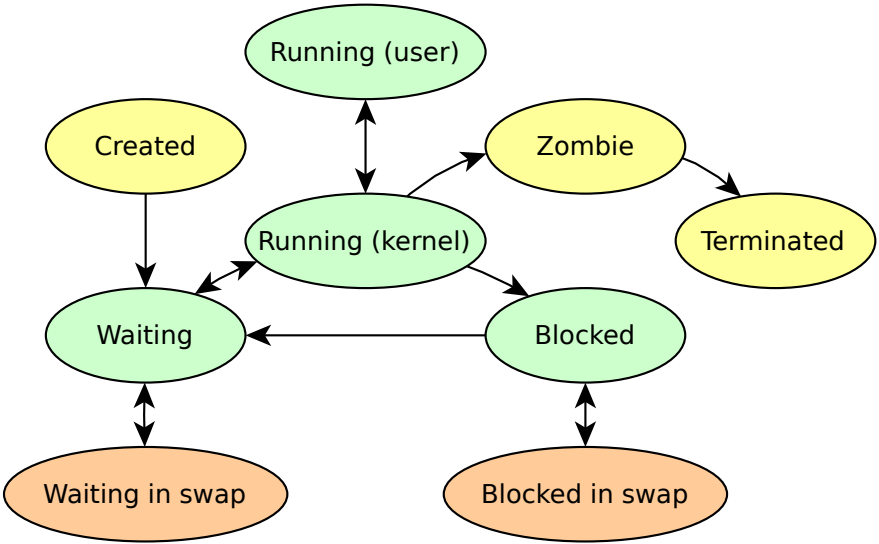
Създаване на процеси

- `fork()`

Изпълняване на програми

- `exec()`

Състояния на процесите



Състояния на процесите в Linux

- **R** running or runnable (on run queue)
- **D** uninterruptible sleep (usually IO)
- **S** interruptible sleep (waiting for an event to complete)
- **T** stopped, either by a job control signal or because it is being traced
- **Z** defunct (“zombie”) process, terminated but not reaped by its parent

Приоритети на процесите

- Всеки процес има niceness - [-20, 19]
- Колкото по-малък niceness, толкова по-висок приоритет
- Пускане на процес с даден приоритет: `$ nice -n 15 foo`
- Смяна на приоритета на процес: `$ renice 15 <pid>`

Преглеждане на процеси

- `/proc` - виртуална директория, съдържаща информация за всеки процес
- `$ ps` - user-friendly инструмент за достъп до тази информация
  - `$ ps` - процесите в текущия shell
  - `$ ps -e` - всички процеси
  - `$ ps -ef` - повече информация за всички процеси
- `$ pgrep` - търси процеси по атрибути
- `$ pstree` - красиво дърво на процесите
- `$ top`, `$ htop` - интерактивни “процес-мениджъри”

Сигнали

- Специални съобщения, които могат да се пращат на процесите
- Имат номер (от 0 до 31) и user-friendly име (KILL, TERM, STOP)
- Имат някакво действие по подразбиране
- Повечето могат да се предефинират, така че да се изпълни потребителска функция вместо действието по подразбиране
- Някои не могат да се предефинират (напр. KILL)

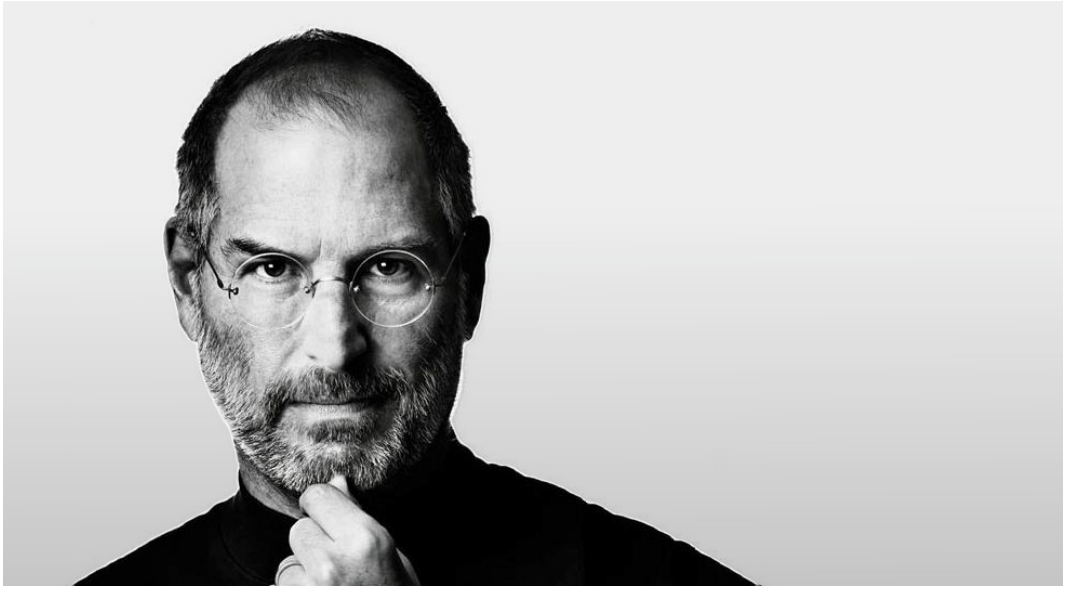
Сигнали, които по подразбиране убиват процеса

- SIGTERM (15) - изпраща се по подразбиране от командата `$ kill`
- SIGINT (2) - при Ctrl-C
- SIGHUP (1) - при затваряне на терминал
- SIGQUIT (3) - при Ctrl-\, убива процеса и показва дебъг информация
- SIGSEGV (11) - при Segmentation Fault
- SIGKILL (9) - убива процеса мигновено, не може да се предотврати “мигновено” в повечето случаи, освен ако не сме в kernel mode

Пращане на сигнали

- `$ kill <pid>` - праща сигнал до процеса с дадения PID. По подразбиране праща TERM.
  - `$ kill -KILL <pid>` или `$ kill -9 <pid>` - пращане на специфичен сигнал по име или номер
  - `$ kill -l` - показва всички възможни сигнали
- `$ killall <name>` - праща сигнал до процес по име
- `$ pkill <pattern>` - праща сигнал до процесите, които мачват израз
- Ctrl-C - праща SIGINT на текущия процес в терминала

Jobs



Jobs

- Процесите в shell-а имат 3 състояния
  - Running (in foreground) - работи в шела
  - Stopped - паузиран във фонов режим
  - Running in background - работи във фонов режим
- Всеки процес може да се паузира (SIGSTOP) и да се рестартира (SIGCONT)
- Паузирането на процес го откача от shell-а
- След рестартиране, процесът продължава да се изпълнява във фонов режим

Работа с jobs в терминала

- `$ foo &` - стартира командата `foo` във фонов режим
- Ctrl-Z паузира текущия процес във фонов режим
- `$ jobs` - показва всички паузирани или фонови процеси в текущия shell
- всеки фонов или паузиран процес си има job ID в shell-а си
- `$ fg <id>` (foreground) - връща фонов или паузиран процес в преден режим
- `$ bg <id>` (background) - праща паузиран процес във фонов режим