Процеси

08 октомври 2020 г. 13:42

- Процеса е група инструкции заредени в паметта.
- Инструкциите идват от програма която работи.

- В информатиката, процес се нарича програма в етап на изпълнение от многозадачна операционна система.
- За разлика от процеса, програмата е просто пасивен набор от инструкции съхраняван в някаква памет.
- След като програмата се зареди в оперативната памет и бъде стартирана, вече става въпрос за процес.
- Той е съвкупност както от кода и данните които обработва и намиращи се в оперативната памет, така и от служебните структури от данни, свързани с него, които се генерират и управляват от операционната система (ОС).

- След като линукс ядрото зареди, прехвърля контрол върху процес с име PID 1.
- PID е акроним за process id. Преди (на по стари системи) този процес е бил init.
- PID 1 е родителският процес всички след него са child процеси.
- След като зареди някой процеси ще станат "родители" на други процеси, а те ще се превърнат в child (деца) на тези процеси.

За да видим процесната йерархия можем да използваме командата:

Ps:

- За да видим процесите на системата използвайки стандартният и синтакс.
- ps -eH | less ==> H е с цел да покажем йерархията на процесите.
- Използвайки само "ps" ще видим процесите в текущата ни конзолна сесия.

- ps -u <username> ще покаже активните процеси на един потребител в системата.

- ps -H --forest ще ни покаже, един вид по разбираемо кой процес е бащин и кой дете.
- ps -eHf ще ни покаже всички процеси, плюс аргументите на командите докато работят. На кратко форматира всичко така

Сега идва въпросът от къде идва тази информация?

От /proc папка която е в директна комуникация с ядрото.

Ядрото изпраща информация от runtime config и с какво работи в proc директорията.

free, top и други команди подават тази информация на потребителя.

ps покзва един вид заснета, като снимка на процесите които работят през това време.

Ако обаче подкараме top ще видим процесите на живо.

Мениджмънт на процеси.

uptime

Поглежда от колко време системата работи. Колко потребителя има логнати. И какво е натоварването върху процесор:

- [nick@localhost etc]$ uptime

22:02:58 up 13 min, 2 users, load average: 0.00 (последната мин) , 0.05 (последните 5 мин), 0.06 (последните 15 мин)

- load average е средният брой процеси които са в runnable state (използва процесора или чака да го ползва) и uninterruptible (непрекъсваемо) state чака някакъв I/O достъп (мрежа или хард диска) а.

Ако имаш 1 процесор с едно ядро и load average е 1 значи процесора е препълнен ако за 4 ядра на процесора и е 1 3/4 са свободни.

free

Показва ползваното и свободната рам памет. Както и swap местото.
free -h => показва в human readable

pgrep

Намира информация за процеса по именно.

sudo pgrep -a httpd

Показва и пътя от къде е генериран процеса.

kill

Изпраща терминиращ сигнал по PID (SIGTERM)

Сигнали:

SIGHUP 1

SIGKILL 9 е все едно вземаш чука и го удряш. Ако 15 не затвори файловете като хората 9 е твоят приятел.

SIGTERM 15 Спира процеса gracefully което означава, че ще спре всичко внимателно и правилно папки и др. Добър начин за затваряне на програми.

sudo pgrep httpd

sudo kill httpd

Вземайки най дъртият процес, спирайки го ние спираме цялата програма.

Ако убием един процес, от дъщерните, то това ще накара httpd сервиза да възобнови процеса.

kill -l

pkill -x httpd

Запазване на процеси в движение докато не работим със системата.

- killall убива всички процеси свързани с аргумента ни.
- killall -s 9 със сигнал

watch :

Периодично можем да гледаме команда.

- watch date
- watch ls за да гледаме дали нещото което сваляме се сваля
- watch -n 1 systemctl status httpd следим риъл тайм апаче

screen

Може да се ползва с цел да започнеш сесия и пуснеш някоя команда и да напуснеш сесията като се откачаш от нея и да се закачиш отново с цел да поддържаш командата в движение целта е да държим сесията жива

Стъпки за да се постигне:

1. Инсталираме screen добавяйки:
2. sudo dnf install epel-release-8-8.el8.noarch
3. След това пускате watch -n 1 date
4. Ctrl +a +d излизате от тази сесия.
5. Reatach --> screen -r
6. screen -ls
7. screen -r <screen id>

tmux подобно на screen

tmux ls

tmux attach-session -t <number of session>

nohup ping google.com & ще изпрати командата в бекграунда

с jobs ще я видим

%<job number> отваря процеса

tail -f nohump.out

jobs –l

Ctrl + Z ще паузира работата.

bg %1 ще я върне в работа

можем да спрем процеса и със kill PID

Приоритети в процесите:

Всички процеси използват така нареченото процесно време - CPU time.

Присвояване на приоритет диктува колко време ще бъде дадено на процес от самият процесор.

Как процеса бива оценнен:

- nice levels:

-20 highest priority върху процеса

19 lowest най-малко

0 default обикновено

- Може да се промени само с root.

ps -o pid,nice,cmd,user

- nice команда ползвана за поставяне на nice ниво на процеса преди апликацията да е стартирана.
- renice е командата която променя това ниво когато приложението (сервиза) работи.

nice -n 5 watch -n 3 free -m &

ps -o pid,nice,cmd,user

- renice -n -1 300823

ps -o pid,nice,cmd,user

bg: Връща процес от бекграунда (отзад) .

fg: Директно го отваря, хвърля го отпред.