24. Команди ps, free, ls, proc, meminfo, vmstat и други

В UNIX базираните системи, каквато е и Linux Mint, съществуват някои основни понятия, с които се обозначават различните стартирани процеси:

процес - изпълнявана програма, заредена в паметта и управлявана от ядрото; всеки процес има идентификатор - PID;

задача - процес, стартиран от шела (на заден фон или прекъснат - използва се Ctrl+Z); номерът на задачата се различава от идентификатора (PID) на процеса;

daemon - процес, който се изпълнява на заден фон до настъпване на определено събитие; за аналогия, в Windows това са услугите на ОС;

cron job - задача, която се изпълнява по точно определен график; аналогично, в ОС Windows това са задачите, които се създават с Task Scheduler.

За управление на процесите могат да се използват някои основни команди. Една от тях е **ps**, която показва текущо изпълняваните процеси в табличен вид:

PID - идентификатор на процес;

TTY - терминал, на който се изпълнява процесът;

STAT - статус на процеса или къде се намира в паметта;

TIME - процесорно време, което е използвал досега;

CMD - местоположението на програма.

Примери за използване на командата:

ps -C chrome - стартирани процеси (по име);

ps U root или ps -U root - стартирани процеси от потребител root;

ps -G root - стартирани процеси от група root;

ps -ejH; ps axjf - извеждане на дърво на процесите;

ps -eLf; ps axms; ps -eLf | grep chrome -извеждане на информация за нишките на всички процеси или на конкретен процес;

Друга команда, която може да се използва за извежда информация за процесите, е **top.** Предимство е, че работи в реално време и чрез нея може да се проследи кои процеси използват най-много процесорно време и компютърни ресурси. В таблица са отбелязани наименованията и описанието на колоните на изходния резултат от изпълнението на **top.**

Значение на колоните в команда top

Колона	Значение
PID	идентификатор на процеса
USER	собственик на процеса
PR	приоритет
NI	„доброжелателност на процеса“
VIRT	количество виртуална памет, използвана от процес
RES	количество използвана RAM памет
SHR	количество споделена памет, използвана от процес
S	статус на процеса - (D - спящ, без възможност за прекъсване), R - работещ, S - спящ, T - спрян, Z - зомби)
%CPU	използван процент от ресурса на процесора
%MEM	използван процент от RAM паметта
TIME+	количество процесорно време, което процесът е използвал, откакто е стартиран
COMMAND	име на стартираната команда

Примери за използване на команда top:

top -o %CPU - подреждане на процесите по заемано процесорно време;

top -u root - извеждане на информация за всички процеси, стартирани от root;

top | grep chrome - извеждане на информация за конкретна програма.

Показване на свободната и използваната памет чрез free

Въпреки че top включва информация за паметта, безплатният инструмент показва количеството на свободната и използваната памет в системата в килобайти. (Ключът -m превключва показването в мегабайти).

На една система, изходът изглежда по следния начин:

```
matthew@seymour:~$ free
             total       used       free     shared    buffers     cached
Mem:       4055680    3327764     727916          0     280944    2097568
-/+ buffers/cache:     949252    3106428
Swap:      8787512          0    8787512
```

Този изход описва една машина с 4GB RAM памет и виртуален дял с размер 8GB. Забележете, че виртуалната памет не се използва и че машината не е натоварена. Linux е много добра в управлението на паметта и „заграбва" цялата памет, която може, в очакване на бъдеща работа.

Един друг полезен инструмент за контролиране на системи е **vmstat** ***(virtual memory statistics)***. Тази команда докладва за процесите, паметта, I/O (вход/изход) и CPU (процесор), обикновено показвайки усреднени стойности от времето на последното презареждане; а вие можете да я накарате да докладва използването за текущ период, указвайки времевия интервал в секунди и броя на стъпките, които искате да бъдат изпълнени, по следния начин:

matthew@seymour: ~$ vmstat 5 10

Това изпълнява vmstat на всеки пет секунди за 10 стъпки.

По подобие на споменатите вече команди **pgrep** също може да се използва за извеждане на **информация за процес, като го намира по име.**

Друга команда, свързана с управлението на процеси, е **jobs.** Тя показва **текущо стартираните задачи от терминала**. Може да се използва за поставяне на заден фон или преминаване към процес по номер на задача. Примери:

jobs -l - извеждане на списък с всички задачи, включително и техните идентификатори;

bg % jobID - поставяне на задача на фонов режим, където jobID е нейният идентификатор;

fg % jobID - извеждане на задача от фонов режим, където jobID е нейният идентификатор.

За извеждане на **общото процесорно време**, използвано от даден процес, а също и времето в потребителски и системен режим (изпълнение от ядрото), може да се използва командата **time.** Например **time wc /etc/passwd.**

Част от управлението на процеси е тяхното извеждане от паметта (прекратяване или „убиване“). Основната команда, която се използва, е **kill.** Могат да се посочат някои примери за практическото й приложение:

kill <pid> - „убива“ процес по неговия идентификатор;

kill -9 -l — „убива“ всички стартирани процеси;

kill -l - извежда списък на всички сигнали.

Други аналогични команди са:

killall - „убива“ процес по име;

pkill - „убива“ процес по име.

Използване на съдържанието на директорията /proc за взаимодействие с ядрото

Съдържанието на директорията /proc се създава от паметта и съществува само по време на работата на Linux. Тази директория съдържа специални файлове, които или извличат информация, или изпращат информация към ядрото. Много инструменти на Linux извличат информация от динамично създавани директории и файлове в тази директория, известна също като виртуална файлова система (virtual file system). Например, командата free получава своята информация от файл, който се казва meminfo:

```
matthew@seymour:~$ free
             total       used       free     shared  buffers  cached
Mem:       4055680    2725684    1329996          0   188996  1551464
-/+ buffers/cache:     985224    3070456
Swap:      8787512          0    8787512
```

Тази информация се променя постоянно по време на използване на системата. Вие можете да получите същата информация, като използвате командата cat, за да покажете съдържанието на файла **meminfo:**

matthew@seymour:-$ **cat /proc/meminfo**

"**meminfo**" дава подробен списък на процесите, които се случват с паметта. Най-често вие достъпвате meminfo данните при работа с други команди, като "**cat**" (*съкратено от "concatenate"*) или „**grep**" ("global regular expression print"). Ако например използвате „**cat/proc/meminfo**", ще имате достъп до информация, която показва това, което се случва в паметта на вашия сървър във всеки даден момент. За достъп до по-обща информация е по-удобна командата „**free**". Това е и разликата между двете команди.

Директорията /proc може също да бъде използвана за динамична промяна на поведението на ядрото на работеща Linux, чрез „повтаряне като exo" на числовите стойности към специални файлове в директорията /proc/sys. Например, за да бъде „включена" защитата на ядрото срещу един тип атака от типа „отказ от услуга" (denial-of-service - DoS), наречена също „SYN flooding", използвайте командата echo, за да изпратите числото 1 към следния път в /proc:

matthew@seymour:**sudo echo 1 >/proc/sys/net/ipv4/tcp_syncookies**

Други начини за използване на директорията /proc включват следните:

Получаване на информация за CPU, като например семейството, типа и скоростта от /proc/cpuinfo.

Разглеждането на важна мрежова информация в /proc/net, като например информация за активните интерфейси в /proc/net/dev, информация за маршрутизирането в /proc/net/route и мрежова статистика в /proc/net/netstat.

Извличане на информация за файловата система.

Помощна информация за точка за монтиране на носител, свързан през USB, например, Linux ядрото докладва кое устройство трябва да се използва за достъп до файлове (като например /dev/sda), ако USB камера или твърд диск бъдат открити в системата. Вие можете да използвате командата dmesg, за да видите тази информация.

Получаване на версията на ядрото от /proc/version, информация за производителността, като например времето от стартиране на системата в /proc/up- time, или друга статистика, като например натоварване на процесора, използване на файла на виртуалната памет (swap file) и процесите в /proc/stat.

Командата ls извежда **<u>съдържанието на директория.</u> Ако се изпише без опции, показва съдържанието на текущата директория.** Някои полезни опции на командата са следните:

-**a** - извежда имената на скритите файлове;

-**f**- извежда съдържанието на директорията, без да сортира;

-**h** - извежда размера в килобайти и мегабайти;

-**1** - извежда списък за вида на файловете, правата за използването им, собствениците им, времето на последното модифициране;

-**t** - сортира файловете според времето на модифициране;

-**u** - сортира файловете според времето на достъп;

-**R** - извежда рекурсивно съдържанието на директориите и техните поддиректории.

<u>Примери за приложението й:</u>

ls -1 /home - извежда съдържанието на директория home с правата за достъп до файлове и поддиректории;

ls -al ≡ ls -la - извежда съдържанието на текущата директория, включително скритите файлове и правата за достъп;

ls - i - извежда съдържанието на текущата директория заедно с индексния номер на всеки елемент;

ls -u -1 - извежда съдържанието на текущата директория, като подрежда елементите по име и дата на последен достъп.

mpstat

Командата „**mpstat**“ докладва дейността на всеки от активните процесори при мултипроцесорен сървър. В наши дни навсякъде се използват многоядрени процесори, а командата е валидна за почти всички видове сървърни конфигурации. Тя позволява да се наблюдават общите статистики за системата или за отделния процесор. Подобна информация ви позволява да забележите възможните проблеми с приложенията, преди потребителите да са ги видели.