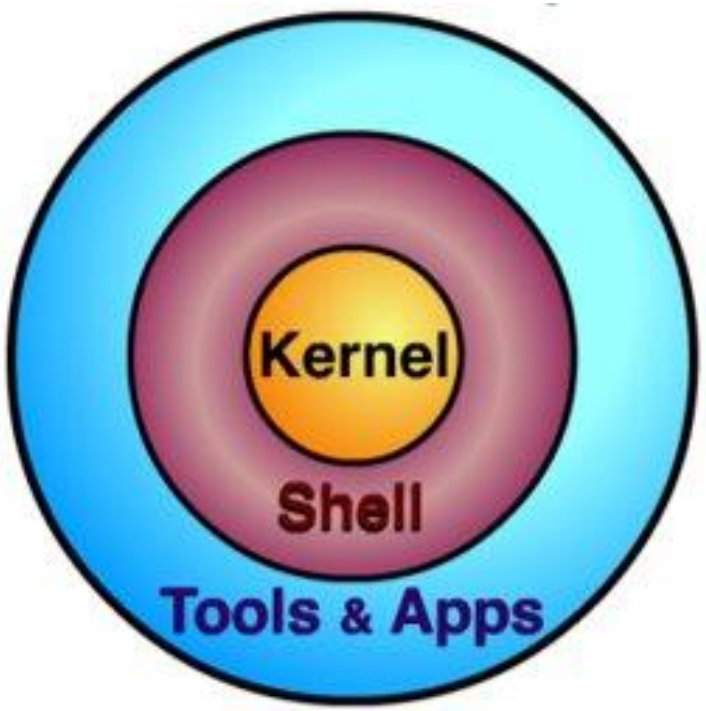
Основи на UNIX

Оперативни системи

Аудиториска вежба 1

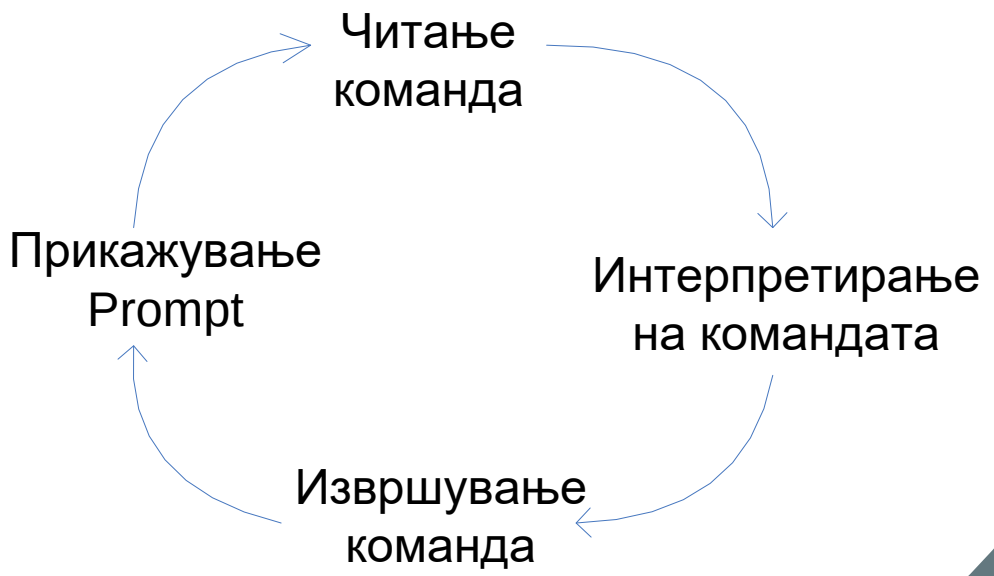
Kernel

- Срце (јадро) на UNIX
  - Ги менаџира ресурсите
  - I/O менаџирање
  - Менаџирање на процеси
  - Менаџирање на уреди
  - Менаџирање на датотеки
  - Менаџирање на меморија



Shell (Кориснички дел)

- Најважната програма од гледна точка на корисникот
- Интерфејс помеѓу корисникот и јадрото
- Командeн интерпретер
- Бесконечна јамка
- Типови: Bourne, C, Korn, Bash
- Прикажување на типот на shell:
  - `$ echo $SHELL`
- Ограничен број команди



Подесување на работна околина

- Секој студент има корисничко име и лозинка на ФИНКИ сервер наменет за работа на овој курс:

  host: os.finki.ukim.mk

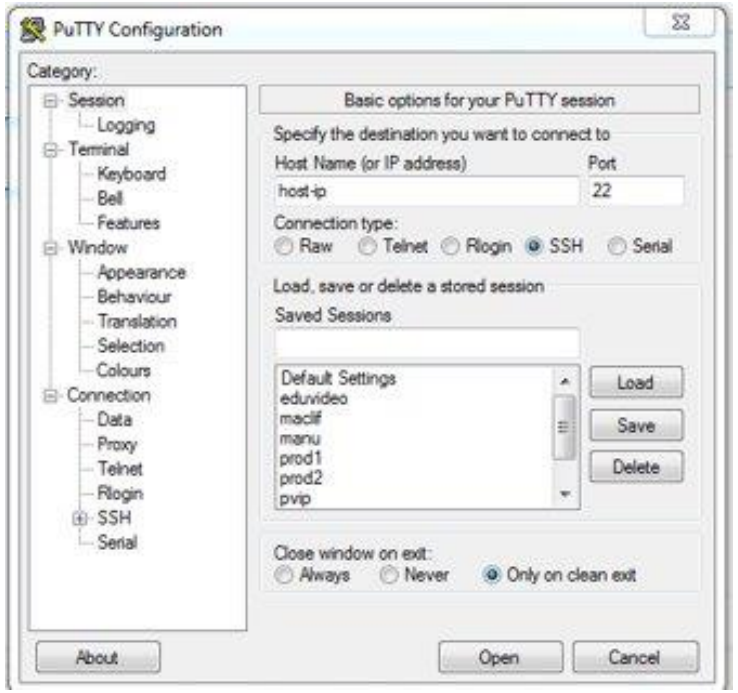
  port: 22

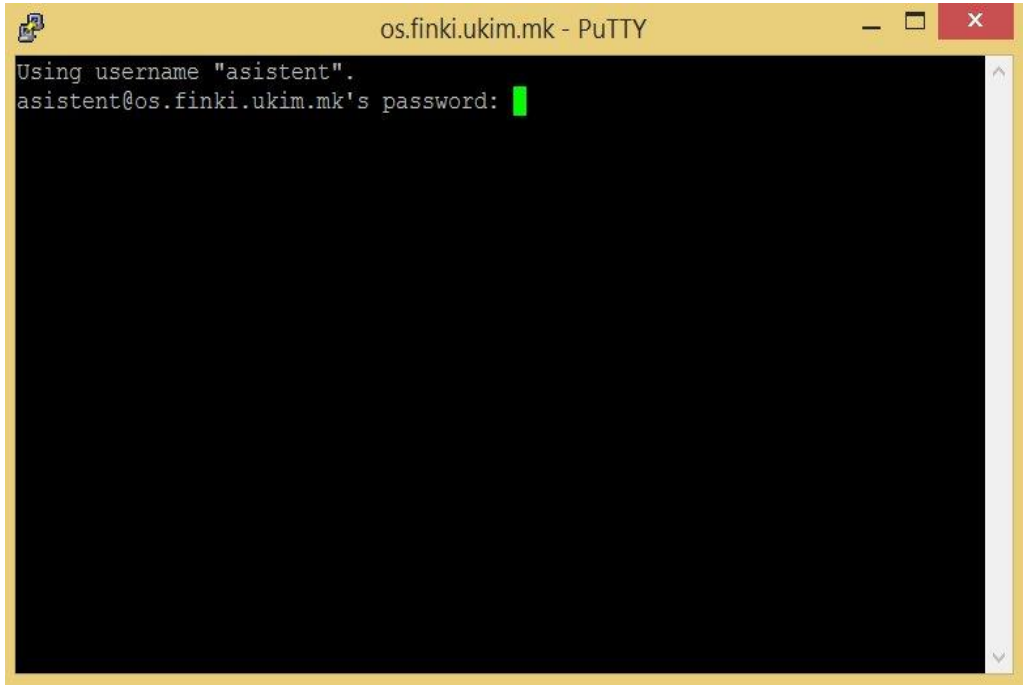
  username: *CAS username*

  password: *CAS password*

Подесување на работна околина

- Доколку се користи Windows мора да се употреби ssh клиент како што е PuTTY





- Доколку се користи Linux или друг *nix систем сосема веројатно е дека дистрибуцијата веќе има инсталирано ssh клиент

  ssh username@host

Команди во UNIX

- Командите кај UNIX обично претставуваат извршни програми ко и интерпретерот ги пронаоѓа и ги извршува.
- Синтакса на Unix команди:
  - **$ име_команда -*опции аргументи***
- Секоја команда се состои од:
  - **Име на командата** (кај UNIX се прави разлика помеѓу мали и големи букви -
    командите мора да се задаваат точно онака како што се опишани)
  - **Опции**
    - од облик -x, каде x е некој карактер
    - Помеѓу '-' и 'x' не смее да постои празно место.

Команди во UNIX

- Аргументи може да бидат имиња на датотеки или изрази
- Пример:
  - `$ ls –a OOS`
  - **`ls` претставува команда за листање на содржина на именик, а `e` опција со која се бара листање** на сите датотеки, а OOS е аргумент со кој се одредува именикот кој ќе се листа.
- Една команда може да има повеќе опции кои го одредуваат начинот на кој ќе се изврши командата.
  - `$ ls –a –l OOS`
  - `$ ls –al OOS`

Help

- Помош во Unix: команда man (manual)
  - `$ man command_name` (пример: `man ls`)
- Постојат неколку правила за толкување на синтаксата на команд ите:
  - Опциите и аргументите кои се во средни загради [] се опциони.
  - Сè што не е во средни загради мора да се внесе.
  - Името на командата и опциите мора да се испише точно онака како што е дефинирано со синтаксата.

  Аргументите се заменуваат со она на што се однесува командата, на пример име на датотека или именик.
  - '...' значи дека е можен произволен број на повторувања на претходниот аргумент.

Работа со датотеки и именици

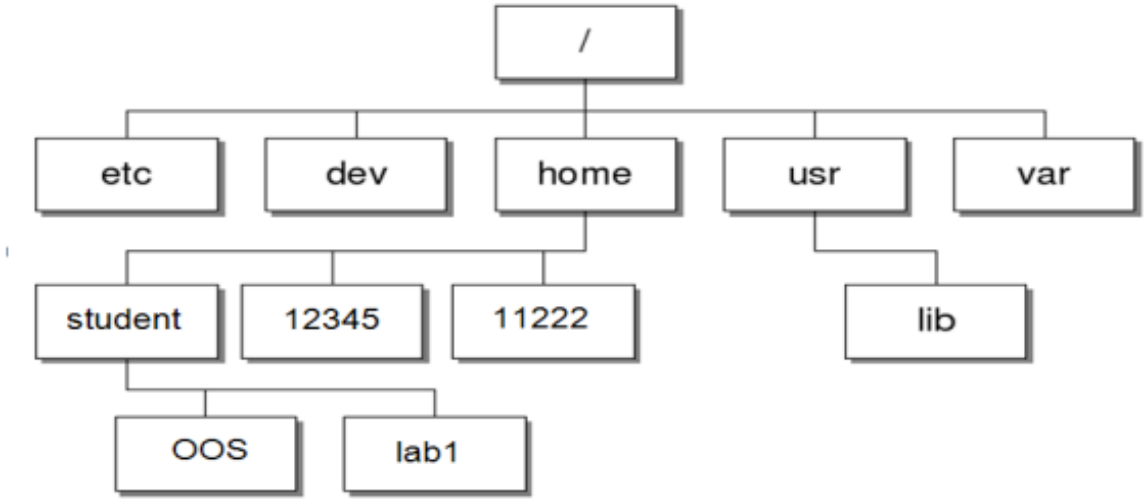
- Директориум е всушност датотека во која се чува листа од други датотеки. Листата не може да содржи други податоци.
- Секој корисник на системот има свој именик (home directory). По најавување на системот, тој станува тековен.

- Командата со која се прикажува кој е тековниот именик е:

  $ pwd

Работа со датотеки и именици

- Кај UNIX постои хиерархиска структура од именици во која се организирани датотеките.
- На врвот од структурата на имениците се наоѓа еден посебен именик наречен коренски именик (root).



Работа со датотеки и именици - Именување

- **Дозволено**: мали и големи букви (case sensitive), цифри и некои специјални знаци (_ # @).
- **Забрането**: не смее да содржи празни места или некои од метазнаците на школката (* ? > < | / ; & ! [ ] $ \ ’ “), не смее да почнува со знаците + или - и не смее да биде исто како системска команда.

- Датотеките чие име почнува со знакот . се скриени (не се гледаат во листањето добиено со ls или ls –l).
- Може да се видат со опцијата –a

  $ ls -a

Промена на работниот именик - cd (change directory)

- $ cd destinacija - тековниот работен именик се променува во destinacija.

  $ cd proekti (или cd /home/student/proekti)

  $ pwd

  /home/student/proekti

  $ cd ..

  $ pwd

  /home/student - се променува работниот именик така што се качуваме едно ниво погоре (кај родителот на претходниот тековен именик).

- cd ~ или cd (без параметри) позиционира во домашниот именик на корисникот

Прелистување на датотеки во именик - ls (list)

- **ls** -> ги прикажува сите датотеки од работниот именик (без специјалните што почнуваат со точка [.]).
- **ls -a** -> ги прикажува сите датотеки од работниот именик вклучувајќи ги и специјалните што почнуваат со точка [.].
- **ls -s** -> ја прикажува содржината на работниот именик и големините на датотеките во килобајти.
- **ls | more** -> ако има повеќе датотеки во именикот, да ги прикаже страна по страна.
- **ls –l** -> ги прикажува датотеките од работниот именик во т.н. долг формат: вклучувајќи ги и дозволите за датотеките, големината и датумот на креирање.

ls -l

(1)	(2)	(3)	(4)	(5)	(6)
total 109					
-r--------	1	student	1382	Oct 18 13:15	Ne_me.brisi
-rwxr-xr-x	1	student	68064	Mar 23 2001	a.out
-rw-r--r--	1	student	1144	Oct 18 13:01	aaa.txt
drwxr-xr-x	2	student	512	Oct 18 13:17	dat
-rwxr-xr--	1	student	28672	Oct 18 13:03	kmml
-rw-r--r--	1	student	374	Oct 30 2000	kmml.p
-rw-r--r--	1	root	299	Oct 25 1999	pj
-rw-rw-r--	1	student	188	Oct 18 13:07	prijava.txt
drwxr-x---	2	student	1024	Apr 19 2000	prog
-rw-r--r--	1	student	432	Mar 23 2001	record1.p
-rw-r-----	1	student	872	Mar 31 1995	rezultati.txt
drwxr-xr-x	2	student	512	Apr 19 2000	string
lrwxr-xr-x	1	student	15	Oct 18 13:10	strlib.h -> string/strlib.h
-rw-------	1	student	106	Mar 20 1997	tajna.txt

- (1) го прикажува типот и дозволата за пристап до датотеката
- (2) број на врски кон датотеката;
- (3) сопственик на датотеката (owner);
- (4) должина на датотеката во бајти;
- (5) време на последната промена на датотеката:

  доколку датумот е од тековната година – датум и време

  доколку датумот не е од тековната година – датум и година
- (6) име на датотеката;

ls –l (дополнување)

- **type**: тип на датотека:

  ‘-‘ обична датотека; ‘d’ именик; ‘l’ врска (link); (има и други видови)

- **user**: права на пристап до датотеката за сопственикот на датотеката;
- **group**: права на пристап до датотеката за корисниците од истата група на корисници на која припаѓа сопственикот на датотеката;
- **other**: права на пристап до датотеката за сите останати корисници на системот;

- **<u>Дозволи за пристап до датотеката:</u>**
  - **r -дозвола за читање;**

    датотека - дозвола да се гледа содржината на датотеката

    именик - дозвола да се прелистува содржината на именикот
  - **w -дозвола за запишување;**

    датотека - дозвола да се менува содржината на датотеката

    именик - дозвола да се менува неговата содржина (додаваат или отстрануваат имениците и датотеки)
  - **x -дозвола за извршување;**

    датотека - дозвола за извршување на датотека

    именик - дозвола да тој стане работен именик (може да се дојде до него)

Промена на привилегии - chmod

- $ chmod - ги менува привилегиите на датотеките и имениците
- Синтакса: chmod [options] mode file

- Примери:

chmod ug+rw sample

chmod +r file

chmod -x file

chmod u=rw,go= file

chmod +rw file chmod -R u+w,go-w docs/

chmod 666 file chmod 755 file

chmod -R u+rwX,g-rwx,o-rwx <directory>

Нумеричко претставување привилегии

- Имено, со број од 1 до 7 може да се зададе секоја можна комбинација на привилегии.
- Тој број може да се добие ако на местото на секој знак во атрибутот се стави 1 ако привилегијата е доделена, а 0 во спротивно.
- Така, бинарната репрезентација на rw-rw-rw- е 110110110, а тоа во октален запис е 666
- Следната табела може да послужи како правило за формирање на соодветниот број за саканата привилегија:

+	u	g	o
r	4	4	4
w	2	2	2
x	1	1	1

Промена на привилегии

- Пример 1:

$ chmod ugo=rw test.txt

$ ls –l test.txt

Излез: Привилегиите на датотеката test.txt се: -rw-rw-rw-

- Пример 2:

$ chmod 640 test.txt

$ ls –l test.txt

- Кои се привилегиите на датотеката test.txt?

Креирање на датотеки

- Датотеки можат да се креираат со уредувачи на текст (Vim, gEdit, Nano, Pico, …) или со користење на UNIX команди и пренасочување на стандардниот излез.
- Пример:
- $ nano dat.txt

  Се отвара Nano едиторот за креирање нова датотека. Откако ќе се внесе содржината на датотеката со Ctrl+X се излегува од едиторот
- $ cat > start.txt
- Откако ќе се внесе содржината на датотеката со Ctrl+D се прекинува cat командата, а текстот е зачуван во датотеката со име start.txt. И вака може да се случи да се избрише датотеката ако таа веќе постои.

Копирање на датотеки - cp (copy)

- $ cp izvor odrediste -> креира копија на датотеката од изворот на одредишната дестинација.
- $ cp star nov -> во истиот именик се прави копија на star под ново име nov;
- $ cp star.txt /users/student -> star.txt од работниот именик се копира во именик со патека /users/student под истото име;
- $ cp res.01 res.02 res.03 work/ -> се копираат трите датотеки во именикот work, под оригиналните имиња;

Копирање на датотеки - cp (copy)

- Опции:
- -i, интерактивен режим кој важи за суперкорисник;
- -f, форсиран режим;
- -l, наместо копија се креира hard link;
- -s, наместо копија се креира симболички линк;
- -r, рекурзивно копирање за обични датотеки;
- -p, опција preserve;
- -b, се креира резервна копија;

Преместување на датотеки и именици - mv (move)

- $ mv izvor odrediste
- Изворот може да биде и датотека и именик.
- Дестинацијата е патека (апсолутна или релативна) до новиот именик каде се врши преместувањето.
- Ако се наведе ново име на датотеката (именикот), таа ќе биде преместена под новото име.

Преместување на датотеки и именици - mv (move)

- $ mv moj.txt /users/student -> moj.txt се преместува во /users/student под истото име.
- $ mv moj.txt /users/student/mojnov.txt -> moj.txt се преместува во /users/student под ново име mojnov.txt
- $ mv student /users/admin -> именикот student се преместува во /users/admin под истото име.

- $ mv proba.txt moj.txt -> proba.txt се **преименува** во moj.txt

- $ mv staroime novoime -> важи и за датотеки и за именици.

Бришење на датотеки и именици - rm (remove)

- $ rm ime -> за датотеки и празни именици
- $ rm moj.txt -> moj.txt се брише
- $ rm -r imeimenik -> за оваа команда е потребно да се наоѓаме во именикот родител на оној што се брише. Со ова ќе се избришат и сите подимениции и датотеки на конкретниот именик.
- $ rm –r student -> се брише именикот student, со целата своја содржина (-r е за рекурзивно бришење);
- $ rm res.01 res.02 -> ги брише двете датотеки;

Прикажување на датотека на екран

- $ cat file
- $ cat proveri.p

- Можни опции:
- -b ги игнорира празните линии и го нумерира секој ред
- -n ги нумерира сите линии (вклучувајќи ги и празните)
- -s ги отфрла двојните празни редови

cat

- Оваа команда има повеќе начини на користење.
- Ќе покажеме дека може да ја искористиме за додавање на содржина на крајот на датотеката, со користење на операторот >>.
- Пример:

```
[       @os ~]$ cat >> results.csv
Add line 1 to the end.
```

```
[       @os ~]$ cat results.csv
Index,Name,Surname,Points
112233,Alex,Jones,89
117899,Don,Malik,70
123456,Sarah,Peterson,60
178999,Peter,Smith,67
199887,Luke,Jones,65

Add line 1 to the end.
```

cat

- Следно, ќе покажеме дека може да ја искористиме за копирање на датотеки, со операторот >.
- Пример:

```
[        @os ~]$ cat results.csv > res.csv
```

```
[        @os ~]$ cat res.csv
Index,Name,Surname,Points
112233,Alex,Jones,89
117899,Don,Malik,70
123456,Sarah,Peterson,60
178999,Peter,Smith,67
199887,Luke,Jones,65

Add line 1 to the end.
```

cat

- Оваа команда има повеќе начини на користење. Следно, ќе покажеме дека може да ја искористиме за спојување на содржината од повеќе датотеки.
- Пример:

```
[       @os ~]$ cat one.txt
This is the first file.
[       @os ~]$ cat two.txt
This is the second file.
[       @os ~]$ cat one.txt two.txt
This is the first file.
This is the second file.
[       @os ~]$ cat one.txt two.txt > three.txt
[       @os ~]$ cat three.txt
This is the first file.
This is the second file.
```

Броење на редови, зборови, знаци во датотека - wc (word count)

- $ wc file
- Збор е максималната низа од знаци одвоена со празни места, таб карактери или знаци за нов ред.
- Пример:

$ wc proveri.p

14 31 190 proveri.p

- Опции:
- –l прикажува само број на редови
- –w прикажува само број на зборови
- –c прикажува само број на знаци

Отсекување на делови од датотека - cut

- $ cut –b lista file
- $ cut –c lista file
- Да се отсечат (земат) податоците дефинирани со lista од секој ред од датотеката.
- Да се земат првите 66 знаци од секој ред од датотеката file.txt:

  $ cut –c 1-66 file.txt

Останати корисни команди - find

- Корисна команда кога сте сигурни дека имате датотека со одредено име, но не ја знаете патеката.
- $ find / -name ispiti -print
- $ find / -name '*alpha*' -print
- $ find / -mtime -5 -print
- $ find / -size +2000 -print
- $ find / -name alpha -size +50 -mtime -3 -print
- $ find / -name core -exec rm -f {} \;
- $ find / -type d -print

Останати корисни команди - touch

- Оваа команда може да се искористи за да се креира нова празна датотека, како и за да се промени времето на пристап и измена на датотеката во моменталното време.

- Начин на користење: touch [option]… file...

- Пример: (ќе се креира празен фајл 'file.txt')
- $ touch file.txt

Команди за работа со процеси

- Процес - инстанца на компјутерска програма која се извршува од една или повеќе нитки
- Process-ID
- Стартување процес (команда, задача) во позадина:

  $ komanda &

  Командата ќе се стартува и ќе работи во позадина

Приказ на активни процеси: ps (process status)

- $ ps [-options]
- Командата ps овозможува прикажување на статусот на сите тековно активни задачи испратени дп јадрото (процеси)

  пр1. $ ps

  пр2. $ ps -ef
- Командата ps без аргументи ги прикажува само процесите на корисникот што ја повикува

  Оваа команда прикажува комплетна листа за сите тековно активни процеси
- Опцијата -e овозможува да се прикажат сите процеси (не само оние на корисникот), додека опцијата -f прикажува детален приказ на процесите

  пр3. $ ps –ef | grep korisnik

Корисни команди за работа со процеси

- Листа на активни процеси: **$ jobs**
- Прекинување (убивање) на процес: **$ kill**

  Насилно прекинување на активен процес: $ kill PID
- Ctrl-C

  Го прекинува активниот процес (го убива)
- Ctrl-Z

  Го испраќа активниот процес во позадина

  (и го прекинува но не го убива)
- bg PID

  Го стартува процесот да се извршува во позадина
- fg PID

  Го стартува процесот да се извршува во преден план
- stop PID

  Го прекинува процесот кој се извршува во позадина

PIDOF команда

- pidof е команда во Linux која се користи за пронаоѓање на ID на процесот (PID) на програма која работи во моментот
- Тоа е едноставна команда која го зема името на програмата како аргумент и го враќа PID на процесот што одговара на тоа име
- Синтакса: pidof [options] program_name

  program_name е името на програмата за која сакаме да го најдеме PID
- Командата pidof го бара процесниот ID на програмата со гледање во табелата со процеси што ја одржува кернелот Linux и враќа PID на сите процеси што одговараат на даденото име

PIDOF команда

- Некои од вообичаените опции што може да се користат со командата pidof се

  -s: Оваа опција се користи само за прикажување на PID на најстарата вклучена инстанца на наведената програма

  -x: Оваа опција се користи за да одговара на целосната патека на програмата, а не само за името на програмата

  -c: Оваа опција се користи само за прикажување на бројот на PID-а кои одговараат на името на програмата, наместо самите PID-и

PIDOF команда - примери

1. Да се пронајде PID-от на програмата со име firefox

```
$ pidof firefox
```

Ќе го врати PID-от на сите инстанци што работат на firefox

2.`Да се пронајде PID на најстарата вклучена инстамца на firefox

```
$ pidof -s firefox
```

Ќе го врати само PID-от на најстарата вклучена инстанца на firefox

3. Да се пронајде бројот на инстанци кои работат на firefox

```
$ pidof -c firefox
```

Ќе го врати бројот на работи на примероци на firefox

4. Совпаѓање на целосната патека на програмата

```
$ pidof -x /usr/bin/firefox
```

Ќе врати PID на сите инстанци што работат на firefox и одговараат на целосната патека /usr/bin/firefox

PMAP - Пмапа

- Командата pmap е алатка која е достапна на оперативните системи базирани на Linux и Unix
- pmap се користи за прикажување на мемориската мапа на процес или збир на процеси
- Излезот од командата pmap ги прикажува мемориските адреси, дозволите и мапираните датотеки на секој мемориски регион што се користи од наведениот процес или процеси
- Пример на синтаксата на командата pmap

  pmap [options] [pid | command [args]]

PMAP - Пмапа

"options" - Опциите го менуваат излезот на командата pmap

Некои вообичаени опции вклучуваат:

- -d: Прикажете ги мапирањата на уредот за процесот
- -q: Тивок режим. Прикажувајте ја само адресата и дозволите за меморијата
- -x: Прикажува продолжен формат, кој вклучува дополнителни информации како што се поместувањето на меморискиот регион и уредот или датотеката на кои е мапиран регионот
- pid: ID на процесот што сакате да го проверите. Ако наведете PID, pmap ја прикажува мемориската карта за тој процес
- command [args]: Ако не наведете PID, можете да наведете команда и нејзините аргументи. pmap ќе ја изврши командата и ќе ја прикаже мемориската карта за добиениот процес

PMAP - опции

- Опцијата -d вклучува дополнителна колона за мапирање на уредот
- Опцијата -q дава поконцизен излез на командата
  - Користењето на опцијата -q може да биде корисно кога ви треба брз преглед на мемориската мапа за некој процес, без дополнителни детали што се обезбедени од стандардниот излез pmap
- Опцијата -p со командата pmap
  - Излезот од командата pmap со опцијата -p вклучува дополнителна линија на врвот на излезот што ги прикажува името на процесот и командната линија
  - Користењето на опцијата -p може да биде корисно кофа треба да го идентификувате името на процесот и командната линија за одреден процес, особено ако работите со повеќе примероци од истата програма или ако името на процесот не е веднаш јасно само од PID

PMAP - опции

- Опцијата -x со командата pmap дава проширен формат
  - Прошириниот формат вклучува дополнителни информации, како што се поместувањето на меморискиот регион и уредот или датотеката на кои е мапиран регионот
  - Излезот од командата вклучува дополнителни колони:
    - Го вклучува поместувањето на секој мемориски регион (колона Offset), како и уредот и бројот на инодата на датотеката на која е мапиран секој мемориски регион (Device и Inode колони)
  - Користењето на опцијата може да биде корисно кога ви требаат подетални информации за мапата на меморијата за некој процес, вклучувајќи информации за мапираните датотеки и уреди

PMAP - Пмапа

Пример за користење `pmap -x 1234`

Во овој пример, pmap ќе ја прикаже мапата со продолжена меморија за процесот со PID 1234

Излезот од командата вклучува неколку колони:

- Address: Мемориска адреса на меморискиот регион
- Kbytes: Големината на меморискиот регион во килобајти
- Mode: Дозволите на меморискиот регион (читање, пишување, извршување)
- Offset: Поместување на меморискиот регион (за мапираните датотеки)
- Device: Уредот на кој е мапиран меморискиот регион (за мапираните датотеки)
- Mapped file: Датотеката на која е мапиран меморискиот регион (за мапираните датотеки)

Пример излез од дадената команда:

```
Address           Kbytes     Mode  Offset   Device           Mapped File
----------------  --------  -----  ------  ---------------  ---------------
00400000-00409000      144 r-x--  00000000  fd:01 11241055     /usr/bin/python
00608000-00609000        8 rw---  00008000  fd:01 11241055     /usr/bin/python
```

Командата pmap покажува два мемориски региони за процесот на python со PID 1234. Првиот регион може да се чита и да се изврши, и е мапиран во датотеката /usr/bin/python. Вториот регион може да се запише и не е мапиран во датотека

/proc/<pid>/status-filesystem

- Во Linux, датотечниот систем /proc е специјален виртуелен датотечен систем кој обезбедува информации за системот и процесите што се извршуваат
- Датотеката /proc/<pid>/status е датотека лоцирана во директориумот /proc и содржи детални информации за специфичен процес идентификуван со неговиот процесен ID (PID)
- Секое поле во датотеката /proc/1234/status обезбедува вредни информации за процесот, како што се неговата употреба на ресурси, распределба на меморијата и состојба
- Оваа информација може да се користи за дијагностицирање на проблеми со процесот, следење на неговата употреба на ресурси и оптимизирање на неговите перформанси

/proc/<pid>/status-filesystem

Некои од најважните полиња се следните:

- Name: Името на процесот
- State: Тековната состојба на процесот, како што се „трчање„, „спиење„ или „зомби„
- Pid: ID на процесот
- PPid: ID на процесот на матичниот процес
- VmPeak: Најголемата големина на виртуелната меморија на процесот
- VmSize: Тековната големина на виртуелната меморија на процесот
- VmLck: Количината на заклучена меморија што ја користи процесот
- VmHWM: Најголемата големина на множеството на жител на процесот
- VmRSS: Тековната резидентна сет големина на процесот
- VmData: Големината на податочниот сегмент на процесот
- VmStk: Големината на сегментот на магацинот на процесот
- VmExe: Големината на извршниот код сегмент на процесот

```
Name:   firefox
State:  S (sleeping)
Tgid:   1234
Pid:    1234
PPid:   5678
TracerPid:      0
Uid:    1000    1000    1000    1000
Gid:    1000    1000    1000    1000
FDSize: 256
Groups: 4 24 27 30 46 116 126 999
VmPeak:  4070400 kB
VmSize:  3144208 kB
VmLck:         0 kB
VmPin:         0 kB
VmHWM:    478912 kB
VmRSS:    382336 kB
RssAnon:   170044 kB
RssFile:    96828 kB
RssShmem:       0 kB
VmData:  2684916 kB
VmStk:       136 kB
VmExe:       456 kB
VmLib:     32664 kB
```

/proc/<pid>/statm

- Датотеката /proc/<pid>/statm е датотека која обезбедува информации за користењето на меморијата на одреден процес

- Кога ќе пристапите до датотеката /proc/1234/statm за процес со PID 1234, ќе видите една линија текст со седум броеби одделени со празни места

  Овие броеви ја претставуваат количината на меморија што моментално се користи од процесот на страници

/proc/<pid>/statm

Значење на секој број:

- size: Вкупната големина на процесот во страници, вклучувајќи ги сите страници со код, податоци и споделена меморија
- resident: Бројот на резидентни (незаменети) страници во адресниот простор на процесот. Ова е количината на физичка меморија што се користи во процесот
- shared: Бројот на споделени страници во адресниот простор на процесот
- text: Бројот на страници што содржат извршна цифра
- lib: Бројот на страници што содржат споделени библиотеки
- data: Бројот на страници што содржат податоци и простор на магацинот
- dt: Бројот на валкани страници (модифицирани страници со податоци што треба да се запишат назад на дискот) во адресниот простор на процесот

/proc/<pid>/statm

Секој од овие броеви дава корисни информации за користењето на меморијата во процесот

На пример, резидентната вредност покажува колку физичка меморија моментално се користи од процесот, додека текстуалната вредност ја покажува количината на меморија што ја користи извршниот код на процесот

Важно е да се забележи дека вредностите во датотеката /proc/1234/statm се претставени на страници, а не на бајти. За да ги конвертирате овие вредности во бајти, ќе треба да ги помножите со големината на страницата, што може да се најде во наредбата getcong PAGESIZE

/proc/<pid>/statm

Пример за тоа како може да изгледа датотеката /proc/1234/statm за процес со PID 1234:

```
123456 7890 2345 6789 1234 5678 9012
```

Во овој пример, процесот користи вкупно 123456 страници, од кои:

- 7890 страници се резидентни (т.е. моментално во физичка меморија)
- 2345 страници се споделени со други процеси
- 6789 страници содржат извршна шифра
- 1234 страници содржат споделени библиотеки
- 5678 страници содржат податоци и простор на магацинот
- 9012 страници се валкани (т.е. изменети страници со податоци што треба да се напишат назад на дискот)

/proc/<pid>/maps

- Датотеката /proc/<pid>/maps е датотека која обезбедува информации за мемориските мапирања на одреден процес

- Кога ќе пристапите до датотеката /proc/1234/maps за процес со PID 1234, ќе видите листа на мемориски мапирања за процесот, каде што секоја линија текст претставува единствен мемориски регион

- Секоја линија е составена од неколку полиња, одделени со празни места

/proc/<pid>/maps

Значење на секое поле:

- Started address: Почетна адреса на меморискиот регион
- Ending address: Завршната адреса на меморискиот регион
- Permissions: Дозволите за меморискиот регион, вклучувајќи ги дозволите за читање (r), запишување (w) и за извршување (x)
- Offset: Поместување на меморискиот регион од почетокот на датотеката или уредот што го поддржува
- Device: Бројот на уредот на датотеката или уредот што го поддржува меморискиот регион
- Inode: Инодниот број на датотеката што го поддржува меморискиот регион
- Pathname: Името на патеката на датотеката што го поддржува меморискиот регион, доколку е достапно

/proc/<pid>/maps

Пример за тоа како може да изгледа датотеката /proc/1234/maps

```
08048000-08056000 r-xp 00000000 08:01 23413      /bin/cat
08056000-08057000 rw-p 0000d000 08:01 23413      /bin/cat
08a89000-08aaa000 rw-p 00000000 00:00 0          [heap]
7f8000000000-7f8000021000 rw-p 00000000 00:00 0
7f8000021000-7f8004000000 ---p 00000000 00:00 0
7f8007bfe000-7f8007c13000 r-xp 00000000 08:01 262148     /lib/x86_64-linux-gnu/libc-2.27.so
7f8007c13000-7f8007e13000 ---p 00135000 08:01 262148     /lib/x86_64-linux-gnu/libc-2.27.so
7f8007e13000-7f8007e17000 r--p 00135000 08:01 262148     /lib/x86_64-linux-gnu/libc-2.27.so
7f8007e17000-7f8007e19000 rw-p 00139000 08:01 262148     /lib/x86_64-linux-gnu/libc-2.27.so
7f8007e19000-7f8007e1d000 rw-p 00000000 00:00 0
7f8007e1d000-7f8007e2c000 r-xp 00000000 08:01 262133     /lib/x86_64-linux-gnu/libgcc_s.so.1
7f8007e2c000-7f800802b000 ---p 0000f000 08:01 262133     /lib/x86_64-linux-gnu/libgcc_s.so.1
7f800802b000-7f800802c000 r--p 0000e000 08:01 262133     /lib/x86_64-linux-gnu/libgcc_s.so.1
7f800802c000-7f800802d000 rw-p 0000f000 08:01 262133     /lib/x86_
```

Останати корисни команди во UNIX

- Печатење на името на хостот (компјутерот): hostname

  $ hostname

  os.labstudenti.finki.ukim.mk

- Информации за хостот (компјутерот на кој сме приклучени) и корисничко име: who am I

  $who am I

  asistent pts/0

  2024-02-25 19:20 (...finki.ukim.mk)

- Печатење на тековно време и датум: date

  $ date

  Mon Mar 28 13:05:59 CEST 2016

Останати корисни команди во UNIX

- Печатење на календар: cal (за тековен месец)

- $ cal godina печати календар за бараната година
- $ cal mesec godina печати календар за баран месец

March 2016

S	M	Tu	W	Th	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		