**LPI – ПОДГОТОВКА**

**SHELL and Getting Help**

1. Type [command] – показва дали командата е вътрешна от шела или външна, като посочва пътя до командата.
2. Help, man, info – три параметъра с които може да се извади помощ за някоя команда.
3. Touch ‘new file’ – използването на единични кавички ще създаде new file с спейс между двете думи.
4. Man –k [command] – изпълнява същото като командата apropos.
5. Man –f [command] – изпълнява същото като командада whatis.
6. Man –a crontab – с тозипараметър ще отвори една след друга 2-те страници за crontab.
7. **http://www.tldp.org/** The “Linux Documentation Project”, which is in charge of man pages and HOWTOs (among other things).
8. **http://www.linux.org/** A general “portal” for Linux enthusiasts.
9. **http://lwn.net/** *Linux Weekly News*—probably the best web site for Linux news of all sorts. Besides a daily overview of the newest developments, products, security holes, Linux advocacy in the press, etc., on Thursdays there is an extensive on-line magazine with well-researched background reports about the preceding week’s events. The daily news are freely available, while the weekly issues must be paid for (various pricing levels starting at US-$ 5 per month). One week after their first appearance, the weekly issues are made available for free as well.

**Работа с редактора VI**

* **Modes** As mentioned earlier, one of the characteristics of vi is its unusual man- ner of operation. vi supports three different working “modes”:
* **Command mode** All keyboard input consists of commands that do not appear on screen and mostly do not need to be finalized using the return key. Af- ter invoking vi, you end up in this mode. Be careful: Any key press could invoke a command.
* **Insert mode** All keyboard input is considered text and displayed on the screen. vi behaves like a “modern” editor, albeit with restricted navigation and cor- rection facilities.
* **Command-line mode** This is used to enter long commands. These usually start with a colon (“:”) and are finished using the return key.

**Основни команди:**

* **а- добавя нов текст след курсора**
* **А – добавя нов текст в края на реда**
* **i – въвежда текст на мястото на курсора**
* **I – въвежда текст в началото на реда**
* **o – отваря нов ред под курсора**
* **O – отваря нов ред на курсора**
* **0 – отива в началото на реда**
* **$ - отива в края на реда**
* **w – отива на следващата дума**
* **b – отива на предишната дума**
* **ctrl+F – следваща страница**
* **ctrl+B – предишна страица**
* **G – отива на последния ред**
* **R – замества символи**
* **dd – изтрива целия ред**
* **x – изтрива символ**
* **u - undo**
* **y – копира реда**
* **p – пейст**
* **“c y 4 w - копирай следващите 4 думи**
* **g p – постави копираното след курсора**
* **:5,11 s /red/blue/ - ще замени от 5 до 11 ред думата ред с думата блу**
* **:10, $s /red/blue – ще направи същото но до края на файла**
* **! – да изпълни някоя команда в шела**
* **r! – като изпълни командата да запише резултата в файла**

**файлове и директории**

ls \* - показва едно ниво поддиректории в директориите.

Опции на командата CP:

* -r – копира директории с цялото съдържание(рекурсивно)
* -p – запазва атрибутите на копирания файл
* -f – презаписва съществуващ файл без да пита
* -v – показва какво прави(вербъс)
* -I – пита за нещата които ще извърши
* -b – прави бекуп копие

cp /etc/passwd . - копира passwd в текущата директория в която се намираме.

Cp file.txt file1.txt – копира файла file.txt още ин път само, че с име file1.txt, така имаме един и същи файл с две имена на едно и също място.

За командите MV и RM действат същите опции както за командата CP. Както при CP за директории използваме параметъра R, така и за rm използваме R като искаме да изтрием директория с цялото и съдържание.

Параметри при командата FIND:

-iname – ще търси по име но без да прави разлика между малки и големи букви.

-size, -type, -name, -perm, -mtime, -user, -group, -exec,

При FIND може да се използват още:

* ! – Отрицание
* а – AND (И)
* о – OR (или)

LOCATE – команда за откриване на системни файлове най-вече, посочва местоположението на файла. Подбна е на FIND но работи по-бързо.

**Регулярни изрази**

Използват се най-вече при командата GREP, допълнителни параметри са:

* -l – списък на фаиловете в които има търсената дума
* -r – рекурсия, посочваме само папка и той търси и в файловете на подпапките
* -I – търси малки и големи букви
* -c – казва на колко реда е намерил търсеното
* -n – показва номера на реда на който е намерил търсеното

Стандартен вход, изход и грешка

Cat file1 file2 file3 | wc –w - ще преброи думите в трите фаила и с пайп ще ги изпрати към WC.

STDOUT > записва изхода

STDOUT >> добавя изхода без да трие другата информация

STDERR 2>&1 file – пренасочва изхода и грешката едновременно в файла.

Tail –n 5 file1 – изкарва последните 5 реда от посочения файл.

Head –n 5 file1 – изкарва първите 5 реда от посочения файл.

Cut –c 5 1-5 file1 – ще изкара 5 колана от фаила които е посочен.

**Shell Commands:**

* P=Hello - така се създава променлива и за да я използваме слагаме $ отпред.
* Export P - ще стане променлива в средата иначе е променлива само в шела.

Echo $p , echo ${p}emil - изкарва промеливата $p, а във вториа пример добавя емил към нея.

* Export TZ=Asia/Tokyo

Date - ще покаже колко е часа в токио.

PATH=$PATH:~/emo - това ще добави папка Емо към системия път. Така се добавят допълнителни неща към променливите.

Which command - връща пътя където се намира командата.

Whereis command – връща пътя и ман страниците на командата. По-подробна е.

**File System**

* Mknod - създава device files ( block device, character devise)
* Mkfifo – създава пайп (pipe)
* File file – дава ни информация какъв е файла.
* Cpio – команда за архивиране, може да използва входа от друга команда и да пренасочи изхода към файл.
* Ls –l | cpio –ov > directory.cpio - o параметъра за архивиране.
* Cpio –idv < directory.cpio - ще разархивира файла. Параметър –d

**Archives and Compression**

При архивите може да се съберат няколко файла в един а при компресията, целта е да се намали размера на файла.

TAR – type archive, команда за архивиране.

SPLIT – команда за рязане на големи фаилове, които да могат лесно да се запишат на дискета или да се изпратят по и-мейл. След това могат да се съединят с командата CAT.

GZIP и BZIP2 – използват се за компресиране.

Синтаксис на TAR:

* tar ⟨*options*⟩ ⟨*file*⟩||⟨*directory*⟩ ...
* **-c** (“create”) creates a new archive
* **-f** ⟨*file*⟩ creates the new archive on (or reads an existing archive from) ⟨*file*⟩, where ⟨*file*⟩ can be a plain file or a device file (among others)
* **-M** handles multi-volume archives
* **-r** appends files to the archive (not for magnetic tape)
* **-t**displays the “table of contents” of the archive
* **-u**replaces files which are newer than their version inside the archive. If a file is not archived at all, it is inserted (not for magnetic tape)
* **-v**Verbose mode—displays what tar is doing at the moment
* **-x** extracts files and directories from an archive
* **-z** compresses or decompresses the archive using gzip
* **-Z** compresses or decompresses the archive using compress (not normally available on Linux)
* **-j** compresses or decompresses the archive using bzip2
* **tar -cvf ~/data.tar data\* - ще направи нов архив с име data.tar на всички дата фаилове**
* **cd /**
* # **tar -cvf /tmp/home.tar /home – влиза в главния каталог и архивира целия HOME в /tmp/home.tar**
* **tar -xvf data.tar - разархивира data.tar**

Важните параметри на GZIP:

* **-c** writes the compressed file to standard output, instead of replacing the original;
* the original remains unmodified
* **-d** uncompresses the file (alternatively: gunzip works like gzip -d)
* **-l** (“list”) displays important information about the compressed file, such as the file name, original and packed size
* **-r** (“recursive”) compresses files in subdirectories **-S** ⟨*suffix*⟩ uses the specified suffix in place of .gz **-v** outputs the name and compression factor of every file
* **-1 ... -9** specifies a compression factor: -1 (or --fast) works most quickly but does not compress as thoroughly, while -9 (or --best) results in the best compres- sion at a slower speed; the default setting is -6.
* The following command compresses the letter.tex file, stores the compressed file as letter.tex.gz and deletes the original:
* Gzip letter.tex - ще компресира файла с име letter.tex.gz и ще изтрие другия файл.
* Gzip –d letter.tex ИЛИ gunzip letter.tex - ще декомпресира файла letter.tex.

Bzip2 – работи както gzip, по същия начин. При разархивиране се използва bunzip2.

Zip – програма за архивиране и компресиране. Пример:

Zip test.zip file1 file 2 - създава test.zip от file1 file2

Zip –r test.zip directory - създава test.zip рекурсивно за директория

Unzip – може да се използва за разархивиране на windows zip фаилове. Пример:

Unzip –v test, unzip –d dir test – посочваме в коя директория да разархивира с параметъра –d.

Unzip test ziptest/file1 - ще разархивира файл1 само но ще създаде и папката ziptest.

Cut – команда за изрязване.

Cut –d: -f 1,7 file - ще отреже първа и седма колона от файла.

**Администриране на Линукс**

**Част – 2**

Getent passwd student - ще искара информация за student от файла /etc/passwd

Getent passwd – ще изведе целия за users от всички изтозници не само passwd, може да се изведе и за няколко определени users, като се изброят един след друг.

**Основни параметри на командата useradd :**

**-c** ⟨*comment*⟩ GECOS field entry

**-d** ⟨*home directory*⟩ If this option is missing, /home/⟨*user name*⟩ is assumed

**-e** ⟨*date*⟩ On this date the account will be deactivated automatically (format “YYYY-MM-DD”)

**-g** ⟨*group*⟩ The new user’s primary group (name or GID). This group must exist. **-G** ⟨*group*⟩[**,**⟨*group*⟩]**...** Supplementary groups (names or GIDs). These groups

must also exist.

**-s** ⟨*shell*⟩ The new user’s login shell

**-u** ⟨*UID*⟩ The new user’s numerical UID. This UID must not be already in use, unless the “-o” option is given

**-m** Creates the home directory and copies the basic set of files to it. These files come from /etc/skel, unless a different directory was named using “-k ⟨*directory*⟩”.

Sudo userdel –r user

Sudo userdel - - remove-all-files user - премахва user с всички файлове.

find / -user xxx –delete - намира и изтрива всички файлове на usera.

Groupadd, groupmod, groupdel - командите за групите.

Създаваме нова група с командата groupadd и след това с командата useradd –G group user той ще влезе в новата група.

Gpasswd –a user group – вкарва user в група.

Gpasswd –d user group – изважда user от група.

Usermod –a –G project joe - вкарва joe в групата project.

Last – команда която изкарва списък на потребителите влизали в системата.

**Access Control**

Chmod –R 755 dir - ще смените правата на всички фаилове и поддиректории в тази директория с права 755.

Chmod - - reference=otherfile thisfile - файла ще получи същите права като другия файл на който е равно reference.

Chown username : groupname file - така се сменява собственика и групата на файла.

Chown : groupname file - така ще се смени само групата.

Chgrp groupname file - ще вкара файла в групата ако се изпълни от обикновен потребител който е собственик на файла и е член на групата в която иска да го вмъкне. Горните команди се изпълняват само от ROOT.

Chgrp и chown също могат да използват параметъра –R за да работят рекурсивно за цяла директория.

Umask –S - показва ни маската която действа в момента.

Ps –u - показва кой юзър какви процеси е стартирал.

Пример как да създадем група и да вкараме юзъри в нея и да създадем тяхна директория.

Groupadd project | създава група

Usermod –a –G project joe | въвежда потребител

Usermod –a –G project sue |

Mkdir /home/project |

Cd /home/project |

Chgrp project /home/project |

Chmod 2775 /home/project | така потребител извън групата ако създаде файл в тази папка той остава за групата.

------------------------------------------------------------

Параметъра -R действа по същия начин и при командата chattr.

Chgrp – записва групата към файл или директория.

**Proccess Managment**

Ps –Al | grep ssh

Pgrep –l ssh - ще покаже същото като горната команда.

Nice – команда за задаване на преоритет на стартираните процеси. Варира от -20 до +19. По подразбиране стойноста е +10. Само руут може да задава преоритет с минус. Обикновенните потребители могат да използват само командата RENICE за смяна на преоритета. Синтаксиса е:

Nice [nice value] command parameter

Renice [nice value] [PID]

Top - команда като PS, която дава много подробна информация. Използва се за управление и мониторинг на процесите.

**Hardware**

Lsmod – дава информация за заредените в момента модули.

Modprobe module – команда за зареждане на модули ( /lib/modules/ ).

Modprobe –r module - премахва модула.

/etc/modprobe.conf - конфигурационен файл.

През директорията /sys/devices/ можем да достъпваме до периферния хардуер.

HAL - hardware abstraction layer. Използва се разпознаване на оериферен хардуер, например като свържем камера и да ни излезе иконка на десктопа.Идеята е, че чрез udev ядрото вижда, че има нов хардуер и кой е той. Самия хардуер може да бъде попитан за някакви базови настрийки. Няма как ядрото да има информация за целия възможен хардуер, за това се инсталира драйвъра. HAL прави връзката между драйвърите и хардуера.

D-BUS -

Udisks

**Hard Disks**

IDE - intergrated drive electronics

ATA - AT Attachment

SATA - Serial ATA

SCSI - Small Computer System Interface ( скъзи )

Съществуват няколко популярни стандарта за дискови контролери:

* ***АТ***
* ***IDE ( Integrated Drive Electronics)***
* ***EIDE (Enhanced Integrated Drive Electronics)***
* ***ESD (Enhanced Small Device Interface)***
* ***SCSI (Small Computer Systems Interface)***
* ***SATA (Serial ATA)***
* **SSD (Solid-State Drive)**

Тези контролери се различават по метода на запис, скорост на обмен на данни. Интерфейсът [SCSI](https://en.wikipedia.org/wiki/SCSI) е по-скоро шина, тъй като към порт SCSI могат да се свържат верижно от 7 до 15 периферни устройства чрез една и съща интерфейсна платка.

***- Integrated Drive Electronics (IDE, интегрирана електроника на устройство)*** – IDЕ обикновено е по-евтина от другите възможности, а за много потребители тя би донесла същата производителност. Само че тя има по-малко възможности и ще работи по-бавно на компютър с голямо натоварване на дисковете. Процесорът върши повече работа при IDE, отколкото при SCSI, намалявайки сложността на адаптера за IDE и по този начин и цената. Поддържат се две устройства на кабел, като едното от тях е главно (Master), а другото е подчинено (Slave). Дисковият интерфейс IDE е прост, доколкото това е възможно. Той представлява дисков контролер (на устройството) с минимален набор от интерфейсна електроника за връзка с компютърната шина. Вариантът на интерфейса зависи от шината и контролера, което означава, че е необходим специализиран интерфейс за връзка между IDE и различните типове шини. Другото име на IDE е АТ Attachment или АТА. От 1999 година се появи нов стандарт на режим на работа, наречен ULTRA АТА (DМA), позволяващ много високи скорости на предаване.

***- Small Computer System Interface (SCSI, интерфейс за малки компютърни системи)*** – SCSI е по-скъпа и устройствата за нея са също по-скъпи. Най-мощните устройства за SCSI са по-бързи и по-големи от IDE. При по-големи изисквания е необходима SCSI. За разлика от IDE, SСSI е входно – изходна шина с общо предназначение, която може да свързва голямо разнообразие от устройства по високопроизводителен начин. Устройствата, поддържани от SCSI включват:

1. **Дискове** – въпреки, че SCSI не е ориентиран специално към дискове, повечето устройства, включени към шина SCSI са всъщност твърди дискове.
2. ***CD-ROM*** – шината SCSI е пакетна, което означава, че SCSI контролерът изпраща пълни командни последователности на устройствата. Също така SCSI има общо множество от команди, включително такива като за четене и запис, които работят с повечето от свързаните устройства.
3. **Лентови устройства** – скенери, модеми, дискови масиви, принтери, други компютри – SCSI поддържа всички тези неща без промяна на спецификацията заради някое от тях.

\* ***SATA (Serial ATA****)*[[4]](https://bg.wikipedia.org/wiki/%D0%A2%D0%B2%D1%8A%D1%80%D0%B4_%D0%B4%D0%B8%D1%81%D0%BA#cite_note-:0-4)

* [](https://bg.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB:Serial_ata.jpg)

Кабели за SATA стандарт

Това е сравнително нов интерфейс за дискове. Той е свързан към твърдия диск чрез тънък четирижилен кабел, вместо с 40- или 80- жилен – така се борави по-лесно, а и циркулацията на въздуха в кутията ня компютъра е по-добра. Трансферната скорост е от 150 МВ/сек и може да достигне до 600 МВ/сек. Предимствата са, че твърдите дискове по-лесно се инсталират, а новите бързи чипове от контролерната система са по-евтини за произвеждане и консумират по-малко енергия. При Serial ATA се премахва и едно от ограниченията на EIDE – границата от 128 GB за капацитета на твърдия диск.

Причини за повреди и мерки за сигурност[[редактиране](https://bg.wikipedia.org/w/index.php?title=%D0%A2%D0%B2%D1%8A%D1%80%D0%B4_%D0%B4%D0%B8%D1%81%D0%BA&veaction=edit&section=17) | [редактиране на кода](https://bg.wikipedia.org/w/index.php?title=%D0%A2%D0%B2%D1%8A%D1%80%D0%B4_%D0%B4%D0%B8%D1%81%D0%BA&action=edit&section=17)]

Най-често срещаните причини за повреди в твърдия диск са:

* При новите високооборотни твърди дискове има основно термични проблеми, свързани с нагряването на въздуха между единичните дискове.
* Задиране на магнитната глава и диска. При нормална работа главата лети върху диска на определена височина. При определени условия – вибрации, удари, замърсяване и други, може да се получи задиране.
* Външните магнитни полета могат да изтрият и разрушат служебната сервоинформация и направят диска негоден.
* Грешка в електрониката и износване на механиката
* Дълъг престой в неработещо положение, когато главата лежи неподвижно на диска, може да доведе до залепването ѝ върху него и невъзможност изобщо диска да се завърти.
* Стресови вибрации и удари могат да доведат до бързо повреждане.
* Висока температура на работа, над работната температура посочена от производителя.
* Твърдите дискове по принцип могат да работят във всяко положение, но трябва все пак да се има предвид и указанието на производителя. Дисковете е добре да са закрепени към конструкцията на компютъра, за да се намали влиянието на собствените вибрации.
* По време на монтаж да се спазват правила за ESD.
* Да се прави редовно копиране на важните документи от твърдия диск.

Средният брой работни часове, преди един диск да се повреди се означава като MTTF (Mean Time To Failure), а при твърдите дискове, които се ремонтират MTBF – (Mean Time Between Failures). Това са данни, които се събират статистически за различните модели на различните производители.

Когато записваме файл на харда, той заема блокове от харда и ако един блок е 4кб а ние запишем файл който е 500байта той ще заеме една осма от един блок но цялото останало място от блока ще е на вятъра или просто загубено. За това ако имаме множество малки фаилове за съхранение можем да създадем дял на който блоковете да са с по малка големина, например 1 или 2 кб.

P-ATA – тук името на харда в линукс ще бъде HAD, HDB и т.н.

В другите варианти са SDA, SDB и т.н. като S идва от SCSI.

/dev/block – там udev записва някакви символични имена на устройствата които се кънектват към машината.

При сървърите е добре да са на отделни дялове:

/home

/tmp

/var

/srv

Добре е да се зададе SWAP дял с големина колкото е RAM памета. По принцип има правило в UNIX систеите което е гласяло, че SWAP дяла трябва да 2-3 колкото RAM памета но в линукс не важи.

Ако изтрием по по грешка някой дял от хард диска с PARTED може да успеем да възтановим информацията. Има същото действие като fdisk но е малко по-мощна от fdisk, в графичния интерфейс се казва GPARTED нещо като PARTITION MAGIC за WINDOWS.

Loop Devices - това е файл, който може да се форматира с файлова система и да се третира като отделен физически диск.

Пример за Loop device:

# dd if=/dev/zero of=~/file.img bs=1MiB count=10

# losetup --find --show ~/file.img

/dev/loop0

#losetup –a (L)

# mkfs -t ext2 /dev/loop0

# mount /dev/loop0 /mnt mount –o loop file.img /mnt/disk

...

# umount /dev/loop0

# losetup --detach /dev/loop0

**File Systems**

Mkfs –t btrfs –L MYBTR –d raid1 /dev/sdb1 /dev/sdc1 - ще създаде файлова система BTRFS с лейбъл MYBTR в раид-1 от посочените два диска и ще пише данни и на двата.

Btrfs subvolume create /home - btrfs подържа създаване на subvolume в които може да се правят SNAPSHOT. Subvolume са различен тип дялове на файловата система.

Btrfs subvolume snapshot /mnt/sub /mnt/sub-snap - създаване на snapshot.

Subvolume - A subvolume is a part of filesystem with it’s own and independent file/directory hierarchy. Subvolumes can share file extents. A snapshot is also subvolume, but with a given initial content of the original subvolume.

SNAPSHOT - A snapshot volume is a special type of volume that presents all the data that was in the volume at the time the snapshot was created. For a more detailed description, see [Section 3.8](http://www.tldp.org/HOWTO/LVM-HOWTO/snapshotintro.html), Snapshots. This means we can back up that volume without having to worry about data being changed while the backup is going on, and we don't have to take the database volume offline while the backup is taking place.

SWAP PARTITION: Създаване на SWAP след като е отделен дял за него:

Mkswap /dev/sda4 - формтира дяла

Swapon /dev/sda4 - активира дяла

Cat /proc/swaps - трябва да се появи в /proc/swaps

Swapoff /dev/sda4 - деактивира дяла.

За да се активира с зареждането на системата се монтира в FSTAB. Допустими са 32 дяла.

Dd if=/dev/zero of=swapfile bs=1m count=256 - създаваме файл, който да се използва за SWAP и продължаваме с горните стъпки.

Задаване на лейбъл с помоща на две команди ако сме забравили да го зададем като сме създавали файловата система: Примери:

e2label /dev/sda3 Home

tune2fs –L home /dev/sda3

mount –t ext2 LABEL=home /dev/sda3

lsblk - извежда информация за дисковете.

Квотите са средство, с чиято помощ указвате колко дисково пространство може да заема даден потребител или група на Вашата Linux система. С квотите определяте лимитита за дисково пространство по два начина:  броят inode-и (файлове, устройства, директории т.н), които даден потребител или група може да притежава, както и броят блокове от диска(в повечето случаи блокове от 1К).

Квотите задължават потребителите или групите от потребители да спазват този лимит, като по този начин предпазват от заемане   
на цялото дисково пространство от един потребител и нарушаване нормалната работа на системата.

Квотите се определят за различни файлови системи, потребители или групи от потребители.

В момента се поддръжа единствено ext2 файлова система.

Мекото ограничение посочва колко капацитет от дисковото пространство може да използва даден потребител на квота.Когато той достигне ограничение се получава предупреждение. Твърдото ограничение показва какъв е капацитета за ползване от потребителя,извън което не може се простира.Когато достигне ограничението си ,не може да използва допълнителното пространство.

Преди да се поставят квоти трябва да се инсталира пакета за съответната дистрибуция. След инсталацията трябва да посочим в коя файлова система ще се използва квота.

Mount –o remount,usrquota /home

Също така е добре да се посочи и в /etc/fstab на 4-та колона след типа файлова система – defaults,usrquota

quotacheck –avu - създава aquota.user DATABASE

quotaon –avu - пуска квотата за да може базата данни направена от quotacheck да се упдейтне.

Edquota –u user - задаваме квота са определен потребител.

Edquota –p tux hugo - създаваме прототип на потребител, така че при нов потребител да не въвеждаме отново.

Quota –q - обикновен потребител може да си провери квотите.

Repquota –a - проверява си използваното дисково пространство и какви са квотите.

Маунт опциите за група са: grpquota a създадената база данни е aquota.group

Quotaon –auvg – за стартиране на груповата квота.

**BOOTING LINUX**

След включването на компютъра нещата се поемат от BIOS или EUFI (unified extensible firmware interface). Bios проверява за операционна система върху първото устройство което му е заложено да зареди. Той чете първите 512байта, така наречения BOOT SECTOR или Master Boot Record (MBR). Първите 446 байта съдържат малка стартуп програма за стартиране на Операционната система, нарича се BOOT LOADER. Останалата част са за partition tables.

Модерния boot loader за линукс се нарича GRUB (Grand Unified Boot Loader). Той не е само boot loader но е и boot manager. Може да зарежда различни ядра на линукс, както и различни операционни системи.

Буутваща процедура през UEFI:

UEFI не използва буутващ сектор, той съдържа в себе си буутващ сектор с информация за различните операционни системи. Boot loaders за различните операционни системи се съхраняват като обикновенни фаилове върху efi system partition (ESP) от където фирмуера чете и ги зарежда.

GRUB може да прочете ядрото директно от директорията /boot/vmlinuz

Основния конфигурационне файл на GRUB e : /boot/grub/menu.list

Конфигурационния файл grub.cfg се създава с командата grub-mkconfig. Не е препоръчително да се редактира директно файл GRUB.CFG.

Update-grub извиква командата grub-mkconfig и пренасочва стандартния и изход в /boot/grub/grub.cfg.

Може да се правят промени и в /etc/grub.d/40\_custom и grup-mkconfig ще вземе новите настройки и ще ги запише в /boot/grub/grub.cfg

**System V , Upstart , Systemd**

Telinit q - кара init да си препрочете конфигуразионния файл.

Mount –o remount,ro / - монтира главния каталог в риид онли. Подходящо е когато администратора работи в сингъл юзър мод.

Upstart модела използва командата initctl

**Dynamic Libraries**

Linux-gate.so.1 – за 32 битови машини

Linux-vdso.so.1 – за 64 битови машини

Динаичните библиотеки завършват на .SO

Статичните библиотеки завършват на .A

Busybox – шел който съдържа различни юникс туулс.

Ldconfig – регенерира кеша на активните библиотеки.

Ldd - изкарва динамичните библиотеки използвани от програмите.

Strip – премахва ненужната информация от изпълнимите фаилове.

**Software Package Management**

Dpkg - - install (-i) file.deb - инсталира пакета file.deb

Dpkg –I - - unpack - - configure (-a) file.deb - така ще започне инсталиране, разопаковане на пакета и неговото конфигуриране.

Ако има инсталиране по-стара версия на програмата, тя ще бъде изтрита и ще се инсталира новата. Ако възникне някаква грешка има много начини да се възтанови старата версия.

Конфигурационния файл е в : /etc/dpkg/dpkg.cfg

Няколко причини за неуспешно инсталиране на пакета:

1. Пакета изисква един или няколко допълнителни пакета (зависимости) които не са включени в инсталацията. Тази проверка може да се прескочи с параметъра - - force-depends но могат да настъпят доста обърквания в системата.
2. Инсталирана е по-стара версия на програмата и е настроена да бъде на HOLD. Това я предпазва от инсталиране на нова версия.
3. Пакета се опитва да ънпакне файл, който вече го има в системата но принадлежи на друг пакет. Тук трябва да се зададе параметъра за REPLACING - - force-overwrite

Dpkg - - remove (-r) program - премахва програмата без конфигурационния файл

Dpkg - - purge (-P) program - премахва цялата програма с всички фаилове

Dpkg-source –x program.dsc - ще изведем сорс кода на пакета

Dpkg - - status hello - показва статуса на инсталирания пакет.

Има няколко вида пакетни зависимости:

Pre-depends – пакети които трябва да са напълно инсталирани върху системата.

Depends – пакети които трябва да се инсталират и конфигурират за пакета които ще се инсталира.

Recommends – не е абсолютно задължителен пакета но е добре да се инсталира.

Suggests – препоръчителни

Enhances – като suggests но са в резерва.

Conflicts – тези пакети не могат да се инсталират с пакета който искаме да инсталираме.

Apt-cache search

Apt-cache show

Apt-cache depends - ако има вертикална черта пред някой от посочените depends той трябва задължително да е инсталиран.

Apt-cache stats

Dpkg-reconfigure my-packege

Когато преконфигурираме даден пакет можем да зададем преоритета на въпросите които да преконфигурираме. Преоритетите са:

Critical – задължителни въпроси които са свързани с грешки възникнали в системата.

High

Medium

Low

Dpkg-reconfigure - - priority=medium my-package - зададено по този начин ще ни зададе въпроси от ниво медиум на-горе и ще пропусне ниво лол.

Преобразуване на RPM пакети в DEB пакети за да може да се инсталират на DEBIAN.

Alian - - to-deb packet.rpm

Alian - - to-rpm packet.deb

**RPM & YUM**

Инсталиране на пакети с RPM:

Rpm –I /tmp/ssh.12.rpm

Rpm –I /tmp/\*.rpm

Деинсталиране на пакети:

Rpm –e /tmp/ssh.12

Rpm –q openssh - показва как се казва пакета

Rpm –qa - показва всички инсталиране пакети

Rpm –qa | grep cups - ще покаже всички пакети на cups

Yum install hello

Yum remove hello – ще премахне и пакетите които използва стига да не се ползват от друга програма. Remove е еквивалент на erase.

Yum update hello – ще update-не пакета.

Yum groupinfo ‘printing support’ - ще покаже списък на пакетите необходими за принтиране.

Yum groupinstall – ще инсталира цялата група.

Yum info hello – информация за пакета.

Yum search mysql

Yum deplist mysgl.132 – ще покаже пакетните зависимости.

Yum downloader - - destdir /tmp/hello – само ще свали пакета.

**Разглеждане на видеата от курса и примерните въпроси**

Началото на скриптовете започва с #! /bin/bash - това показва кой команден интерпретатор да се стартира и скрипта се насочва към него.

Echo “Hello $USER” - имаме вкарана променлива на обкръжението, тогава ехо я прочита и ще изкара отговор Hello Student.

Единичните кавичи няма да интерпретират написаното, а когато няма кавички ще интерпретира всичко което е написано до края на реда.

Echo $? – връща статуса на предната команда. Ако е 1 всичко е на ред, 0 има грешка.

Env – извежда променливите на обкръжението. Може да стартира команди в модифицирана среда.

Export - експортира променливите в обвивката с която в момента работим.

Не слагаме знак за долар когато експортираме променливата и го слагаме когато експортираме стойността й. Може да се дописва във всеки файл в който сме задали променлива отдолу дописваме export promenliwata.

Source – експортира нещата само в текущия шел, в другите шелове няма да има нищо от експортираното.

History –d 11 - ще изтрие 11 команда.

History –c - ще изтрие цялото хистори.