БИЛЕТ 8– Операционни системи

I. Посочва, различава и демонстрира знания за отделните хардуерни компоненти на компютърна система. – 12т.

II. Компютърна система (КС) – характеризира се със слоеста структура, като всеки слой изпълнява конкретни задачи и освобождава по-горните слоеве (фиг. 1).

Фиг. 1 Структура на КС

1. Хардуер (Hardware) - апаратната част на компютъра, която предоставя машинни команди за управление на КС. Хардуер са процесор, дънна платка, твърд диск, видео карта, RAM памет, захранващ блок компютърна кутия и входно-изходни устройства.

1.1. Централен процесор (Central Processing Unit, CPU) - програмируемо устройство, което, използва инструкции от паметта на КС, дешифрира ги, изпълнява стъпка по стъпка необходимите действия (аритметични или логически операции, прехвърляне на данни) и записва резулатаите. Основни характеристики на процесорите са - тактова честота, производителност и архитектурата. Процесорите се класифицират по няколко основни признака – предназначение, използвани команди, кеш памет и брой ядра.

Процесори:

Intel (Pentium, Celeron и Core i.) – мощни, високо производителни процесори, за които е характерна висока скорост на обработка, голям кеш, консумация на малко енергия. Честотата варира от 3000 MHz до 4500 MHz. Недостатъци на тези процесори са високата им цена и фактът, че при надстройка на процесора е необходима подмяна на дънната платка поради несъвместимост с по-стари модели. Голяма част от процесорите са Intel-съвместими, т.к имат набор от команди и интерфейси за програмиране, сходни с използваните в Intel.

AMD – по-голем брой ядра, добра тактова честота, по-малък кеш. Като предимство на тези процесори може да се отбележи фактът, че нови процесори могат да се инсталират на по-стари платформи, както и ниската им цена. Недостатъци - шумни, консумират много енергия.

1.2 Дънна платка – основата върху която се изгражда компютъра, от нея се определя избора на вида процесор, типа и количеството оперативна памет, броя и видовете допълнителни контролери (видео и звукови карти, модеми и т.н.).

1.3 Твърд диск (hard disk) - запаметяващо устройство от енергонезависим тип, използвано за четене, запис и съхранение на голем обем от информация. Информацията се съхранява чрез магнитен запис върху една или няколко магнитни плочи, разположени една под друга и глава, която чете и записва инфомацията. Плочите се въртят със висока скорост, главната е на микрони разстояние от останалите плочи, с цел да недопускане на триене между плочите. Хардискът е затворен херметически в кутия, за да се предпази от замърсяване. Основните характеристики на твърдите дискове са: капацитет, латентност (закъснение), скорост на въртене на плочите и интерфейс. Днес работим с хард дисковете с капацитет до няколко 1TB и с латентност около 17ms.

1.4 Видео карта – електронна платка, осигуряваща връзката между потребител и компютър, която отговаря за прехвърляне и визуализиране на обработената информация от компютър на монитора.

1.5 RAM (Random Access Memory) - набор от електронни чипове или модули, инсталирани директно на дънната платка или в специални слотове върху нея. RAM паметта е работната памет на процесора, памет за четене и писане. Тя е енергозависима. При стартиране на всяка програма, кодът й се записва в RAM паметта, процесора прочита последователно инструкциите и ги изпълнява. Мерна единица на RAM паметта - байтове (B).

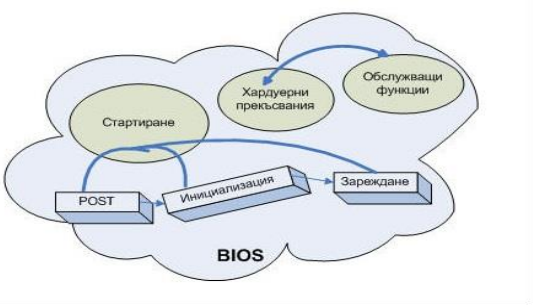
1.6 Периферни устройства - от тях до голяма степен зависи производителността и надеждността на работата на КС. Те се използват за въвеждане/извеждане на информационен поток, обработен от компютъра или насочен за обработка към него. Периферни устройства са:

- ✓ Запаметяващи устройства - CD-ROM, DVD-ROM, хард дисково устройство, устройство с магнитна лента, флашка и др.
- ✓ Входно-изходни устройства - клавиатура, мишка, джойстик, монитор, уеб камера, скенер, принтери, плотери, 3D фрези, факс машини, тонколони и др.

1.7 . **Захранване** – електрически модул, осигуряващ необходимите напрежения и мощности на компоненти, монтирани на дънната платка на компютъра, осигуряващи неговата синхронна работа.

2. Базова входно/изходна система (Basic Input/Output System, BIOS) - съвкупност от програми в ROM паметта, които осигуряват прехвърляне на данни и управление на инструкциите между компютъра и периферните устройства. Основни програми в BIOS (фиг. 2):

- ✓ ***POST*** (Power On Self Test) - тест при включване на захранването;
- ✓ ***BIOS SETUP*** - установяване параметрите на КС и взаимодействие с *CMOS*-паметта.



Фиг. 2 Процедури в BIOS

3. **Ядро** - първият софтуерен слой на Операционната система (ОС). То осигурява:

✓ операции за вход и изход за най-горния слой в ОС чрез драйвери.

✓ реализира системните примитиви и осигури потребителски интерфейс. Обръщанията към ядрото в ОС (system calls) се нарича примитиви, те обуславят интерфейса на ядрото.

✓ разполага процесите в оперативната памет.

✓ управлява входно-изходните устройства

4. **Обвивка** (Shell) - потребителска програма, която въвежда и извежда информация чрез терминал. Обвивката е видимия за потребителя интерфейс на една ОС.

5. **Приложни програми -** набор от програми за решаване на конкретни задачи. Те биват:

✓ **Обслужващи програми –** за редови потребители - текстови редактори, програми за обработка на файлове и др (Word, Excel, Access, Edit и др.).

✓ **Програми на системния администратор** - средства за настройка и проверка на състоянието на ОС (Command Prompt)

✓ **Системи за програмиране** - инструменти за създаване на софтуер (среди за програмиране- Visual Studio), свързващи редактори и др.

II. Обяснява структурата на операционната система. Демонстрира знания за архитектурата на операционните системи – 10т.

2.1 Структура на ОС

ОС е част от системения софтуер, който управлява и координира работата на КС. ОС получава данни от потребителя, обработва ги и връща отговор на заявките във вид на HTML код или друг формат, поддържан от съответния браузър. Съвременните ОС са многопотребителски – ресурсите се разпределят между потребителите, като те се конкурират за достъп до системата. ОС се състои от обвивка, ядро и приложни програми

Ядро на ОС - със слоеста структура, вградено (постоянно) в основната памет, с цел обработката на прекъсванията (събитие при което се променя нормалната последователност на изпълнение на командите от процесора).

✓ Първият (най-долен) слой - управлява процесите, като осигурява механизъм за комуникация.

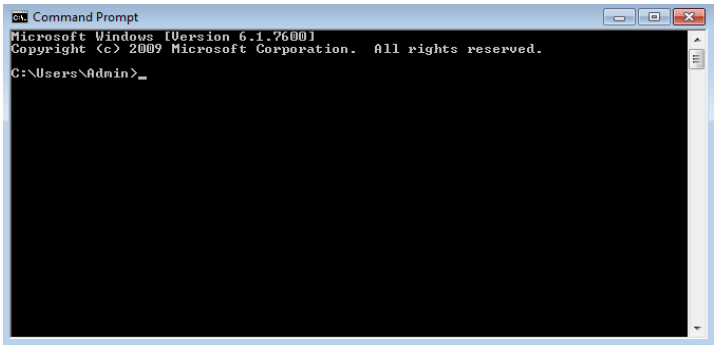
✓ Следващият слой - входно-изходни задачи, като за всяка задача има драйвер, изпълняван като отделен процес.

✓ Чрез третия слой се реализират системните примитиви, като се използват файлове (*file system*), или *memory manager*.

Ядрата биват - еднопроцесни – в даден момент ядрото поддържа работата на един процес и **многопроцесни** – ядрото подържа работата на няколко процеса, като един потребител може да изпълнява няколко процеса в конкретен момент.

Обвивка (Shell) - обезпечава интерфейса на потребителя. Обвивките са два вида – команден ред или графична среда.

Команден интерпретатор - осигурява интерфейс с команден ред, използващ определен команден език (фиг. 3).



Фиг. 3 Команден интерфейс

Най-използваните командни интерпретатори:

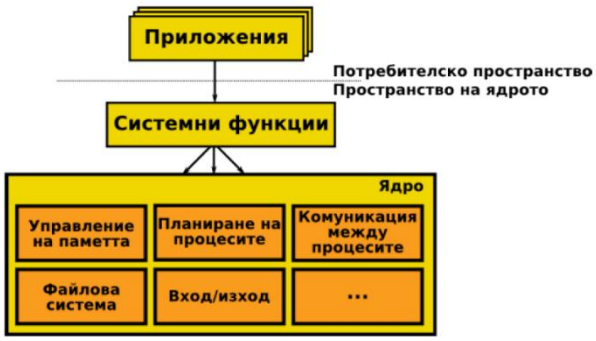
- ✓ *в Windows*: *cmd.exe* (*Command Prompt*), *4DOS, Command.com*;
- ✓ UNIX (и модификациите му): *bash, tcsh, ash, csh, sh, zs и др.*;

Графични среди - графически интерфейс на потребителя (Graphical User Interface – GUI), при който обработваният елемент (файл, програма) се представя на екрана на монитора с помоща на икони.

2.2 Архитектура на ОС

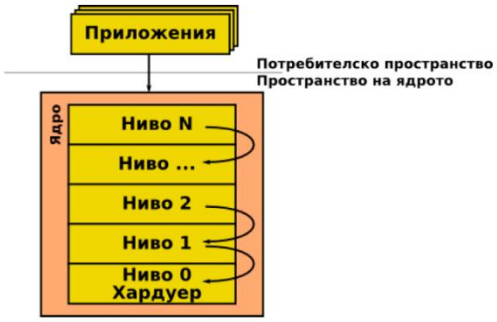
Ядрото е съвкупност от вътрешните компоненти на ОС, които изпълняват основните функции, поддържащи компютъра в работно състояние. Съществуват два подхода за структуриране на ядрото:

✓ **монолитно** – ядрото е един изпълним файл, съдържащ модули, като всеки модул е със строго определен интерфейс (например в *UNIX и LINUX)*. Всеки компонен на ОС се съдържа в ядрото и директно комуникира с останалите компоненти. Характерно е висока производителност и трудно се проследяване на бъгове. Грешки в един компонент води до нежелан ефект в друг компонент (фиг. 4).



Фиг. 4 Монолитно ядро

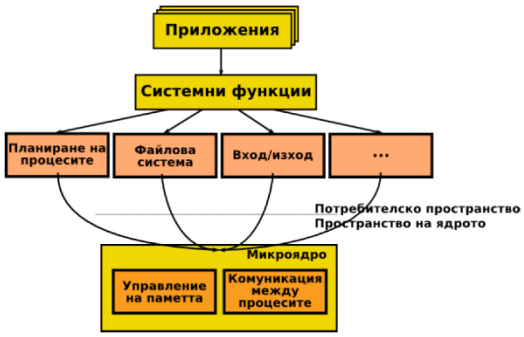
✓ **йерархично** (многослойни) – ядрото се разделя на слоеве, осигуряващи операции, чрез които по-горния слой се обръща към по-долния (например в *MINIX*) (фиг.5).



Фиг. 5 Многослойно ядро

Многослойната ОС е разделена на определен брой от слоеве, като всеки слой се изгражда върху основата на по-долните слоеве. Най-долното ниво (ниво 0) е хардуер, най-горното ниво (ниво N) се състои от потребителския интерфейс и потребителски програми. Всеки слой използва функции и услуги само от по-долните нива, като всеки слой комуникира само със съседните си слоеве. Между слоевете е налична защита. Необходимостта от комуникация между слоевете води до намаляване бързодействието на ядрото.

✓ **Архитектура с микроядра** – предоставя малък брой услуги. Тази архитектура се характеризира с висока степен на модулност. ОС е лесно преносима и мащабируема. Услугите, предоставяни от ядрото се преместват в потребителското пространство. Многоядрената структура е по-надежда тъй като по-малка част от кода работи в незащитен режим (фиг.6).



Фиг.6 Микроядрена структура на ОС

III. Прави заключения и изводи за файловата структура на ОС при конкретна поставена задача. – 8т.

Системата за управление на файлове е основна за съвременните ОС. UNIX-базираните ОС не могат да функционират без файлова система, чрез нея те свързват данните на различни програми за системна обработка. По този начин се решава проблема с централизираното разпространение на дисковото пространство и на данните. Чрез файловите системи се предоставя на потребителите достъп до данните, които са записани на външна памет. Най-често срещаните файлови системи са FAT, FAT32 и NTFS.

Файлът е единица за данни със структура, която се определя от неговото приложение. Файловете се класифицират по:

- *съдържание* - програмни файлове, файлове данни, текстови;
- *форма на представяне* - символни, двоични, смесени;
- *метод на достъп* - последователни, директни, индексни и др.

В UNIX-базираните ОС файлът е последователност от байтове, без никаква интерпретация. Тази схема допуска максимална гъвкавост, но минимална поддръжка.

IV. Посочва, обяснява и демонстрира команди, чрез които се показва функционалността на операционните системи. – 12т.

ОС Linux – осигурява средства за лесно опериране с файловете (команди за създаване, копиране, изтриване, преместване на файлове между директории и т.н).

Извеждане на съдържанието на текущата директория се постига с команда ***ls.*** При логическо включване в системата се влиза в специална директория HOME (за всички потребители).

- ✓ За да създаване на файл и добавяне на съдържание се използва команда *cat*

  $ cat > names
  Ani Mimi Lili Ivan Petyr
  Ctrl+d
  $

- ✓ При задаване на *$ cat > names* се извежда съдържанието та файл *names.*
- ✓ За създаване на празен файл се използва команда *touch*

  $ touch empty_file
  $ ls empty_file

- ✓ Сортиране (във възходящ ред) на съдържанието на файл *sort*

  $ sort names
  Ani
  Ivan
  Lili
  Mimi
  Petyr
  $

✓ Команда *wc* извежда броя на редовете (*-l*), думите (*-w*) и символите (*-c*) във файл:

$ wc names

5, 5, 20, names

$

✓ Изтриване на файл *rm;*

✓ Преименуване на файл *mv*

$ mv saved_names new_names

$

✓ Премахване на дублирани редове *uniq*;

✓ Команда *tr* се използва в комбинация с пренасочване на входа и изхода. Пример - символът < е за пренасочване на входа и изхода

$ tr a z < names - преобразува всички „а”-та в „z” във файла *names*.

Работа с директории

✓ Извеждане на текущата директория - *pwd*

$ pwd

/home/student

$

✓ Създаване на директория или няколко директории (*documents, test, proba, 1*) *mkdir*

$ mkdir documents test proba 1

$

✓ Промяна на директория – *cd*

✓ Команда *chmod* се използва за промяна на файловите права. Само потребителят, който е собственик на файла и root могат да променят правата над файла. Когато се използват числови права:

$ chmod 770 my.file - задава пълни права за *user* (***rwx***) и **group** (***rwx***) и отнема всички права на останалите потребители (---) в ОС.

✓ *chown* - сменя потребителя собственик на файл.

✓ Команди за наблюдение на процесите в ОС: - ***ps, ps lax, ps aux***, ***top***

ps - основен инструмент за наблюдение на процеси от системните администратори. С него се вижда PID и UID идентификаторите, приоритета и контролният терминал на процесите. Командата дава и информация за използваната от процесите памет, процесорно време и текущото им състояние (работещи, спрени, спящи и т.н.). Зомбитата в списъците са означени като <defunct>;

ps lax - предоставя полета като PPID (идентификатора на родителския процес), NI (nice стойност) и WCHAN (ресурсът, за който чака процесът);

ps aux - стандартен списък на всички работещи в ОС процеси.

USER - потребителско име на собственика на процеса

PID - идентификатор на процеса

%CPU - процент, до който този процес е натоварил процесора, *%MEM* - процент заета от процеса реална памет

VSZ - виртуален размер на процеса, *RSS* -брой страници в паметта

TTY - идентификатор на контролния терминал

STAT - текущото състояние на процеса: R - готов за изпълнение D - непрекъсваем сън, S спящ (< 20 сек.), T - следен или спрян, Z – зомби, допълнителни флагове: W - процесът е преместен във виртуалната памет 5 < - с по-висок от нормалното приоритет, N - с по-нисък от нормалното приоритет L – страници, заключени в *core*, S - водач на сесия, START -времето на стартиране на процеса,

TIME -консумирано процесорно време,

COMMAND -име на команда и нейните аргументи.

top - постоянно обновяващ се списък на активните процеси и заеманите от тях ресурси.

За Windows команди:

✓ *systeminfo* - информация за конфигурацията на компютъра и неговата ОС, информация за сигурността, идентификационен номер на продукта и хардуерни свойства като RAM, дисково пространство и мрежова карта, времето за стартиране на системата, версия BIOS, процесор, паметта, информация за файловете на страницата и др.

✓ *tasklist* - списък на текущите процеси на локалния компютър, текущи задачи

tasklist > D:ListOfProcesses.txt

✓ *net user*- за създаване и промяна на потребителски акаунти.

V. Обяснява пакетните системи в ОС. – 4т.

Повечето модерни Unix-базирани ОС предлагат централизиран механизъм за намиране и инсталиране на софтуер, разпространяван под формата на пакети , съхранявани в хранилища. Работата с пакети се нарича управление на пакети. Пакетите предоставят основните компоненти, споделени библиотеки, приложения, услуги и документация.

Система за управление на пакети освен инсталация на софтуер, предоставя инструменти за надграждане на вече инсталирани пакети. Повечето пакетни системи са изградени около колекции от пакетни файлове. Файлът на пакета обикновено е архив, който съдържа компилирани двоични файлове и други ресурси, съставляващи софтуера, заедно с инсталационни скриптове. Пакетите съдържат метаданни, включително техните зависимости, списък на други пакети, необходими за тяхното инсталиране и стартиране. Въпреки че тяхната функционалност и предимства са сходни, форматите и инструментите за опаковане варират в зависимост от платформата:

ОС	**Формат**	**Инструменти**
Debian	.deb	***apt*, *apt-cache*, *apt-get*,*dpkg***
CentOS	.rpm	***yum***

Пример: CentOS, Fedora и други членове на семейството Red Hat използват RPM файлове. В CentOS ***yum*** се използва за взаимодействие както с отделни файлове с пакети, така и с хранилища.

Актуализиране на списъци с пакети

Повечето системи поддържат локална база данни за пакетите, достъпни от отдалечени хранилища.

ОС	Команда
Debian / Ubuntu	***sudo apt-get update, sudo apt update***
CentOS	***yum check-update***

Надстройка на инсталираните пакети

Във FreeBSD надстройването на инсталираните портове може да доведе до пробивни промени или да изисква стъпки за ръчна конфигурация.

ОС	Команди	
Debian / Ubuntu	sudo apt upgrade	Надстройва само инсталираните пакети
CentOS	sudo yum update	

VI. Диференцира процесите в операционните системи. Демонстрира знания за виртуалната памет на конкретна операционна система. – 14т.

Съвременните ОС са мултипрограмни, при тях се създават множество процеси, всеки от които се нуждае от процесорно време, памет, набор от файлове и входно/изходни устройства при изпълнение на своите задачи. Процесът описва работата на програма. Той е единица работа в ОС, която е съвкупност от команди и ресурси в текущия момент на неговото изпълнение. Процесът описва състоянията, през които преминава дадена програма от стартирането до нейния край. Програмите са пасивен обект, докато процесите – активен, т.к процесът е дейност по изпълнение на програмата. За да се опише понятието процес са въведени термините:

- ✓ задача (task) - синоним на процес,
- ✓ задание (job) в по-старите операционни системи се използва за група от задачи,
- ✓ нишката (thread) - част от процес, т.е. един процес се състои от няколко нишки

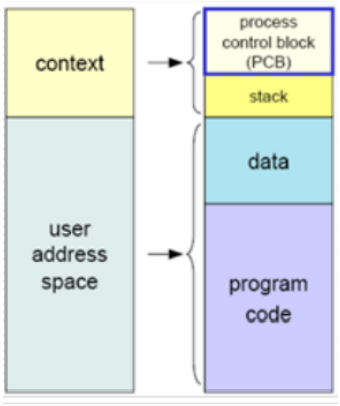
Еднонишкови процеси - притежават един брояч, определящ положението на всяка следваща инструкция;

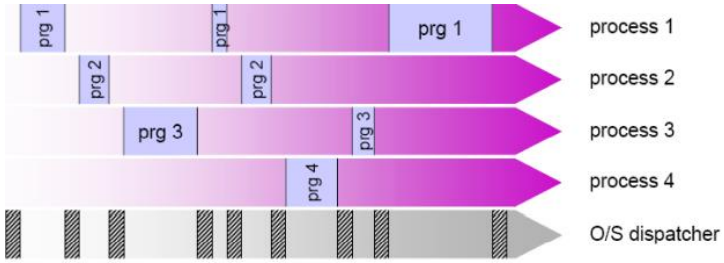
Многонишкови – притежават брояч за всяка нишка.

Многозадачността е характеристика на ОС, тя описва работата на множество процеси, стартирани паралелно.

Всеки процес се състои от изпълнима програма, данни за програмата и контекст на изпълнението на процеса - ID, състояние на процеса, CPU регистри, стек, и т.н.

ОС по отношение на процесите изпълняват следните дейности: създаване, унищожаване, спиране, възстановяване и предоставяне на механизъм за синхронизация и комуникация. Всеки процес има блок за управление (Process Control Block , PCB). PCB е включен в контекста заедно със стека; той съдържа всички необходими и достатъчни данни за рестартиране на процеса при необходимост (фиг. 7).





Фиг. 7 Блок за упражление

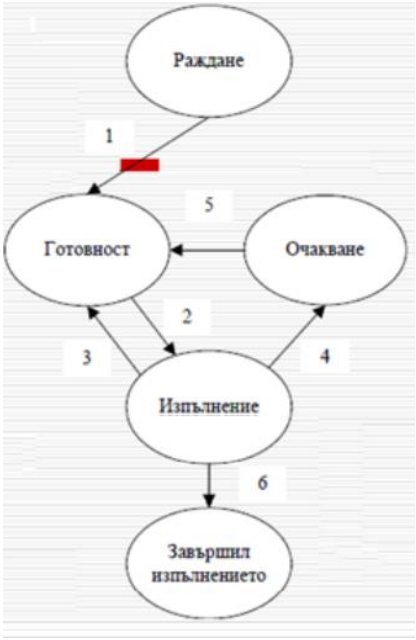
Фиг. 8 Диспечер

Диспечерът е програма в ядрото, която отговаря за превключването на процесора между отделните процесите (фиг. 8).

Процесите могат да се намират в пет основни състояния – раждане, готовност, изпълнение, очакване и завършване на изпълнението.

Диаграма на преходните състояния е представена на фиг. 9.

1. Допуснат до планиране
2. Избран за изпълнение
3. Прекъснат
4. Блокиран
5. Деблокиран
6. Край на изпълнението



Фиг. 9 Диаграма на преходните състояния

Виртуалната памет не е хардуерен компоненти, тя е услуга на ОС в защитен режим и използва вградените свойства на процесора за управление на външната памет. Виртуалната памет е част от дисковото пространство, което се третира от ОС като RAM и позволява едновременно да работят повече приложения. При използване на виртуалната памет:

- ✓ Програмата използва по-голяма от наличната памет
- ✓ Процесора достигне адрес който не съществува

- ✓ Генерира код за грешка за недействителна страница
- ✓ Активира програма за управление на виртуалната памет
- ✓ Програмата избира част виртуална памет и я прехвърля на диска (суопинг)
- ✓ Ако прехвърлената информация от диска е необходима, тя се копира отново в паметта.
- ✓ Диска се явява „склад“ за съхранение на части от неизползваната виртуална памет.

Процесора подава команда към твърдия диск да зареди програмата в паметта и не използва диска. Когато програмата е много голяма и не може да се побере изцяло в RAM, процесорът работи и с диска (swap файл), където се намира остатъка от програмата и общата работа на системата се забавя чувствително.

VII. Изброява и обяснява услуги в ОС. При конкретна поставена задача за стартиране и спиране на услуга ръчно или по график, избира правилния начин– 14т.

1. **SSH** – (***S**ecure **SH**ell – сигурна обвивка*) - мрежов протокол, позволяващ криптирано предаване на данни. Най-често се използва за изпълняване на команди на отдалечена машина, прехвърляне на файлове от една машина на друга и самото ѝ менажиране. **SSH** предоставя високо ниво на автентификация и сигурност по време на комуникацията между машините през незащитена връзка. Проектиран е да замести подобни протоколи, като TELNET. При изпращане на данни към мрежа, SSH автоматично ги криптира. След получаването им от крайния потребител, SSH (автоматично) ги декриптира, като по този начин потребителите работят, без да подозират, че техните съобщения се криптират и ползват безопасно мрежата. SSH използва клиент/сървърна архитектура. На сървъра се инсталира SSH програма от системния администратор, която приема или отхвърля изпратените заявки от SSH клиент до самата нея. Всички заявки между клиента и сървъра са криптирани.

Приложения на SSH

- посредством SSH клиент се осъществява отдалечена администрация на компютър с инсталиран SSH сървър, през терминална конзола.
- в комбинация с SFTP, като алтернатива на FTP (за сигурност), която може да се инсталира по-лесно в малки мащаби.
- в комбинация с SCP, като алтернатива на RCP пренос на файлове – по-често използван в UNIX среди.
- за пренасочване на портове или тунелинг (алтернатива на VPN). Пренасочената връзка е криптирана и защитена (между SSH клиента и сървъра). Приложенията на SSH за пренасочване на портове включват достъп до сървъри с бази данни, сървъри за електронна поща, защита на връзките през X11, VNC и отдалечен работен плот на Windows, и пренасочване на споделени файлове.

2. **FTP- File Transfer Protocol** (протокол за пренос на файлове, FTP)- мрежов протокол от тип клиент-сървър, предоставящ възможност за обмен на файлове между машини, свързани в локална мрежа или в интернет. FTP позволява на потребителите да изтеглят, качват, преглеждат, преименуват, изтриват файлове и др. Разработките на протокола включват варианти за криптирана комуникация и пренос на данните, наречени SFTP и FTPS, на основата на SSH. Режими на FTP връзка:

✓ активен FTP режим - порт 21 - за изпращане на контролни команди, които диктуват какво се случва по време на FTP сесия. Порт 20 - за реалния трансфер на данни.

✓ пасивен FTP режим - вместо произволен порт се използва порт по-голям от 1023.

✓ режими на трансфер - протоколът работи в двоичен или текстов ASCII режим.

FTPS - FTP изпраща файлове и идентификационни данни в мрежата в чист текст, т.е данните не са криптирани. Поради тази причина често се използва криптиран метод за прехвърляне FTPS (FTP Secure) или SFTP (SSH протокол за прехвърляне на файлове). FTPS е ефективно FTP с поддръжка за TLS (защита на транспортния слой). SSL е много по-рядко срещан от FTPS поради проблеми със сигурността.

Оновни типове FTPS връзки, неявни и изрични.

SFTP - SSH file transfer protocol е протокол за трансфер и манипулация на файлове през мрежова връзка. Обикновено се ползва с протокола SSH-2, предлагащ сигурен трансфер на потока данни. SFTP предлага и операции върху отдалечената файлова система. Допълнителните възможности на SFTP са: възобновяване на спрени трансфери, листинги на директории и изтриване на отдалечени директории. Протоколът не предлага удостоверение и сигурност – той разчита на по-долния протокол да осигури това. SFTP се ползва като подсистема на SSH-2. Възможно е SFTP да ползва SSH-1 или други потоци от данни. При SFTP сървър със SSH-1 се губи платформената независимост, тъй като SSH-1 не поддържа подсистеми. SFTP клиент, закачащ се към такъв сървър, трябва да знае къде (на коя система) се намира изпълнимия файл на SFTP.

3. Мрежови услуги (dns, dhcp)

DHCP (Dynamic Host Configuration Protocol - протокол за динамично конфигуриране на хостове) е комуникационен протокол, чрез който компютър, маршрутизатор или всякакъв друг вид устройство, използващо IP адрес, заявяват Интернет адрес от сървър, който от своя страна притежава определено пространство от IP адреси за раздаване. Чрез този протокол клиентите, се сдобиват със следните параметри: default gateway, subnet mask и IP адрес на DNS сървър. DHCP сървърът се грижи за уникалността на IP адресите – т.е. в подмрежата не може да съществуват два еднакви IP адреса по едно и също време, въпреки че един и същ адрес може да бъде раздаван на различни хостове в зависимост от времето на заявката за получаването му. На практика при подходяща конфигурация клиентите могат да правят заявки за всякакъв тип конфигурационни параметри от сървъра.

DNS- (**Domain Name System**) е разпределена база от данни за компютри, услуги или други ресурси, свързани към интернет или частни мрежи, с чиято помощ се осъществява преобразуването на имената на хостовете в IP адреси. Това улеснява работата на потребителите на интернет услуги. Вместо въвеждане на IP адрес (комбинация от цифри), за да достигне до даден ресурс в мрежата, потребителят може да въведе неговото име (домейн). Информацията за IP адресите и имената на домейни се съхранява на DNS сървърите. DNS е разпределена дървовидна система от обвързани чрез логическа йерархия сървъри. В основата на тази структура са сървърите, съхраняващи:

домейни от първо ниво (top-level domains) – например .com, .org, .edu и т.н. и

множество домейни на държавно ниво (country-level domains) – .bg (за България), .fi (за Финландия), .fr (за Франция) и т.н.

Следващото ниво образуват регистрираните домейни (registered domains) – about.com, abv.bg, pirin.com и т.н. Местните домейни (local domains), наричани още поддомейни (subdomains), като compnetworking.about.com, sdyn.pirin.com, се определят и администрират от собствениците на съответните главни домейни. За разделяне на различните равнища се ползва точка (.).

Системата за имена на домейни разпределят отговорността от възлагане на имената на домейните и свързването им със съответните им IP адреси чрез използването на „достоверни именни“ сървъри (authoritative name servers) за всеки един домейн. Този метод осигурява ниво на услугата с качества на разпределеност и устойчивост на грешки и причината за приложението му е в потребността за избягване на единна централизирана база от данни за управлението на системата за имената на домейните. Организацията, която се занимава с регистрирането и администрацията на домейните от първо ниво, е Internet Corporation for Assigned Names and Numbers (Интернет корпорация за присвоени имена и адреси).

VIII. Прави заключения и изводи за файловите системи на различните операционни системи. – 8т.

Файлови системи:

File Allocation Table (FAT) е файлова система на Microsoft, при която в началото на дисковия носител се разполага файл, който е таблица, съдържаща информация за физическото местонахождение на всеки файл в диска, на всеки фрагмент от този файл, на всяка директория, и описва йерархията от файлове и директории в едно дисково устройство.

FAT32 е файлова система за Windows 95, при която се използва 32-битово адресиране и това дава възможност размерът на файла да достигне до 4 GB, а големината на дяла или дискът, който се адресира, да достигне размери от порядъка на 8 TB. Тази система не позволява директно да се използват дълги имена на файлове1 (LFN). Използването на дългите имена се постига с VFAT (от Virtual FAT). Днес с FAT32 могат да се форматират флаш-устройства, дялове на дискове или по-малки дискове.

Предимства: подходяща за по-непретенциозни потребители и по-слаби машинни, напълно документирана и поддържана от всички ОС файлова система, лесен достъп от DOS (текстови, конзолен режим), в случай на сериозни поражения по операционната система, при обработка на видео и аудио е по-подходяща от NTFS, ако използваната машина е бавна, с недостатъчно памет или малък диск.

FAT32 недостатъци:

- лимитирана по отношение на максималния размер на диска;
- силно се поддава на фрагментиране на файловете;
- по-ниска скорост на достъп до данните, особено при повече на брой, малки по размер файлове;
- нестабилна, при внезапен срив е възможно да се достигне до повреждане на FAT и до загуба на данни;
- липсва поддръжка на квоти, сигурност и защита на данните.

Извод: по-подходяща за домашни потребители.

New Technology File System (NTFS) е разработена от Microsoft и IBM, заимствани от UNIX. Системата не е добре документирана. Позволява адресиране до 256 TB. NTFS е файловата система за Microsoft Windows NT, Windows 2000, Windows XP, Windows Vista, Windows 7, Windows 8.
NTFS предимства:

- журнална файлова система;

- вградена поддръжка на сигурност, на ниво "файл";

- поддържа множество едновременни потоци от данни;

- поддържа "*fault tolerance*";

- вградена поддръжка на компресия;

- криптирани данни;

- значително по-голяма скорост за достъп до данните от тази при FAT32, особено в небуфериран режим;

- по-малки размери на клъстерите.

NTFS недостатъци:
- частен и недостъпен публично формат на Microsoft. Липсва пълна поддръжка за четене и писане върху такива дялове, от други ОС;

- изисква повече памет и бърз процесор от FAT32.

Extended File Allocation Table (**exFAT**) – система на Microsoft, създадена за флаш устройства. Използва се при Windows XP, Windows Vista, Windows 7 и Windows 8.

Universal Disk Format (UDF) е файлова система за четене и запис за всички оптични носители. На практика UDF се използва най-често при DVD дискове.

Файлова система REFS в Windows 10 - нова файлова система, въведена в някои версии на Windows 10, защитата срещу загуба на данни (по подразбиране контролните суми на метаданни или файлове се съхраняват на дискове). По време на операциите за четене и запис данните във файловете се проверяват спрямо съхраняваните за тях контролни суми.

Разлики между файловата система REFS и NTFS

- по-висока ефективност, особено за дисковите пространства.
- теоретичен размер на обема е 262144 екзабайта (срещу 16 за NTFS).
- няма ограничение за пътя на файла до 255 знака (REFS има 32768 знака).
- REFS не поддържа имена на DOS файлове
- REFS не поддържа компресия, допълнителни атрибути, криптиране чрез файлова система

IX. Обяснява основни оператори в shell програмирането. Демонстрира знания за създаване на shell скриптове. – 10т.

Всички командни интерпретатори са описани във файла */etc/shells*. Когато е необходимо използването на друг, команден интерпретатор, той се добавя от root с помощта на командата *chsh* в директорията */etc/shells*.

Bash е не само команден интерпретатор, но и език за писане на скриптове. Шел (Shell) скриптовете се използват за автоматизиране на конкретни задачи.

Пример за написване на скрипт "Students PGTK":

• писане на код:

$!/bin/bash

echo "Students PGTK"

- записване на файла с код – използва се текстов редактор (vi, emacs, pico, mcedit)
- Skript.sh
- променянето на файла в изпълним – $ chmod u+x Script.sh
- стартиране – $ ./Skript.sh

2. Команди използвани в bash

- Командата echo разпечатва на екрана своите аргументи. В случай, че се зарежда изпълним файл които съдържа bash код, в началото му трябва да се укаже пътя към интерпретатора по следния начин:

Пример:

```
#!/usr/local/bin/bash
echo "SP!"
```

- Команди за създаване на променливи и функции. Типът на променливите не е задължителен, но при нужда може да се укаже посредством - declare.

Деклариране на променливи:

```
ime_promenliva="stoinost na promenlivata"
```

Например: *echo $ime_promenliva*

Създаване на функция:

- ✓ чрез ключовата дума function:

```
function ime_funciq
{
# kod na funciq
}
```

- ✓ със скоби:

```
ime_funciq()
{
# kod na funciq
}
```

Функция се извиква по име, без скоби или допълнителни символи:

```
ime_funciq "параметър 1" "параметър 2"
```

параметрите на функцията се задават от $1 до $n където n е число и е последен параметър. Те могат да се изместват с вградената команда shift, например:

```
echo $1
shift 2
# $3 става $1
```

Дефиниране на променливи

int - цяло число

int1 - eq int2 Връща True ако int1 е равно на int2 i

int1 - ge int2 Връща True ако int1 е по-голямо или равно на int2.

int1 - gt int2 Връща True ако int1 е по-голямо от int2.

int1 - le int2 Връща True ако int1 е по-малко или равно от int2

int1 - lt int2 Връща True ако int1 е по-малко от int2

int1 -ne int2 Връща True ако int1 не е равно на int2

Сравнения на низове

str1 = str2 Връща True ако str1 е идентична на str2.
str1 != str2 Връща True ако str1 не е идентична на str2.
str Връща True ако str не е null (т.е. празен)
-n str Връща True if дължината на str е по-голяма от null
-z str Връща True ако дължината на str е равна на 0. (0 е различна от null)

X. Прави изводи за виртуализация и контейнери. – 8т.

Виртуализация (*virtu-alization*) прави възможно стартирането на множество ОС и приложен софтуер на една хардуерна машина, позволява ефикасното използване на наличните ресурси. Една ОС не прави разлика между виртуална или физическа машина. Виртуалната машина е съставена изцяло от софтуер и не съдържа каквито и да е хардуерни компоненти, тя предлага редица предимства в сравнение с отделен физически хардуер:

- Съвместимост - съвместими с всички стандартни x86 компютри.
- Изолация - физически отделни компютри.
- Капсулация - капсулират пълна компютърна среда.
- Хардуерна независимост.

Видове:

Виртуализация на хардуер - мениджърът на виртуални машини (VM), който се извиква като хипервизор, прави възможно виртуализацията на хардуера. Хипервизорът създава и консолидира виртуални версии на компютри и операционни системи в един голям физически сървър, което прави възможно използването на всички хардуерни ресурси по-ефективно. Потребителите могат да извършват едновременно няколко ОС на една машина.

Виртуализация на приложения – в Application virtual приложенията се виртуализират на сървъра. След което се изпраща до устройствата на крайните потребители. Вместо да влизат в работните си компютри, потребителите имат достъп до приложението директно от устройството си с подходяща интернет връзка.

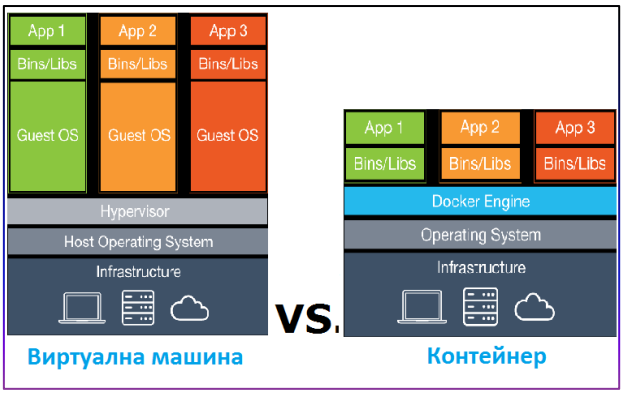
Виртуализация на сървър – маскиране на сървърните ресурси от потребителите на сървъри, включително броя и идентичността на физическите сървъри, процесори и ОС.

Виртуализация на мрежа предоставя обобщение за работа в мрежа и услуги чрез хардуер в логическа, виртуална мрежа, която е свързана с физическа мрежа на хипервизор и работи независимо от мрежата. Виртуализацията на мрежата може да осигури мрежа, която е независима от други мрежови ресурси във виртуална среда.

Виртуализация на работен плот (виртуализация на клиента) - настолна техника, която отделя физическия компютър от работната среда. Предимство е дистанционното влизане от всяко място за достъп до работен плот. Тази виртуализация се използва в модели на сървърни изчисления.

Виртуализация на съхранението- предпазва от внезапни сривове в системата. В момента на срив или липса на хардуер или софтуер, потребителят може да възстанови данни. Недостатъци - не всяко приложение или сървър работи в среда за виртуализация. Налага се инвестиране в обучение на съществуващи мрежови администратори, които нямат възможности да управляват виртуална мрежа.

Контейнери (Фиг.10) - изолирана инстанция на потребителското пространство на ОС. Контейнерите, благодарение на ядрото на ОС или друг софтуер (Container engine) разпределят и изолират ресурсите на ОС в отделни пространства на имената (name-spaces). Те се изпълняват директно върху физическата машина, но приложенията им използват само част от ресурсите (процесор, памет, файлове и папки, В/И устройства и др.);



Фиг. 10 Контейнери