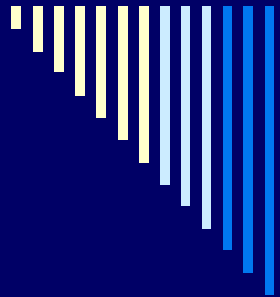


# 13. ZOS

**ZOS 2008, L. Pešička**

---



# Popis filesystemů

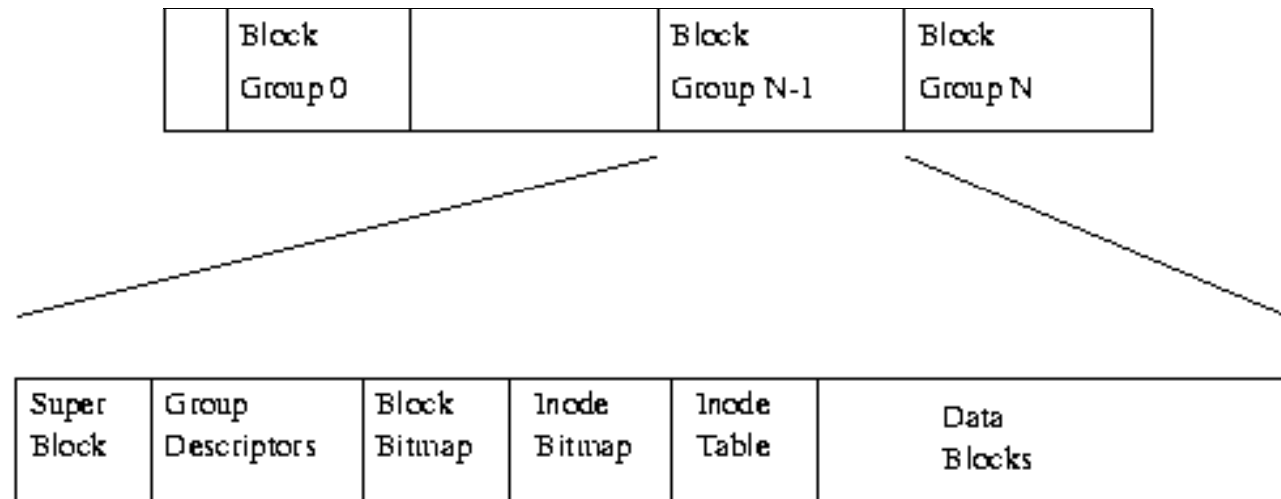
## Odkazy

<http://www.tldp.org/LDP/tlk/fs/filesystem.html>

<http://www.abclinuxu.cz/clanky/system/na-co-se-casto-ptame-etc-fstab>

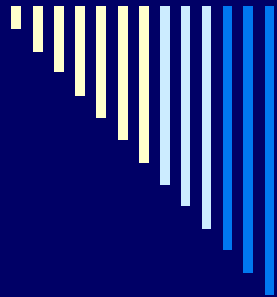
---

# Ext2 filesystem



Partition je rozdělena do Block Groups

Každá skupina duplikuje informaci kritickou pro integritu fs



## Ext2 inode

**Mode** – file, adresář,  
blok/znak zařízení

Vlastník (uživ, skupina)

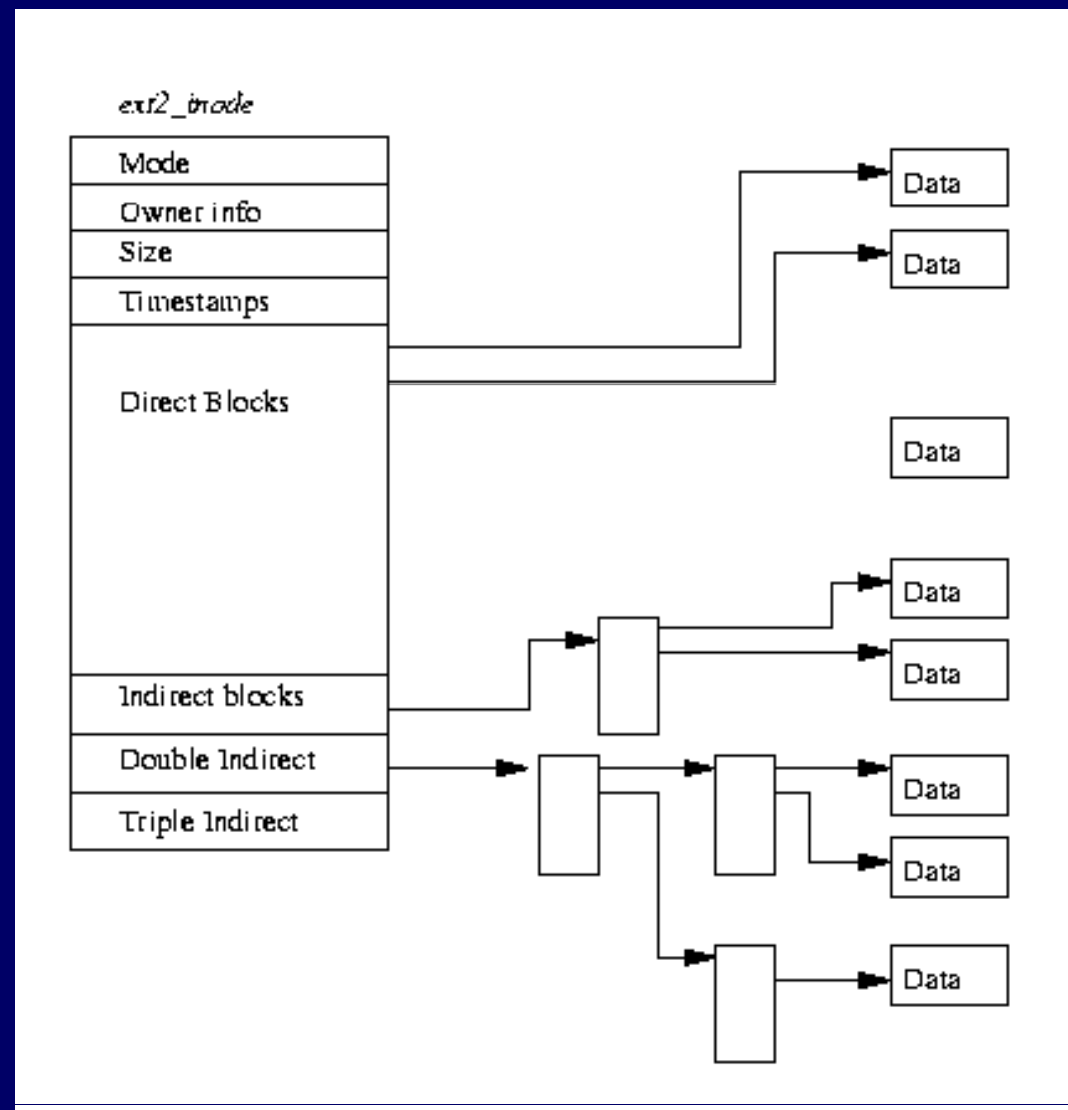
Size (v bytech)

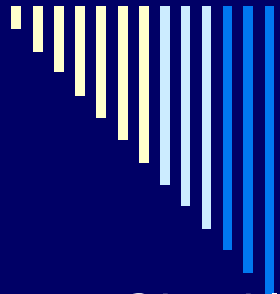
Timestamp (vytvoření,  
modifikace)

### Datové bloky

12 ukazatelů přímých

3 různě nepřímé úrovně





# Ext2 superblok

Obvykle se při mountování čte superblok z block group 0  
Každý Block Group kopii pro případ poškození

**Magic number** – říká, že jde opravdu o ext2 fs

**Revision Level** – jaké nové vlastnosti lze bezpečně použít

**Mount count, Maximum mount count**

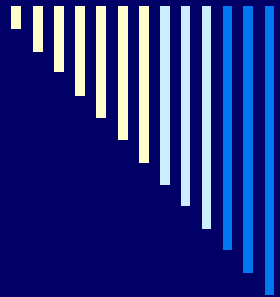
- Zvětšuje počet přimountování, po dosažení maxima obvykle puštění kontroly konzistence, e2fsck

**Block group number**

**Blok size** – velikost bloku v bytech, např. 1024, při vytvoření

**Block per Group** – počet bloků ve skupině, fixováno při vytvoření

**Free blocks, Free Inodes, First inode** (obsahuje adresář /)



# EXT2 group descriptor

- Jako superblok, všechny group desc. pro všechny block groupy duplikovány v každém block group

**Blocks bitmap** – kde je bitmapa bloků pro tuto skupinu

**Inode Bitmap** – kde je bitmapa iuzlů této skupiny

**Inode Table** – číslo bloku, kde začíná inode table pro tuto skupinu

**Free blocks count**

**Free inodes count**

**Used directory count**

---

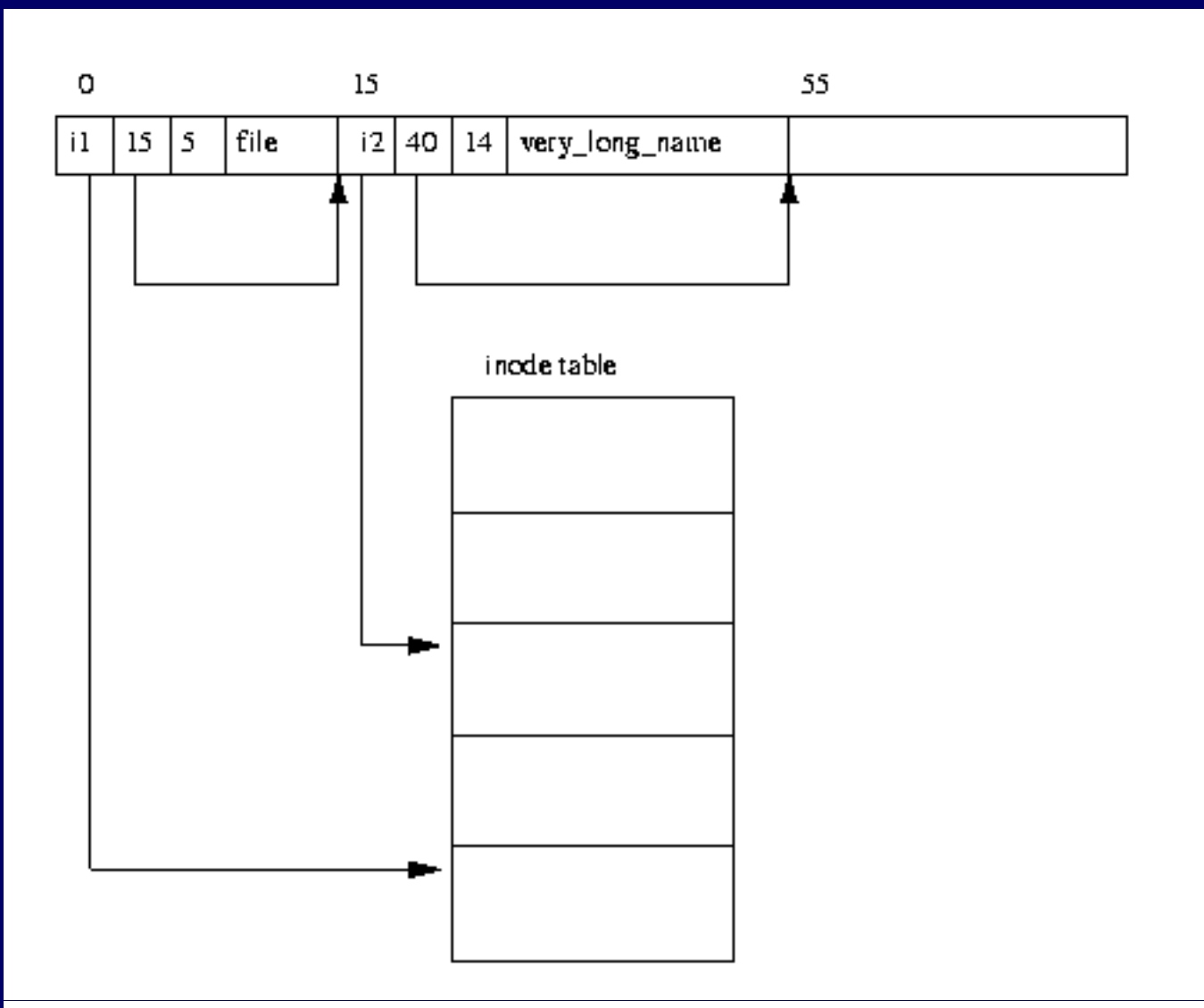
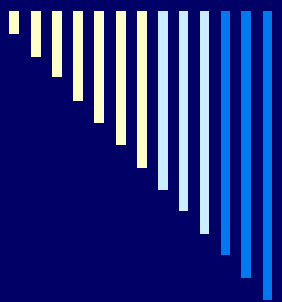


# Ext2 adresář

- Adresář – speciální soubor

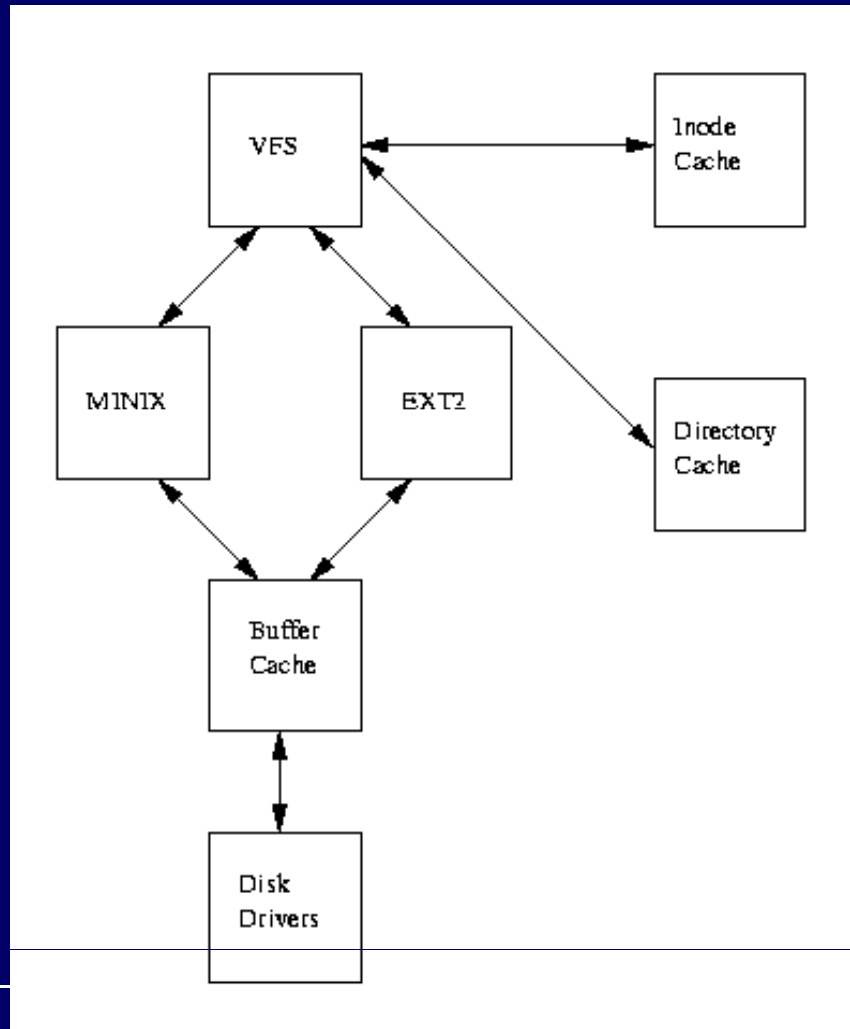
## Záznam v adresáři

- **Inode** – i-uzel pro danou položku (soubor)  
index do pole inodů v Inode table
- **Name length** – délka adresářové položky v bytech
- **Name** – jméno adresářové položky (souboru)





# VFS (Virtual File System)





# Mounting a file system

□ `mount -t iso9660 -o ro /dev/cdrom /mnt/cdrom`

Soubor `/etc/fstab`:

```
/dev/fd0 /mnt/floppy vfat noauto,user 0 0
```

1. Zařízení, které připojujeme
  2. Připojovací bod (adresář)
  3. Typ souborového systému
  4. Parametry připojení
  5. Zálohování (pro dump 0 ne, 1 ano)
  6. Pořadí kontroly při startu
-



---

# Parametry připojení

**noauto** – nepřipojuje automaticky při startu

**users** – s tímto svazkem mohou pracovat i běžní uživatelé

**codepage** – znaková sada, ve které jsou názvy souborů

**iocharset** – znaková sada, do které se budou převádět názvy souborů

**noexec** – nespouštět soubory na tomto médiu

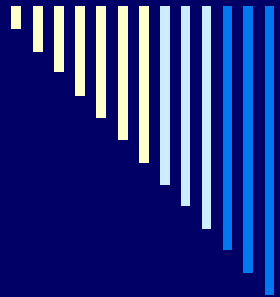
**ro, rw** – samovysvětlující 😊

---



# Sdílení souborů

- soubor/adresář viditelný  
ve **více podadresářích**  
pod **více jmény**
- Pevné odkazy (hardlinks)
  - Více odkazů na stejný **i-uzel**
  - Všechny odkazy rovnocenné
  - V i-uzlu je počet odkazů, zrušení když je = 0
- Symbolické odkazy
  - Obsahují **jméno odkazovaného** souboru
  - Obecnější, větší režie



# Správa volného prostoru

(které bloky jsou volné?)

## □ Seznamy diskových bloků

- Blok obsahuje odkaz na volné bloky a adresu dalšího bloku
- Problém najít blok s určitými vlastnostmi

## □ Bitové mapy

- používanější
- Konstantní velikost
- Snadno najdeme volný blok s určitými vlastnostmi



# Kvóty

- Maximální počet bloků obsazených soubory uživatele
  - Pro víceuživatelské systémy
  - Soft a hard kvóta
    - **Soft** – krátkodobě překročitelná
    - **Hard** – nepřekročitelná
  - Problémy
    - Profile windows x dostatečná kvóta
    - Potřeba nárazově zpracovat větší množství dat (adresáře tmp, scratch, ...)
    - Přesnost určení kvóty (započítání ADS z NTFS?)
-



---

# Konzistence fs

- OS blok přečte, změní, zapíše
  - Havárie před zápisem všech modifikovaných bloků
  - **Nekonzistentní** stav fs
  - Kontrola konzistence
    - Win: *chkdsk, scandisk*
    - Unix: *fsck.ext2 /dev/sda2*
-



---

# Mechanismus ochrany

- co je potřeba chránit – **objekt**
    - *cokoliv, k čemu je potřeba omezovat přístup*
    - soubory
    - hw objekty (segmenty paměti, I/O zařízení)
    - sw objekty (procesory, semaforey)
  - **subjekt**
    - entita, schopná přistupovat k objektům
-

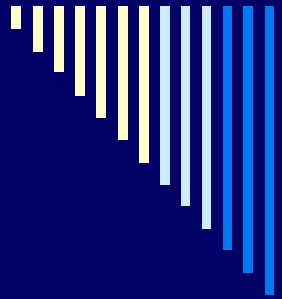




---

# Přístupové právo

- *Read, Write, Execute, Delete*
  - právo vykonat jednu z možných operací
  
  - informace o přístupových právech
  - ACL
    - objekt má seznam subjektů a jejich přístup. práv
  - capability list
    - se subjektem je sdružen seznam objektů a přístupových práv k nim
-



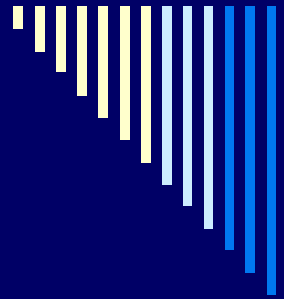
# Poznámky

## Linux

- přístupová práva – symbolický link
- setuid bit

## Windows

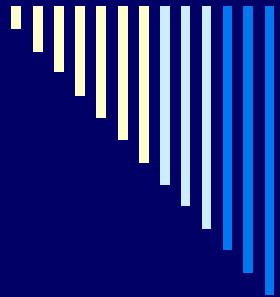
- AD



---

# Ukázka – AD uživatelé, skupiny

- dcpromo
  - Active Directory Users and Computers
-



# Ukázka – vytvoření uživatele

```
net user %2 abcdefgh /add /fullname:%3 /homedir:j:\  
  /profilepath:\\herakles\students\%2\profile-KIV /scriptpath:login-KIV-  
  students.bat /passwordchg:no
```

```
mkdir E:\students\%2
```

```
caccls E:\students\%2 /e /g %2:f
```

```
mkdir E:\students\%2\profile-KIV
```

```
ksetup /domain /mapuser %2@ZCU.CZ %2
```



# GNU / Linux

## □ Jádno + aplikační sw

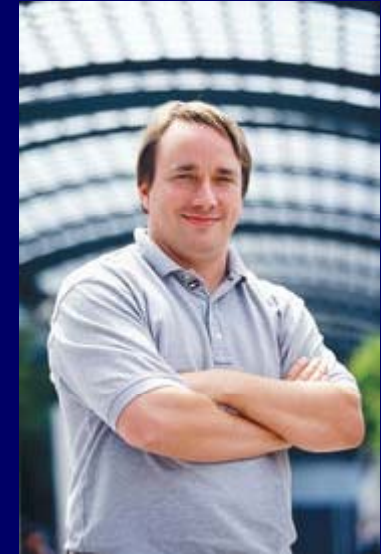
- Linuxové jádro
- Knihovny a nástroje GNU

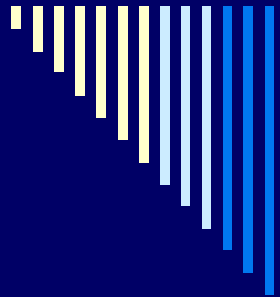
Linus Torvalds

Richard Stallman

## □ GNU / Hurd

- Např. Debian GNU/Hurd
- Množina serverů pracujících nad mikrojádro
- GNU Mach jako mikrojádro aktuálně





# Linux

- Volně šířený systém – kdokoliv si ho může nainstalovat a dále šířit
  - Zdrojové texty – kdokoliv může modifikovat jádro nebo systémové programy
  - Vypadá a chová se jako UNIX – kompatibilita
  - Různé distribuce
-



# Komponenty Linuxu

## □ Monolitické jádro OS

- Jediné běží v režimu jádra
- Poskytuje základní abstrakce (procesy, virtuální paměť, soubory)
- Aplikace žádají o služby jádra prostřednictvím **volání služeb** systému
- Umožňuje za běhu přidávat a vyjímat **moduly** (ovladače)
  - Přidat / zrušit části kódu jádra na žádost

## □ Systémové knihovny

- Standardní funkce pro použití v aplikacích
  - Některé komunikují s jádrem OS (write apod.)
  - Mnoho knihovných funkcí jádro nevolá (sin, cos, tan, ..)
-





---

# Komponenty Linuxu

## □ Systémové programy

- Nastavení konfigurace systému při startu
- Démonové procesy
  - inetd – obsluha příchozích síťových spojení
  - Syslogd – zápis logů

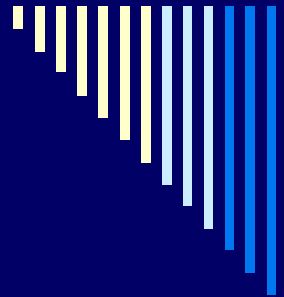
## □ Uživatelské programy

- ls, cat, joe, ...
-



# Linux – jádro systému

- Linux – jádro systému; od roku 1991
  - Uživatelské programy, knihovny, většina systémových vznikla v nezávislých projektech (především GNU – ls, cp, ..)
  - GNU / Linux, GNU / Hurd
  
  - Distribuce (RedHat, SUSE, Mandriva, Debian)
    - Jádro
    - Programy z různých zdrojů
    - Správa balíků (RedHat – rpm, Debian – dpkg)
-



# Privilegované módy v různých systémech

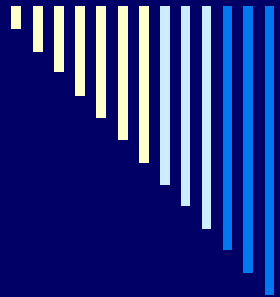
- Linux – 2 úrovně (User a Kernel mód)
- FreeBSD – také 2
- VMS – 4 (Kernel, Executive, Supervisor, User)
- Multics – 8 úrovní
  
- **User mode**
  - nedovolí privilegované instrukce – ohrozit systém
  - omezen přístup do paměti
  - zakázána komunikace na I/O portech



---

# Přístup do paměti - Linux

- mechanismus **stránkování**
    - program přistupuje na virtuální adresu
    - využití tabulky stránek pro převod na fyzickou adresu
    - mapování není – výjimka
    - v tabulce stránek – práva přístupu a druh přístupu
    - cache na několik posledních mapování – TLB (transaction lookaside buffer)
-



# Procesy, paměť

- procesy běží v USER modu, celé jádro v KERNEL modu
- každý proces – vlastní tabulka stránek – určuje jeho adresový prostor
- kód i data jádra – sdílena mezi všemi procesy, z KERNEL modu vždy přístupné
- nepoužívá se segmentace – je **lineární** adresový prostor



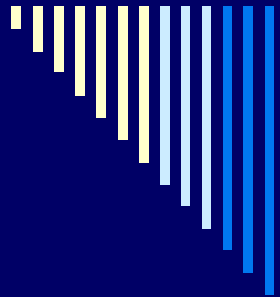
# Přepínání procesů

- proces – USER mod – obsluha systémového volání nebo zpracování výjimky – KERNEL mód
  - USER mód – **preemptivní** multitasking
    - k přepnutí může dojít kdykoliv, rozhodnutí plánovače
  - KERNEL mód – **kooperativní** multitasking
    - když sám požádá
    - čekání na nějakou událost (zablokování procesu)
    - zjednodušuje návrh jádra
    - preemptivní kernel – lze při kompilaci jádra 2.6 – lepší interaktivita
-



# Přepínání procesů

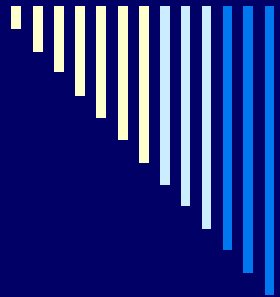
- kernel thready
  - nemají vlastní tabulku stránek ani uživ. adres. prostor
  - běží celou dobu v jádře
  - činnosti na pozadí – např. zápis modifikovaných bufferů
  - druhý význam pojmu (thready přepínané v jádře)
  
- Linux – přepnutí procesu v jádře
  - `cond_resched()` nebo `schedule()`



# Linux – vytvoření procesu

- Oddělení operace **vytvoření procesu** a **spuštění nového programu**
  
- **Vytvoření nového procesu**
  - Volání **fork ()**
  - Nový proces provádí stejný kód jako původní
  
- **Spuštění nového programu**
  - Volání **execve()**





# Fork ()

- ❑ Přesná kopie původního procesu
- ❑ Po vytvoření mají oba procesy stejný stav (obsah proměnných..)
- ❑ Každý z nich běží ve vlastní kopii VM
- ❑ Oba pokračují v běhu návratem z volání fork() a nadále běží nezávisle
- ❑ pid = fork()
- ❑ Původní proces – rodič, nově vytvořený – potomek
- ❑ Fork vrací
  - Pro rodiče – PID potomka
  - Pro potomka – vrací nulu



# Implementace fork()

## □ První generace Unixů

- Fork zduplikoval celý adresní prostor rodiče fyzickým kopírováním
- Časově náročné, navíc často hned exec – tedy zbytečně
- vfork() – nedělá kopírování, potomek nesmí modifikovat data, smí pouze exec nebo exit

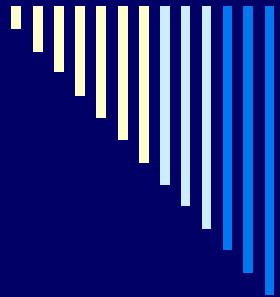
## □ Moderní systémy – copy on write

- Stránky sdíleny nejprve jako read only
- Zápis do stránky – výjimka, zduplikuje a nová kopie read write
- Původní rámec zůstává read only, zápis – zkontroluje, zda je jen jeden vlastník – přeznačí na read write



# Implementace fork()

- V Linuxu využívá volání jádra **clone()**
- Obecnější, lze specifikovat co bude rodič a potomek sdílet a co bude duplikováno (paměť, kořenový a pracovní adresář, otevřené soubory, obslužné rutiny pro signály)
- Fork – nesdílí nic
- Vfork – sdílí virtuální paměť



# Spuštění programu

- Služba `execve` nahradí obsah paměti volajícího procesu procesem spuštěným ze zadaného souboru
- `If ( fork() == 0 )`  
`execve ("/bin/lS", argv, envp);`
- Proces spustí svého potomka, který místo sebe spustí `/bin/lS`



---

# Ověřování uživatelů

- Autentizace (authentication)
  - Jednoznačné prokázání totožnosti (identity) uživatele
  - **Identifikace** – vůbec vědět, kdo je
  - **Autentizace** – ověření, zda je opravdu tím, za koho se vydává
  - Uživatel zadá jméno (identifikuje se), po zadání hesla dojde k jeho autentifikaci – systém ověří, zda je opravdu tím, za koho se vydává
-



---

# Metody autentikace

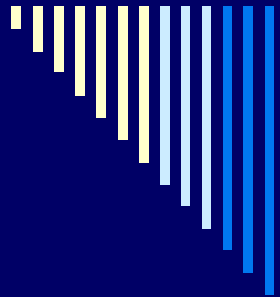
- Uživatel něco **zná**
    - Heslo
    - Transformační funkce (výzva – odpověď)
  - Uživatel něco **má**
    - Kalkulátor pro jednorázová hesla
  - Uživatel má určité **biometrické vlastnosti**
    - Otisky prstů
    - Sítnice oka
    - ? Spolehlivost
-



---

# Autorizace

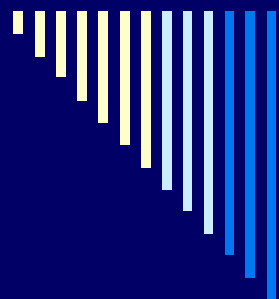
- **Řízení přístupu** ke zdrojům systému
  - Např. zda máme právo nějaký soubor číst, zapisovat do něj..
  - Musíme systému prokázat kdo jsme (autentizace) a on ověří, zda pro danou činnost máme právo (autorizace)
-



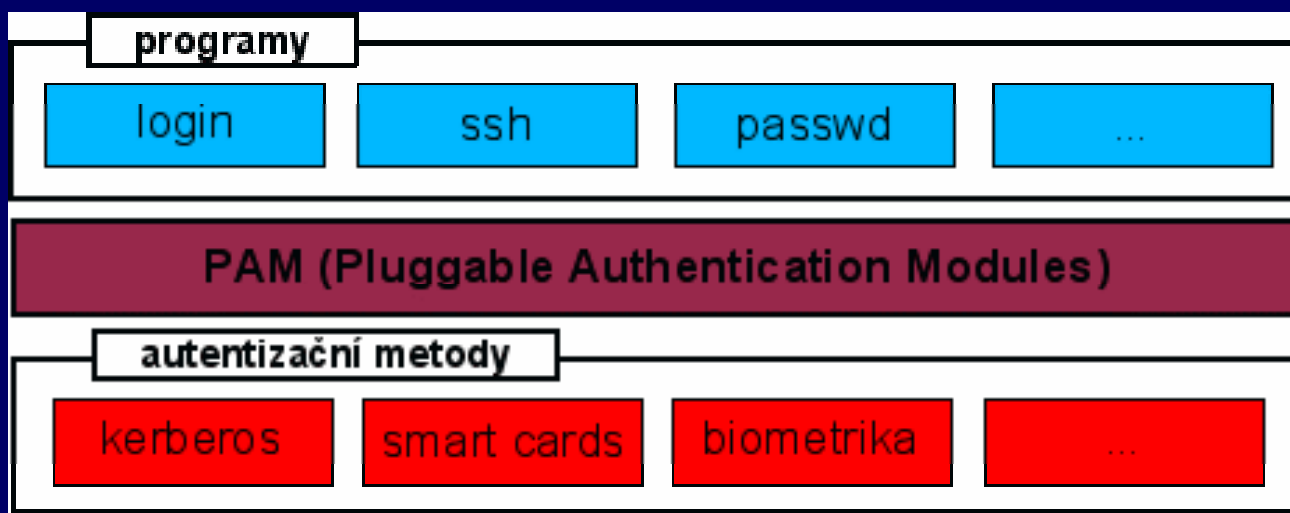
# PAM (Pluggable Authentication Modules)

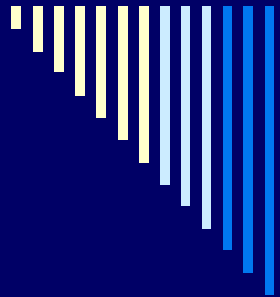
- Zajistit nezávislost programů na konkrétních autentizačních postupech
- Aplikaci zajímá pouze to, zda má službu poskytnout nebo ne
- Mezivrstva
  - Aplikace, požadující ověření identity
  - Autentizační metoda (/etc/passwd, Kerberos, biometrie)





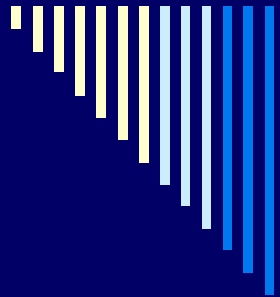
# PAM





# PAM

- 4 funkční oblasti
- **Autentizace uživatele**
  - Ověřuje identitu uživatele – kontrola hesla, připojení k serveru Kerberos, ...
- **Kontrola účtu**
  - Např. povolení přístupu jen během určité doby
  - Omezení počtu současně přihlášených uživatelů
- **Správa relace**
  - Před a po provedení služby
  - Nastavuje proměnné prostředí, omezuje systémové prostředky
  - Připojení dalších filesystemů, chroot, ...
- **Změna hesla**
  - Kontrola délky a kvality hesla



# Kerberos

- Použití tzv. **lístků** vydávaných centrálním autentizačním serverem (spravuje databázi všech uživatelů)
- Podpora **single sign-on**
- Uživatel se **autentizuje** vůči serveru Kerbera pouze **jednou**
- Získá základní **lístek** – Ticket Granting Ticket (TGT)
- TGT použije pro získání **dalších** lístků pro přístup ke službám vyžadujícím **autentizaci**
- Kromě prvotní autentizace vše ostatní transparentně
- Použití např. distribuovaný souborový systém AFS



---

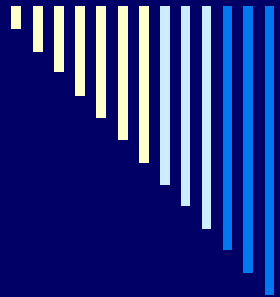
# Kerberos, distribuovaný fs AFS

## □ Kerberos

- **kinit** – získání TGT lístku
- **klist** – seznam akt. lístků na disku
- **kdestroy** – zničení uložených lístků
- **kpasswd** – změna kerberovského hesla

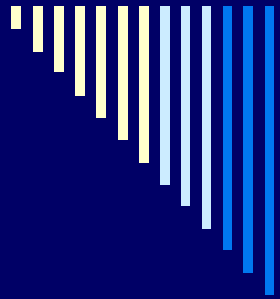
## □ AFS

- oprávnění k systému afs - token
  - **aklog** –c kiv.zcu.cz
  - **tokens**
-



# Škodlivé programy

- Malicious software
  - Vyžadují hostitelský program nebo nezávislé
  - **Bacteria**
    - Konzumuje systémové zdroje tím, že se replikuje
  - **Logic bomb**
    - Pokud v systému nastanou určité podmínky, provede se nějaká škodlivá akce.
  - **Trapdoor**
    - Tajný nedokumentovaný vstupní bod do programu, obejdou se běžné přístupové kontroly
-



# Škodlivé programy

## □ Trojan Horse

- Tajná nedokumentovaná rutina obsažená v jinak užitečném programu. Při spuštění programu dojde i k vykonání tohoto kódu.

## □ Virus

- Kód zapouzdřený v programu, kopíruje se a vkládá do dalších programů. Kromě vlastního šíření vykonává různé škodlivé činnosti

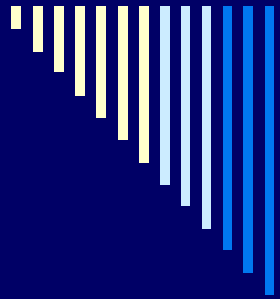


---

# Škodlivé programy

## □ Worm (červ)

- Program může replikovat sebe a šířit se z počítače na počítač přes síť. Na cílovém stroji ve své činnosti pokračuje. Může vykonávat další škodlivé činnosti.
-



# Viry

- Parazitní
- Memory-resident
- Boot sector
- Stealth virus
  - Skrývá se před detekčními programy
- Polymorphic
  - Mění se kód viru
  - Snaží se bránit detekci podle signature viru





# Antiviry

- **Detekce**

- Zjistit, že je soubor napadený

- **Identifikace**

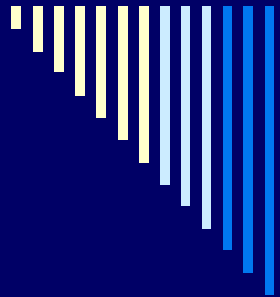
- Zjistit, o jaký virus se jedná

- **Odstranění**

- Databáze **signatur** virů (aktualizace)

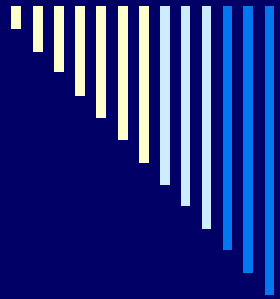
- Heuristika

---



# Koncept verifikačního monitoru

- Reguluje přístup subjektů (uživatelské procesy) k objektům (paměť)
- Verifikační monitor má přístup k security kernel databázi
  - Seznam přístupových privilegií každého subjektu
  - Seznam ochranných atributů každého objektu
- Audit file
  - Loguje důležité události
  - Přístupy, změny přístupových práv



# Bezpečnostní chyby

- Příklad starší bezp. chyby
- Lpr vytiskne zadaný soubor na tiskárnu
- Lpr -r soubor po vytisknutí zruší
- Ve starších verzích Unixu bylo možné, aby kdokoliv vytisknul a zrušil /etc/passwd



# Vnější útoky na systém

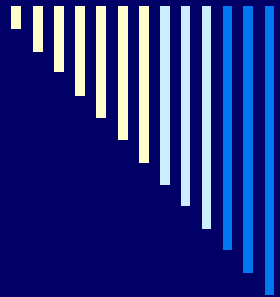
- Vyhledání vhodného systému
    - „skenování“ sítě
  - Zjištění informací o počítači
    - OS, spuštěné servery
  - Napadení systému
  
  - Problém – pro všechny fáze existují na síti vytvořené nástroje, útočník nemusí být ani žádným odborníkem na dané téma
-



---

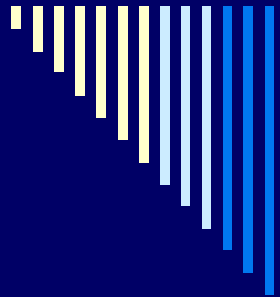
# Zjištění informací o počítači

- Zjištění typu stroje (fingerprint)
    - Např. reakce na nesmyslné příznaky (SYN-FIN-URG-PUSH) - každý OS a protokolový zásobník odpoví jinak
    - Odposloucháváním provozu
  - Scan portů
    - Servery poslouchají na určeném portu, nejčastěji na well-known portech
-



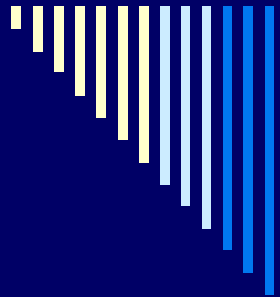
## Přetečení bufferu uloženého na zásobníku

- ❑ Programy vytvořené v C
- ❑ Nekontroluje např. meze polí, umožňuje přepsat část paměti
- ❑ Volání procedury – na zásobník návratovou hodnotu
- ❑ Spustí proceduru, vytvoří místo pro lokální proměnné
- ❑ Uživatel zadá např. delší řetězec, přepíše část další paměti, včetně návratové adresy
- ❑ Návrat – instrukce RET – začne vykonávat instrukce od jiné adresy (pokud náhodné, program většinou zhavaruje)



# Odkazy

- Některé použité při zpracování této přednášky
  
  - Báječný svět Linuxu 2.6 (z roku 2003)
    - <http://www.linuxzone.cz/index.phtml?ids=10&idc=782>
  
  - Porovnání systémů Linux a FreeBSD
    - <http://www.root.cz/serialy/porovnani-systemu-linux-a-freebsd/>
  
  - PAM
    - <http://www.root.cz/clanky/pam-sprava-autentizacnich-mechanismu/>
    - <http://www.root.cz/clanky/pam-pouziti-v-praxi/>
-



## Odkazy 2

### □ PAM, Kerberos

- [http://www.fi.muni.cz/~kas/p090/referaty/2005-jaro/st/referat\\_kerberos\\_pam.html](http://www.fi.muni.cz/~kas/p090/referaty/2005-jaro/st/referat_kerberos_pam.html)

### □ Kerberos

- <http://meta.cesnet.cz/cs/docs/software/system/kerberos.html>