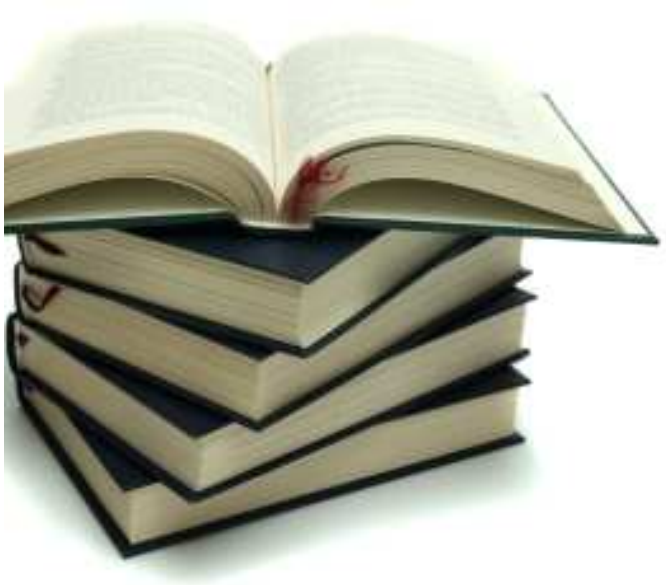
so2
kernel crunch

Cursul 2



2

Arhitectura Linux/Windows

5 martie 2009

Test Curs 1

- Care din următoarele componente vor rula, în general, în user-space, respectiv kernel-space într-un kernel monolitic sau microkernel: server web, linker, subsistemul de I/O, sistemul de fișiere, memory management, process scheduling, rutine de tratare a întreruperilor?
- Pe un sistem de 32 de biți cu 2GB memorie şi 64GB spațiu swap, cât spații de adresă virtuale se pot crea?
- De ce nu poate fi următoare secvență de cod folosită în kernel?

```
void f(char *c, int len)
{
    char tmp[64*1024];
    memcpy(tmp, c, min(len, sizeof(tmp));
```

Arhitectura Linux/Windows

- Modelul de dezvoltare Linux
- Arhitectura Linux
  - Arch
  - Device drivere
  - Core kernel: I/O management, VFS, memory management, security framework, network stack
  - Procese si fisere de bază
- Arhitectura Windows:
  - HAL
  - Device Drivere
  - Kernel, Executiv
  - Susbsisteme de mediu
  - Procese si fisere de bază

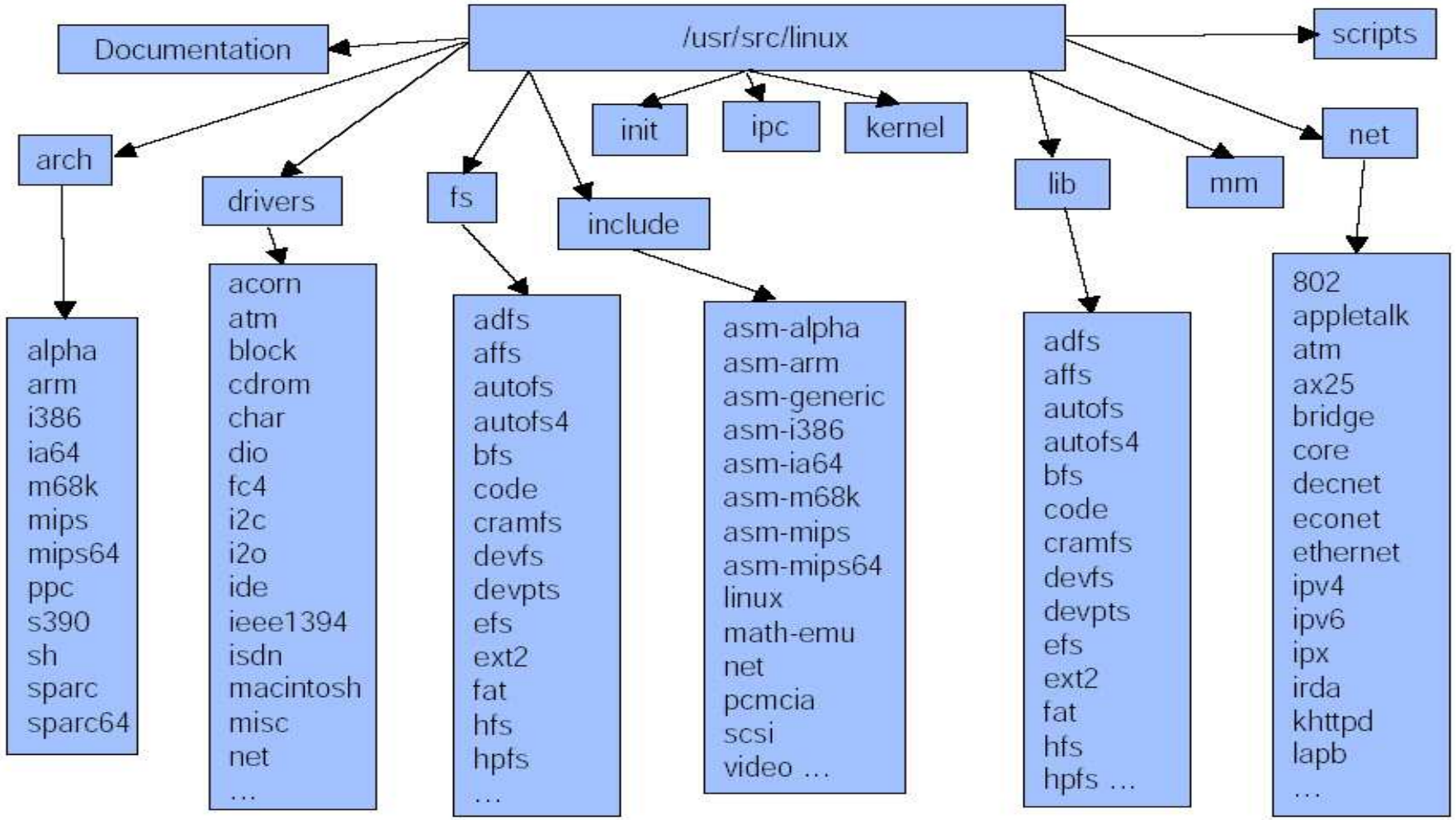
Modelul de dezvoltare Linux

- Open source
  - Companii (RedHat, Novell, SGI, IBM, Intel, Oracle, MontaVista, Qumranet, Nokia, HP, Google, etc.) (concurente) lucrează împreună
- Vechiul model
  - Versiuni stabile = pare: 1.0, 1.2, 2.0, 2.2, 2.4, 2.6
  - Versiuni de dezvoltare = impare: 0.x, 2.1, 2.3, 2.5
  - Ciclul de dezvolatare: 2 – 3 ani
- Noul model:
  - Fiecare versiune 2.6 este stabilă,
  - Ciclul de dezvoltare: 3 – 4 luni

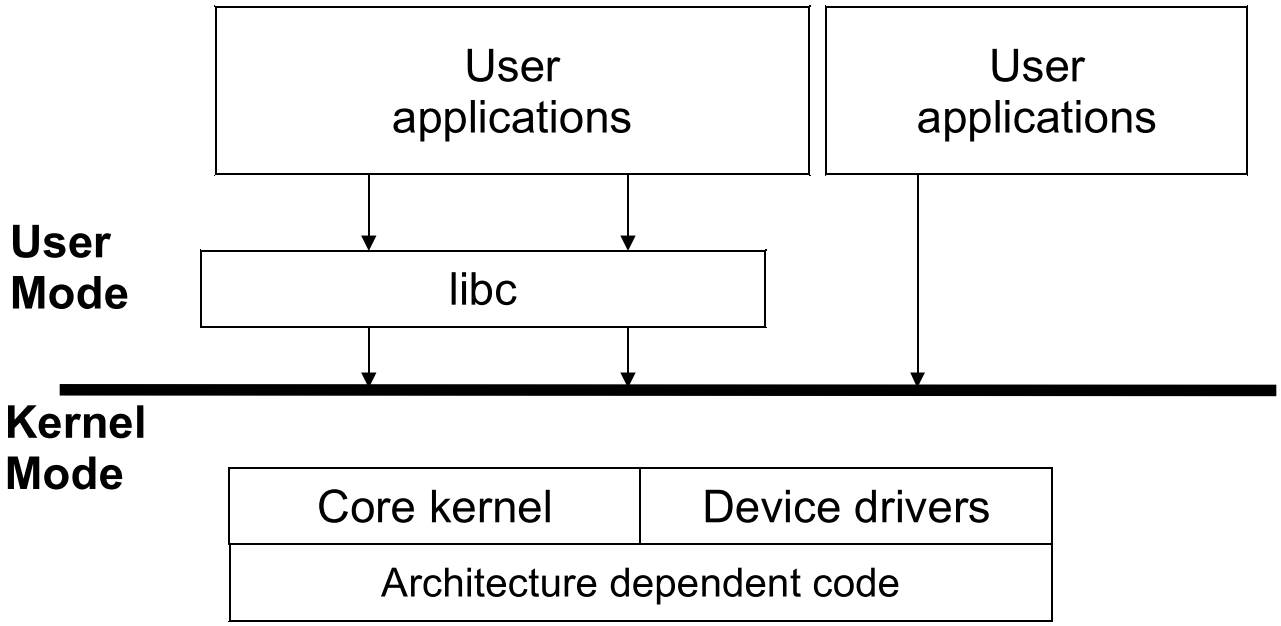
„Git trees”

- Oficial: Linus Torvalds
  - git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux-2.6.git
- -mm git tree: Andrew Morton
- Fiecare distribuție își menține propriul tree

Organizarea surselor



Arhitectura Linux



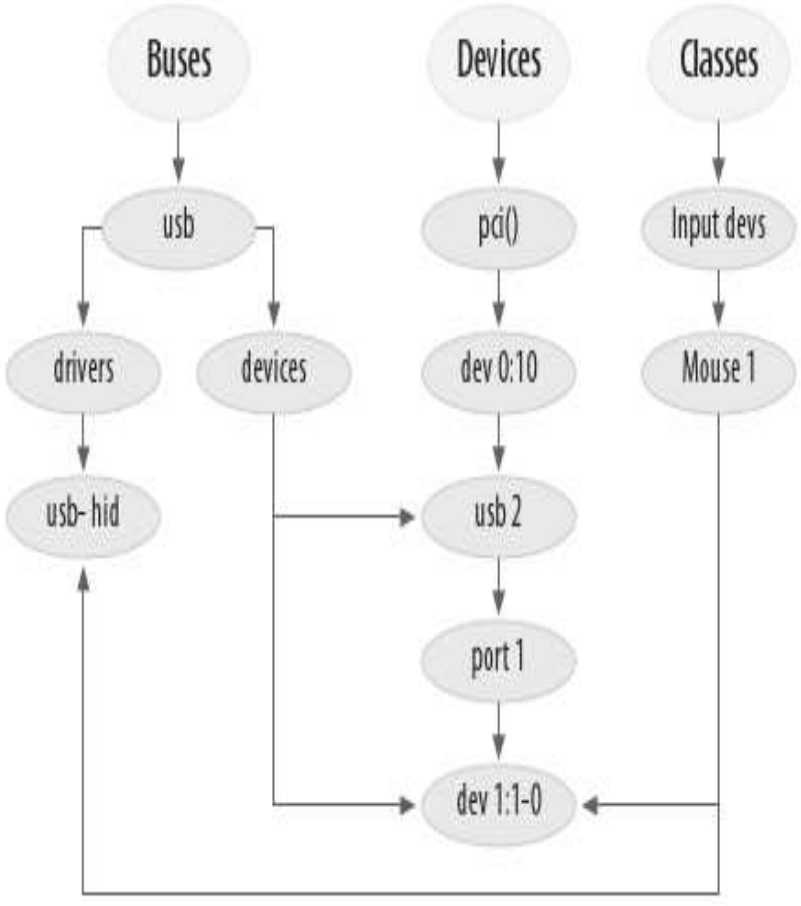
Arch

- Cod dependent de arhitectură
- Fiecare arhitectură poate conține mai multe sub-arhitecturi
- Linux este unul dintre cele mai portate kernele:
  - “Linux was first developed for 32-bit x86-based PCs (386 or higher). These days it also runs on (at least) the Compaq Alpha AXP, Sun SPARC and UltraSPARC, Motorola 68000, PowerPC, PowerPC64, ARM, Hitachi SuperH, IBM S/390, MIPS, HP PA-RISC, Intel IA-64, DEC VAX, AMD x86-64 and CRIS architectures.”

Arch (2)

- Interfațarea cu bootloader-ul, BIOS-ul
- Accesul hardware pentru: controlerele de întrerupere, controler SMP, controlere BUS-uri, setup trap-uri (handlere întreruperi, excepții, apeluri de sistem)
- Acces hardware pentru memoria virtuală
- Optimizări specifice arhitecturii pentru funcții de lucru pe şiruri, copieri, etc.

Device drivere



- Unified device model
- Reduce duplicarea codului
- Posibilitatea enumerării device-urile
- Crearea unui arbore de dependență
- Legături între drivere şi device-uri

Device drivere (2)

- Modularitate, multe clase ce oferă infrastructură pentru clase specifice de device drivere
  - Character device drivers: TTY device drivers, serial device drivere
  - Block device drivers: SCI device drivers
  - Filesysteme device drivers
  - Network device drivers
  - USB device drivers
  - Frame buffer drivers
  - 3D acceleration device drivers

Core kernel

- Process management
- Memory management
- Block I/O
- VFS – Virtual Filesystem Subsystem
- Stiva de rețea
- Securitate: LSM, SeLinux

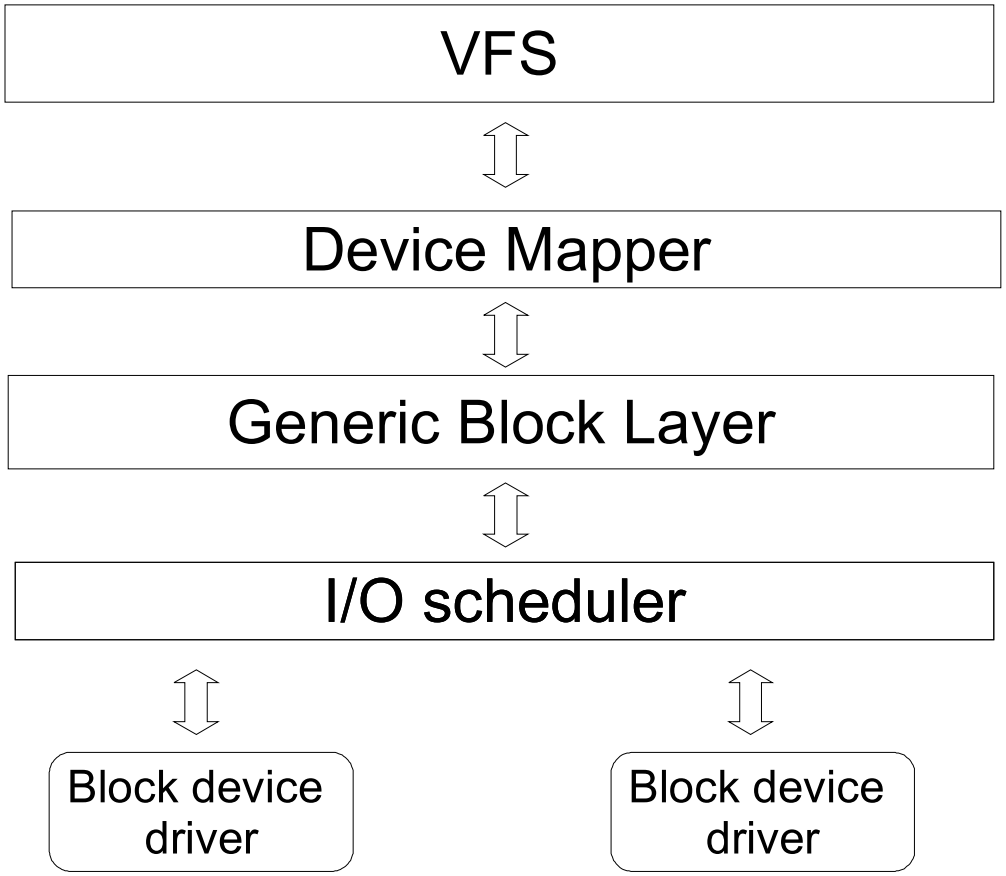
Process management

- Domenii de execuție
- Scheduling
- Procesele şi thread-urile sunt abstractizate în task-uri
  - Task-urile conțin resurse
  - Thread-urile = task-uri ce partajează aceleaşi resurse

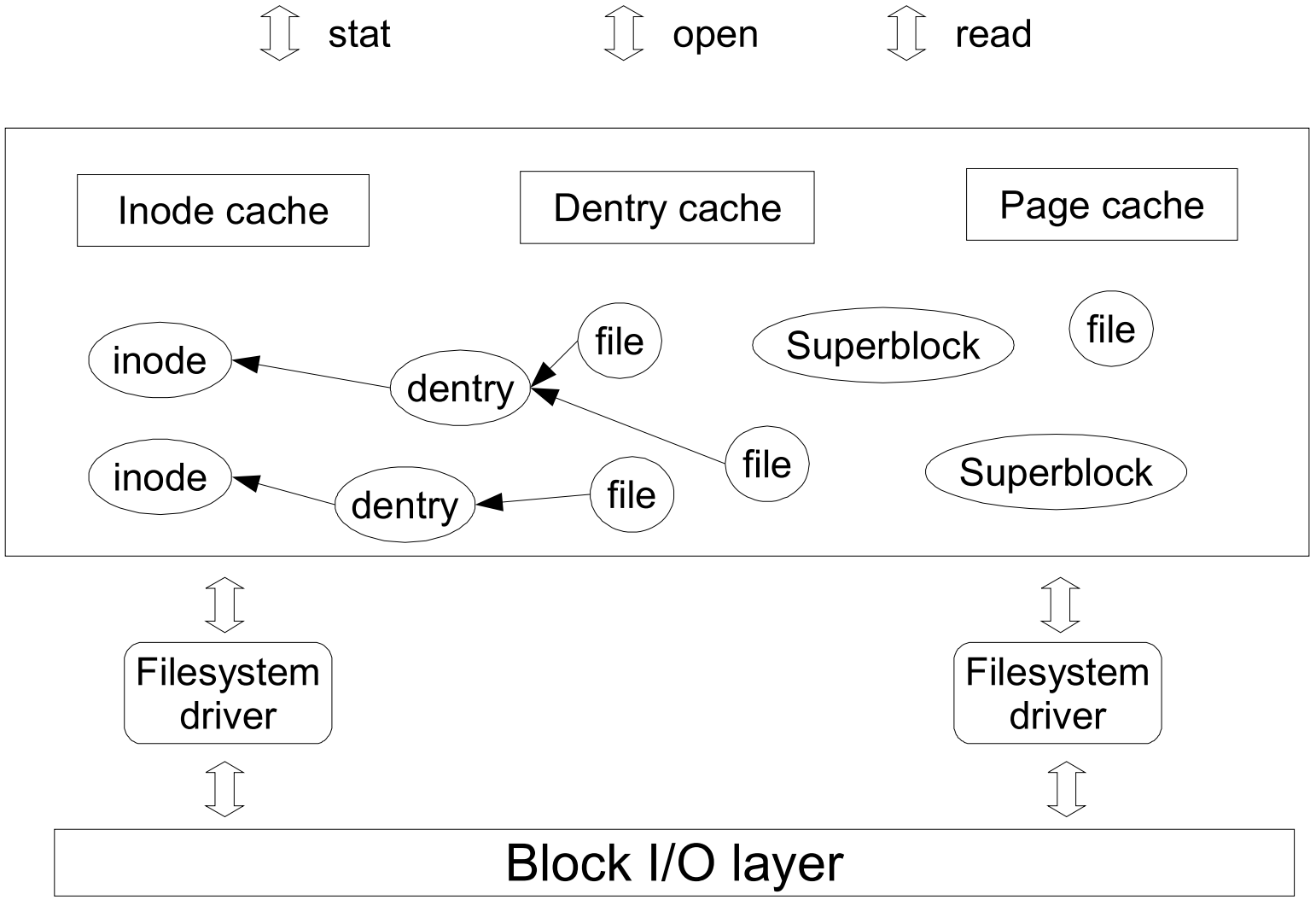
Memory management

- Gestiunea memoriei fizice: alocarea memoriei fizice
- Gestiunea memoriei virtuale: paginare, swapping, demand paging, copy on write
- Servicii pentru user-space: spațiul de adresă proces, mmap(), brk(), memorie partajată
- Servicii pentru kernel-space: slab, vmalloc

Block I/O management



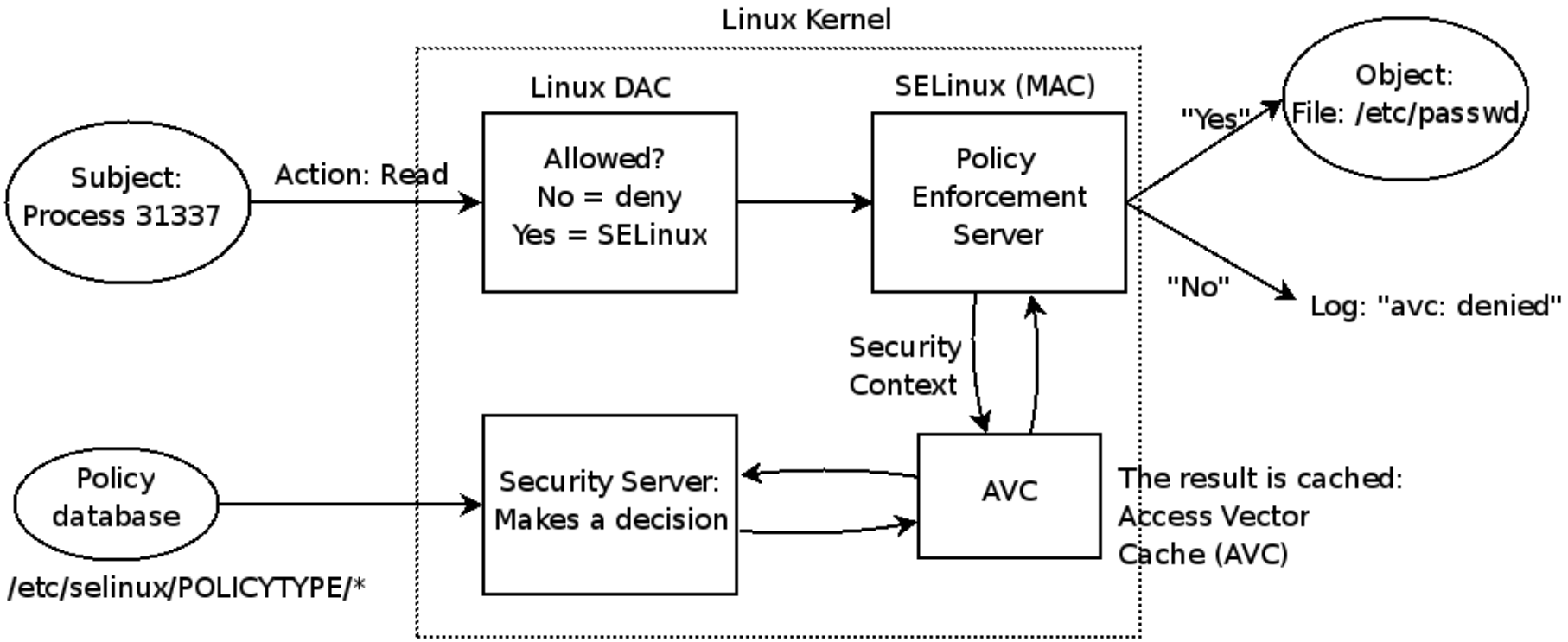
VFS


stat
open
read
Inode cache
Dentry cache
Page cache
inode
dentry
file
Superblock
file
inode
dentry
file
file
Superblock
Filesystem driver
Filesystem driver
Block I/O layer

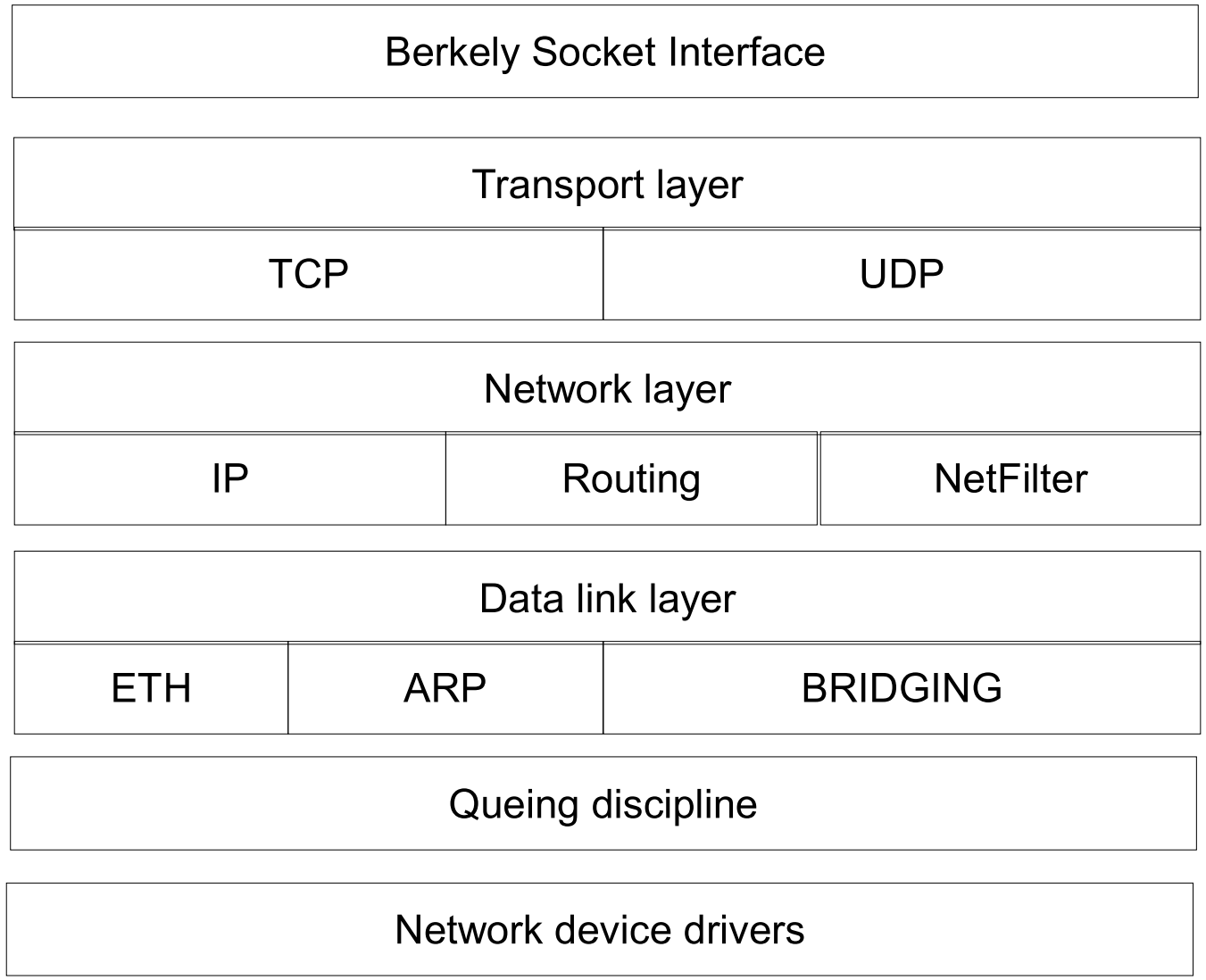
LSM

- Linux Security Modules
- Hook-uri pentru module de securitate
- Security Enhancened Linux - SELinux – proiect NSA, extinde modelul clasic de securitate UNIX
- AppArmor – proiect similar dezolvatat de Novell, bazat pe LSM

Security Enhanced Linux



Stiva de rețea



Fişiere sistem de bază

- vmlinux – imaginea kernelului obținută după compilare
- Vmlinuz – imaginea de kernel compresată şi stripuită; este încărcată şi executată de către bootloader
- Initrd – Initial Ram Disk – conține driverele necesare montării sistemului de fişiere root (compilate ca module)

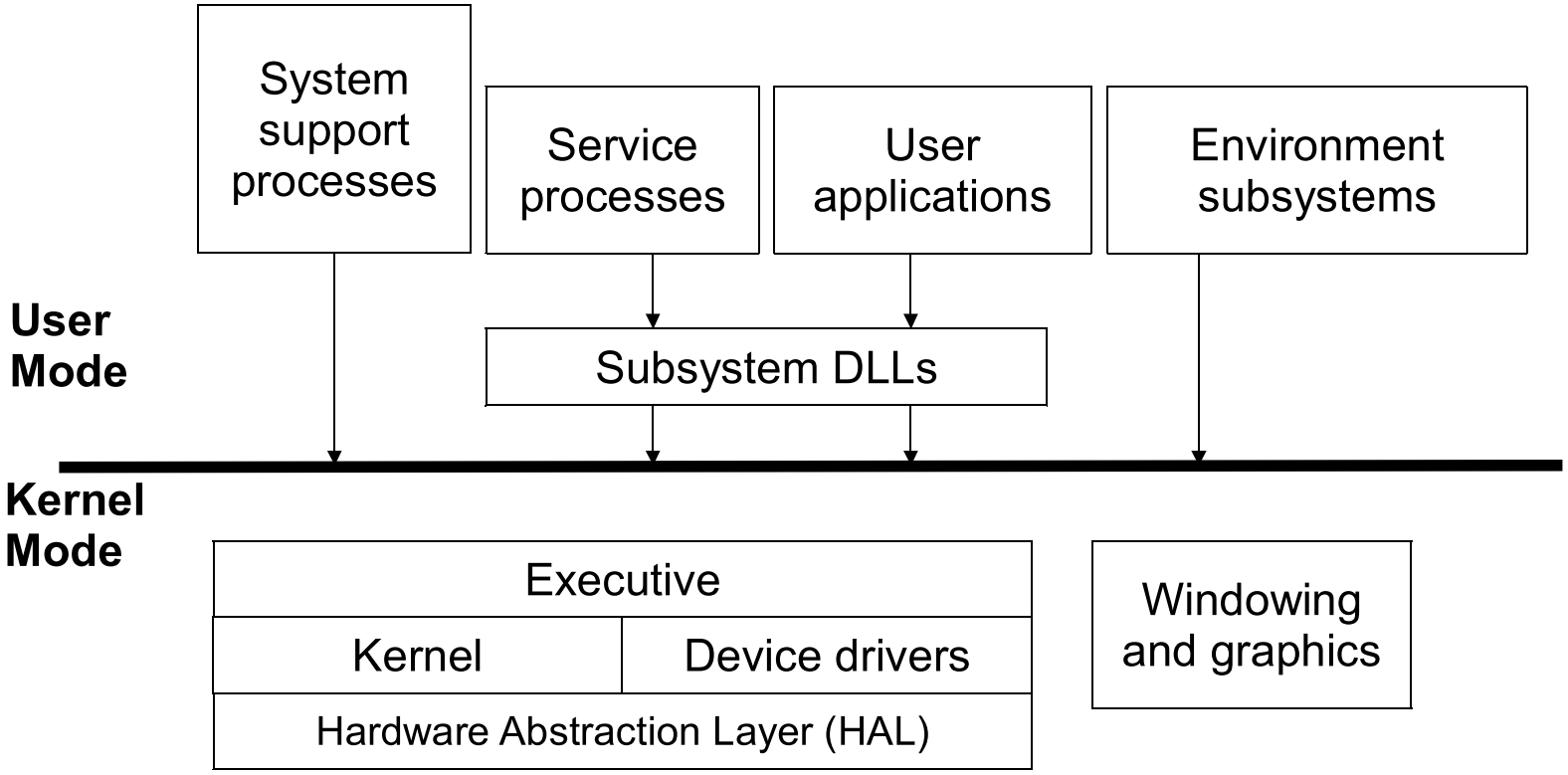
Procese sistem

- Swapper
  - PID 0
  - Contorizează timpul nefolosit (idle)
- Init
  - PID 1
  - În contextul acestui proces rulează kernel thread-urile
    - Primul proces rulat de kernel; rulează scripturile de inițializare a sistemului /etc/rc.d/, gestionează nivelele de rulare
    - Nu poate fi terminat
    - Înfiază procesele orfane

Kernel thread-uri

- Migration
- Ksoftirqd
- Watchdog
- Events
- Khelper
- Kblockd
- Pdflush
- Kswapd
- Aio
- Kjournald
- Kondemand
- ...

Arhitectura W2K



HAL

- Hardware Abstraction Layer
- Modul kernel
- Izolează sistemul de operare de diferențe hardware pe aceeaşi arhitectură
- Hardware-ul nu se accesează direct ci indirectat prin HAL
  - Accesul la I/O (porturi)
  - Controlerul de întreruperi

Device Drivere

- Module kernel (.sys)
- Tipuri de device drivere
  - Hardware device drivere
    - Ex: ethernet, PCI, mouse, etc
  - File system drivers
    - Traduc cereri I/O la nivel fişier în cereri I/O la nivel bloc
  - File system filter drivers
    - Sunt interpuse între file system drivers şi gestiunea I/O la nivel bloc
    - Criptare, RAID software

Device Drivere (2)

- Tipuri de device drivere
  - Redirectoare de rețea
    - Tipuri de drivere file system care traduc operațiile de I/O la nivel fişier în pachete destinate unei alte maşini
  - Protocol drivers
    - Implementează protocoale de rețea (TCP/IP, NetBEUI, IPX/SPX)

Device Drivere (3)

- Imbunatatiri aduse în W2K
  - Suport pentru Plug&Play şi Power Management
  - Device driverele sunt clasificate în 3 mari clase (WDM)
    - Bus driver
    - Function driver
    - Filter driver

Kernel-ul

- Funcții implementate în NTOSKRNL.EXE; sunt folosite de executiv
  - Independete de arhitectură
    - Sincronizare
    - Planificare
  - Dependente de arhitectură
    - Suport întreruperi şi excepții
- Implementat în C si ASM
- Implementează mecanismele necesare SO, dar nu ia decizii de policy (cu execepția planificării)

Obiecte Kernel

- Fiecare entitate manipulată în kernel este reprezentată de un obiect (o structură care adună la un loc informațiile despre acea entitate)
  - Obiecte de control
    - APC, DPC, obiecte întreruperi
  - Obiecte dispecer (sincronizare şi planificare)
    - Thread-uri kernel
    - Mutex-uri, semafoare
    - Timere

Executivul

- Funcții implementate în NTOSKRNL.EXE
  - Exportate şi apelabile din user-space (system services = apeluri de sistem)
  - Exportate şi apelabile din kernel-space (folosite de device drivere)
- Foloseşte serviciile (funcțiile) puse la dispoziție de kernel
- Aici se implementează politica dorită

Obiecte executiv

- Încapsulează (conțin pointeri către) obiecte kernel
- Conțin şi alte informații necesare pentru a servi cererile utilizatorului (handler-uri pentru manipularea obiectelor)
- Implementarea politicii necesită de asemenea informații si operații în plus: informații/verificări de securitate

Componente ale executivului

- Managerul de configurare
  - Implementarea si gestiunea registry-ului
- Managerul de thread-uri şi procese
  - Crearea si terminarea thread-urilor
- Managerul de securitate
- Mangerul operațiilor de I/E
  - Transferă cererile de I/E către device driverele de tip bloc
  - Interactionează cu Cache Managerul

Componente ale executivului (2)

- Power Managerul
- Cache Managerul
- Managerul de memorie virtuală

Componente speciale ale executivului

- Managerul de obiecte
  - Crează, gestionează si distruge obiectele executiv
- Funcții LCP (Local Procedure Call)
  - Execuție de proceduri inter-proces
  - Versiune optimizată a RPC
- Biblioteca run-time
  - Operatii cu şiruri, matematice, etc.
- Rutine de suport
  - Alocator de memorie (paged şi non-paged)

Subsisteme de mediu

- Prezintă utilizatorului mai multe vederi prin care acesta să acceseze funcțiile sistem
- În W2K există trei subsisteme
  - OS/2
  - POSIX
  - Win32
- Win32 este necesar, OS/2 si POSIX sunt opționale şi sunt încarcate doar la cerere
- Fiecare subsistem are asociat un proces

Subsistemul Win32

- Procesul asociat subsistemului este creat prin rularea csrss.exe
  - Gestiunea consolei
  - Crearea si terminarea proceselor şi thread-urilor
  - Părti din maşina virtuală pentru aplicații DOS
- Un device driver (Win32k.sys)
  - Managerul de ferestre
  - GDI

Subsistemul Win32 (2)

- DLL-uri (kernel32.dll, user32.dll gdi32.dll advapi32.dll)
  - Face apeluri de sistem în kernel sau LPC-uri către csrss.exe

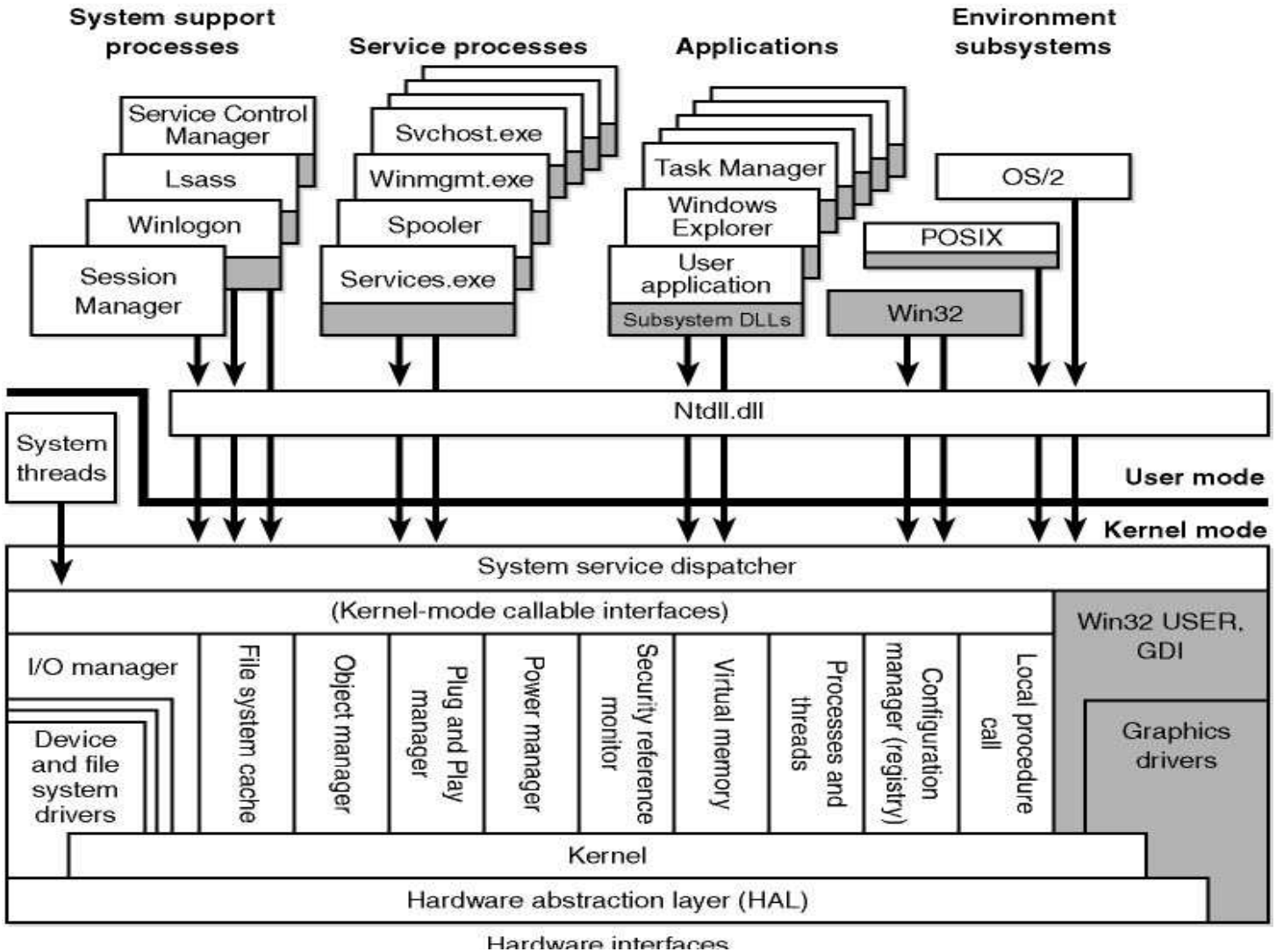
Subsistemul POSIX

- POSIX = “a Portable Operating System Interface based on uniX”
- Suport doar pentru POSIX 1003.1
  - Pentru că era inclus în lista necesară unui sistem pentru a fi folosit de guvern
  - Din acest motiv portarea aplicatiilor UNIX nu este comodă
- Procesul asociat subsistemului este creat prin rularea psxss.exe
- Aplicațiile POSIX sunt rulate cu ajutorul posix.exe şi sunt legate cu psxdll.dll

Selectarea subsistemului

- Selectarea subsistemului se face automat de SO, examinând un câmp din executabil (creat de către linker)
- Subsistemele POSIX si OS/2 se incarcă dinamic, la prima invocare a unui executabil din acel subsistem
- Subsistemele sunt izolate între ele (un program POSIX nu poate face apeluri Win32)

Arhitectura W2K (detaliu)



Fisiere sistem de baza

Filename	Components
Ntoskrnl.exe	Executive and kernel
Ntkrnlpa.exe	Executive and kernel with support for Physical Address Extension (PAE), which allows addressing of up to 64 GB of physical memory
Hal.dll	Hardware abstraction layer
Win32k.sys	Kernel-mode part of the Win32 subsystem
Ntdll.dll	Internal support functions and system service dispatch stubs to executive functions
Kernel32.dll, Advapi32.dll, User32.dll, Gdi32.dll	Core Win32 subsystem DLLs

Procese sistem

- Idle process
  - PID 0
  - Cate unu pentru fiecare procesor
  - Nu are ca suport o imagine (un executabil)
  - Este folosit pentru a contoriza timpul cât procesorul este liber
- System process
  - PID 8
  - Este folosit pentru a rula kernel thread-uri: memory managerul le foloseşte pentru a scrie paginile murdare pe disc, cache manager-ul pentru a implementa read ahead şi write-behind

Procese sistem (2)

- Session manager (Smss.exe)
  - Primul proces user-mode creat
  - Efectueaza operatii vitale pentru boot: crearea variabilelor de mediu, lansarea subsistemelor configurate să fie pornite şi a procesului de login
  - Daca csrss.exe sau winlogon.exe se termină se generează un crash
- Logon process (Winlogon.exe)
  - Se ocupă de login-urile si logout-urile (interactive)
  - După ce utilizatorul se autentifică (se apeleaza lsass.exe pentru verificare) se lansează userinit.exe care termină inițializarea sistemului şi ruleaza explorer.exe

Procese sistem (3)

- Logon process (Winlogon.exe)
  - Identificarea şi autentificarea sunt incapsulate într-o bibliotecă GINA cu interfața bine definită astfel încat poate fi inlocuită
- Local security authentication server (lsass.exe)
  - Este apelat de către winlogon, face verificările de rigoare şi generează un token de acces care va fi verificat de către executiv la operațiile făcute de proces
  - Token-ul de acces va fi folosit de winlong pentru a crea shell-ul (implicit explorer.exe)
  - Token-ul de access va fi apoi moştenit de fiecare process lansat din shell

Procese sistem (4)

- Service control manager
  - Procesul care porneşte, gestionează şi opreşte servicii sistem
  - Serviciile sunt programe cu o interfata bine definită prin care pot interacționa cu SCM-ul
  - SCM-ul citeste configurația serviciilor din registry
  - Tot SCM-ul încarcă şi device drivere

Intrebări

?