

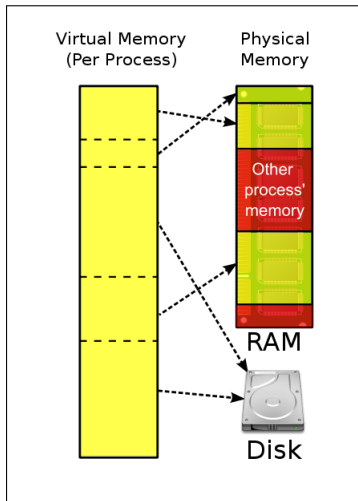
Laborator 6

Memoria virtuală

Sisteme de Operare

19 mai 2016

- ▶ Mecanism folosit implicit..
 - ▶ de către nucleul sistemului de operare pentru a implementa o politică eficientă de gestiune a memoriei
 - ▶ ce astfel de optimizări cunoașteți?

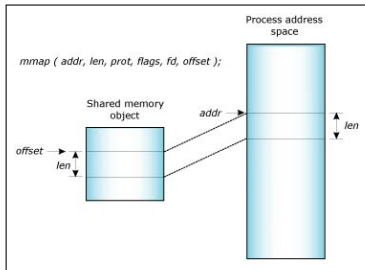


imagine preluată de pe wikipedia.org

- ▶ .. dar și explicit, pentru a mapa în spațiul de adresă al unui proces:
 - ▶ fișiere
 - ▶ memorie
 - ▶ dispozitive
- ▶ Mapare fișiere
 - ▶ memorie partajată
 - ▶ paginare la cerere
 - ▶ biblioteci partajate
- ▶ Mapare memorie
 - ▶ pentru alocarea unei cantități mari de memorie
- ▶ Mapare dispozitive
 - ▶ acces direct la memoria dispozitivului
 - ▶ când ar putea fi necesar?

- ▶ Accesare fişier similar cu un vector
- ▶ Mapările pot depăşi dimensiunea memoriei fizice
- ▶ Nu pot fi mapate dispozitive cu acces secvenţial (socket-uri, pipe-uri)
- ▶ Unitate: pagina (număr întreg, alinieri)
- ▶ Familia de funcţii mmap(2)

► mmap/munmap



- start poate fi NULL
- prot: PROT_READ, PROT_WRITE, PROT_EXEC, PROT_NONE
- flags: MAP_PRIVATE, MAP_SHARED, MAP_FIXED, MAP_LOCKED, MAP_ANONYMOUS (pt mapare memorie)
- mapare memorie: ignoră fd şi offset
- msync - sincronizare explicită fişier cu maparea din memorie

- ▶ CreateFileMapping/OpenFileMapping
 - ▶ primeşte HANDLE fişier
 - ▶ tip mapare: PAGE_READONLY, PAGE_READWRITE, PAGE_WRITECOPY
- ▶ MapViewOfFile
 - ▶ primeşte HANDLE FileMapping
 - ▶ mod acces: FILE_MAP_READ, FILE_MAP_WRITE, FILE_MAP_COPY
- ▶ UnmapMapViewOfFile

- ▶ VirtualAlloc/VirtualAllocEx
 - ▶ tip operație: MEM_RESERVE, MEM_COMMIT, MEM_RESET
 - ▶ start poate fi NULL; multiplu de 4KB pentru alocare și 64KB pentru rezervare

- ▶ VirtualFree/VirtualFreeEx
 - ▶ tip operație: MEM_DECOMMIT, MEM_RELEASE

- ▶ Interogarea zonelor mapate VirtualQuery/VirtualQueryEx
 - ▶ adresa de start a zonei, protecție, dimensiune
 - ▶ struct _MEMORY_BASIC_INFORMATION

- ▶ Accese la memorie nonconforme cu drepturile
 - ▶ Linux - generează semnale SIGBUS, SIGSEGV
 - ▶ sigaction, siginfo_t
 - ▶ Windows - generează excepții
 - ▶ AddVectoredExceptionHandler, VectoredHandler
- ▶ Linux - mprotect
 - ▶ acces: PROT_READ, PROT_WRITE, PROT_EXEC, PROT_NONE
 - ▶ adresa multiplu de dimensiunea unei pagini
- ▶ Windows - VirtualProtect/VirtualProtectEx
 - ▶ pt regiuni alocate cu VirtualAlloc/VirtualAllocEx folosind MEM_RESERVE

- ▶ Utilă pentru procese care trebuie să execute anumite acțiuni la momente de timp bine determinate
- ▶ Nu se va mai face swapout - ulterioare nu mai produc page fault
- ▶ Linux
 - ▶ mlock
 - ▶ mlockall
 - ▶ flags: MCL_CURRENT, MCL_FUTURE
 - ▶ munlock/munlockall
- ▶ Windows
 - ▶ VirtualLock/VirtualLockEx
 - ▶ rezultat: TRUE succes, FALSE altfel
 - ▶ VirtualUnlock/VirtualUnlockEx