Gestionarea I/E

Apelurile de sistem I/O in Linux si apelurile API de I/E pentru Windows

Herea Cristian

431 A

1. Linux

Apeluri de sistem (system calls)

Sistemele de operare au un nivel suplimentar de interfete oferit proceselor din spatiul utilizatorului prin care se realizeaza comunicare cu componentele calculatorului (hard-disk, procesor, dispozitive audio etc). Cateva dintre avantajele acestui nivel: protejeaza programatorul de limbaje primitive, securitatea sistemului este sporita iar programele sunt portabile (acestea functioneaza chiar daca anumite componente sunt schimbate).

Aceste interfete dintre utiliztori si hardware sunt implementate in Linux prin apeluri de sistem.

POSIX

Acesta este un standard in care se definesc API-urile ("Application Programming Interface") compatibile cu unix-ul.

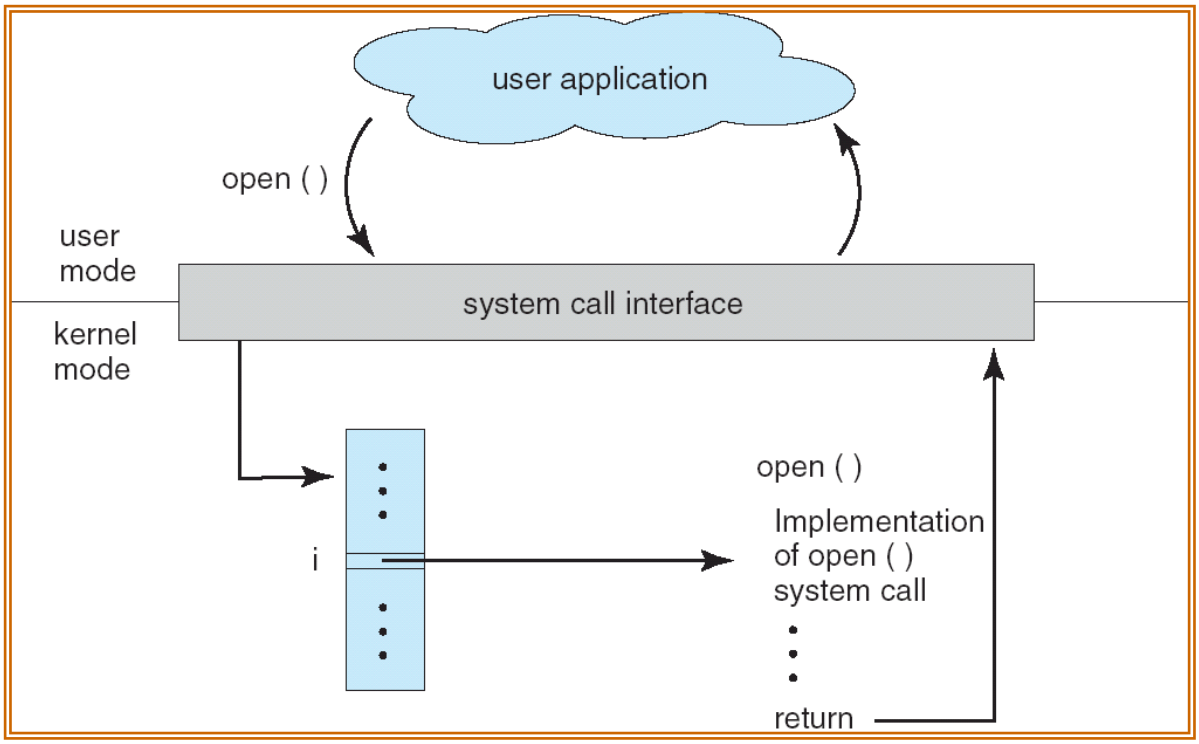
Cerfiticarea POSIX se acorda unui sistem de operare daca acesta ofera un set de api pentru aplicatii, implementarea fiind fara importanta.

Linux-ul este un sistem de operare compatibil POSIX.

Pentru programator nu este nici o diferenta intre API si apeluri de sistem, el urmarind doar functia, parametrii ei si rezultatul cautat. In schimb cei care construiesc kernel stiu ca API-urile sunt librarii care se afla in spatiul utilizatorului pe cand apelurile de sistem se afla in kernel.

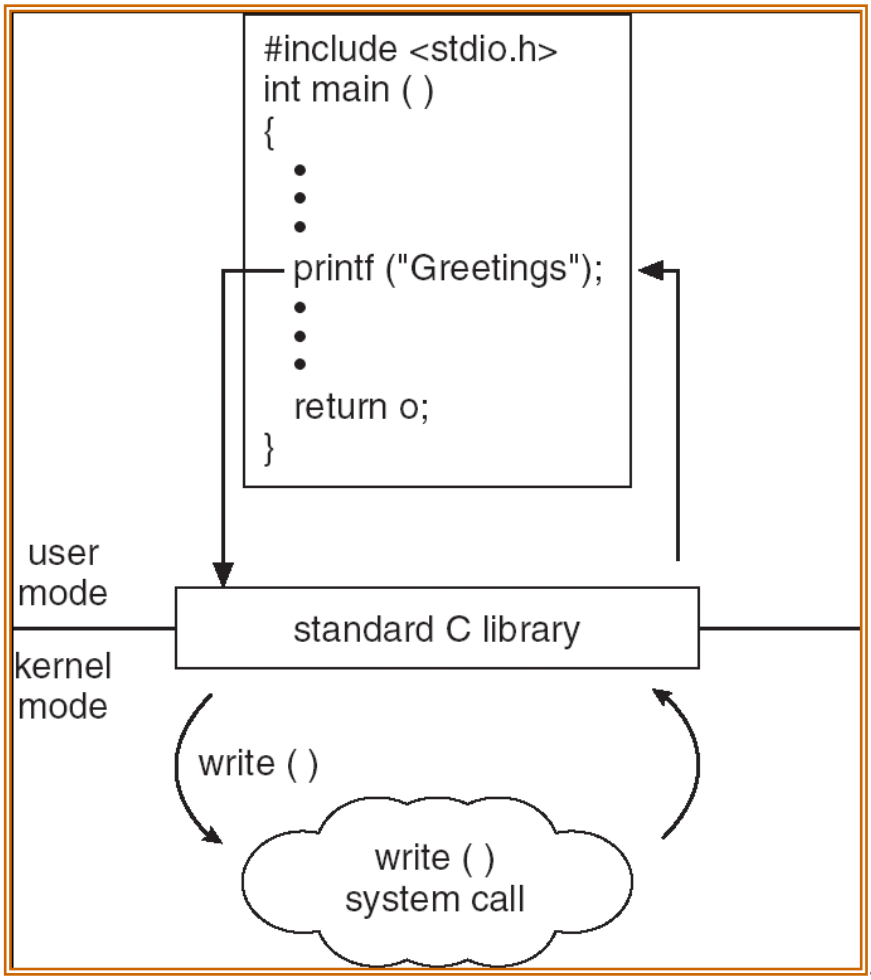
Deosebirea dintre API si apeluri de sistem este ca API-urile reprezinta functii care care permit obtinerea unui serviciu iar apelul este cere explicit

kernel-ului un anumit lucru prin intreruperi software.

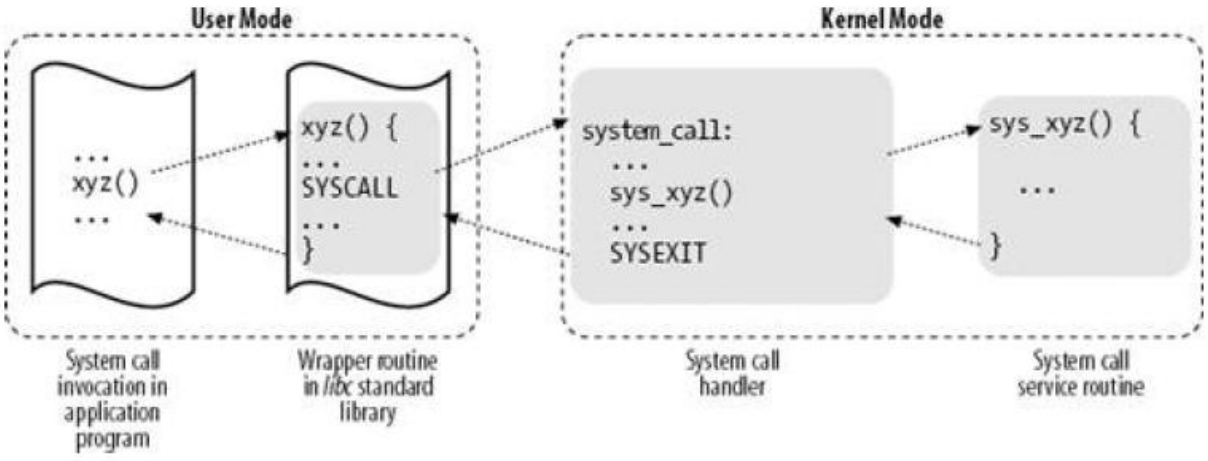


Relatia API – Apel de sistem [6]

In sistemele unix exista si librarii de functii API in ajutorul programatorilor. Cateva API-uri pe care le gasim in libraria C libc se refera la *rutine wrapper* [3] (singurul scop al acestor rutine este de a efectua un apel de sistem). In general fiecare apel de sistem are o rutina API corespondenta.



Program C care invoca apelul din librarie “printf()”, care la randul lui apeleaza apelul de system “write()”. [6]



[3]

In figura de mai sus observam legatura in o aplicatie care foloseste un apel de sistem prin routina sa *wrapper* si diferenta in spatiul utilizatorului (user mode) si spatiul kernel-ului. Sagetile indica modul in care se realizeaza procesele.

In general rutinele *wrapper* returneaza o valoare intreaga. Daca aceasta este egala cu -1 se indica faptul ca kernel-ul nu a putut executa cererea respectiva.

Apeluri de sistem I/O

Exista doua tipuri de I/O (In/Out):

a. Sincrone – fiecare cerere asteapta ca cererea anterioara sa incheie actiunea cu descriptorul de fisier iar programul asteapta terminarea cererii pentru a executa urmatoarea instructiune
b. Asincrone – sistemul de operare gestioneaza cererea iar programul poate executa urmatoarea instructiune

I/O sincrone

Exista cateva apeluri de sistem principale in linux pentru I/O sincrone:

- *Open* – deschide descriptorul de fisier.
- *Lseek* – seteaza descriptorul de fisier la pozitia byte-ului specificat.
- Read – citeste si copiaza datele din descriptorul de fisier intr-un buffer.
- Pread – citeste si copiaza datele din descriptorul de fisier intr-un buffer de la o locatie specifica in fisier.

I/O asincrone

Apeluri de sistem pentru I/O asincrone

- Aio_read – cerere de citire asincrona.
- Aio_write – cerere de scriere asincrona.
- List_listio – este un apel special care permite efectuarea unor mai multe citiri si scrieri intr-un singur apel.

2. Apeluri API. WINDOWS

Apeluri API I/E

Manevrarea dispozitivelor de intrare/iesire se face printr-un framework sau spatiu de lucru. Windows-ul are un spatiu de lucru creat in asa fel incat sa suporte cu usurinta dispozitive I/E de toate felurile: de intrare (tastatura, mouse, microfon), de iesire (monitor, imprimanta, CD/DVD writere), de stocare (hard-disk, SSD).

Categorii Win32 API

Unul dintre cele mai importante grupuri de apeluri API poate fi cel al sistemului grafic.

In acest grupt exista diferite apeluri prin care se pot crea, modifica si sterge ferestre. Ferestrele la randul lor au o sumedenie de stiluri si optiuni precum culori, marimi setate de utilizator, meniuri si bare de meniuri samd..

Exista foarte multe functii pentru grafica, de la modificarea unui pixel pana desenarea unor figuri geometrice.

Exemple de grupuri API:

- Text
- Managementul ferestrelor
- Casute de dialog
- Pictare si desenare
- Meniuri
- Paleta de culori
- Clipboard

Doarece majoritatea display-urilor nu pot afisa toate cele 2^{24} culori si folosesc doar 256 sau 65536 de culori este necesara existenta unei palete de culori pentru a determina cate culori sunt pot fi disponibile pentru sistem (256 sau 65536 etc). Apelurile din acest grup au grija de aceste lucruri prin crearea si distrugerea paletelor de culori, prin selectarea culorii de redae potrivite pentru o anumita culoare.

Afisarii textului ii este necesar un alt grup de apeluri. In acest grup apelul de afisare *TextOut* nu este foarte complicat. In schimb partea complicata se refera la culoarea textului, la marimea sa, la latimea caracterului etc. Acest lucru se poate face prin conversia la Bitmap. Bitmap-urile sunt de fapt blocuri rectangulare de dimensiuni mici care pot fi afisate pe ecran prin apelul Win32 BitBlt.

In continuare vor fi prezentate cateva dintre cele mai importante functii Win32API pentru I/E al fisierelor si echivalentele lor din Unix.

- *CreateFile (Unix: open)* - creaza sau deschide un fisier
- *DeleteFile (Unix*: *unlink*) - distruge un fisier
- *CloseHandle (Unix: close)* - inchide un fisier
- *ReadFile (Unix: read*) - citeste continutul unui fisier
- *WriteFile (Unix: write)* - scrie date intr-un fisier
- *SetFilePointer (Unix: lseek)* - seteaza pointerul intr-un fisier pe o anumita pozitie
- *GetFileAtribute (Unix: stat)* - returneaza proprietatile fisierului
- *LockFile (Unix: fcntl) - blocheaza o anumite parte a fisierului*

Bibliografie

1. S. Tanenbaum - Sisteme de operare moderne

2.www.wikipedia.com

3.Understanding the Linux Kernel, 3rd Edition , By Daniel P. Bovet, Marco Cesati

4.CS360 Lecture notes -- Introduction to System Calls (I/O System Calls)
by Jim Plank

5.Operating Systems System Calls and I/O by Henry Newman

6.Operating systems concepts – Silberschatz,Galvin and Gagne)