Sisteme de operare

25 iunie 2009

Timp de lucru: 70 de minute

NOTĂ: toate răspunsurile trebuie **justificate**

1. "Sistemele de operare moderne nu au probleme de fragmentare externă a memoriei fizice alocate din user-space." Indicați și motivați valoarea de adevăr a propoziției anterioare.

2. Un sistem de operare dispune de un planificator de procese care folosește o cuantă de 100ms. Durata unei schimbări de context este 1ms. Este posibil ca planificatorul să petreacă jumătate din timp în schimbări de context? Motivați.

3. Dați exemplu de funcție care este reentrantă, dar nu este thread-safe. Dați exemplu de funcție care este thread-safe, dar nu este reentrantă.

4. Într-un sistem de fișiere FAT un fișier ocupă 5 blocuri: 10, 59, 169, 598, 1078. Știind că:

- un bloc ocupă 1024 de octeți
- o intrare în tabela FAT ocupă 32 de biți
- tabela FAT NU se găsește în memorie
- copierea unui bloc în memorie durează 1ms

cât timp va dura copierea completă a fișierului în memorie?

5. Două procese P1, respectiv P2 ale aceluiași utilizator sunt planificate după cum urmează:

`fd = open("/tmp/a.txt", O_CREAT | O_RDWR, 0644);`
`write(fd, "P1", 2);`
`--- schedule ---`	

	`--- schedule ---`
	`fd = open("/tmp/a.txt", O_CREAT | O_RDWR, 0644);`
	`write(fd, "P2", 2);`

Ce va conține, în final, fișierul /tmp/a.txt? Ce va conține fișierul în cazul în care se folosesc thread-uri în loc de procese?

6. Are sens folosirea unui sistem de protejare a stivei (stack smashing protection, canary value) pe un sistem care dispune de și folosește bitul NX?

7. Pe un sistem quad-core și 4GB RAM rulează un proces care planifică 3 thread-uri executând următoarele funcții:

```
thread1_func(initial_data)
{
    for (i = 0; i < 100; i++) {
        work_on_data();
        wake_thread2();
        wait_for_data_from_thread3();
    }
}
```

```
thread2_func()
{
    for (i = 0; i < 100; i++) {
        wait_for_data_from_thread1();
        work_on_data();
        wake_thread3();
    }
}
```

```
thread3_func()
{
    for (i = 0; i < 100; i++) {
        wait_for_data_from_thread2();
        work_on_data();
        wake_thread1();
    }
}
```

Care este dezavantajul acestei abordări? Propuneți o alternativă.

8. În spațiul de adrese al unui proces, zona de cod (*text*) este mapată read-only. Acest lucru este avantajos din punct de vedere al securității, întrucât împiedică suprascrierea codului executat. Ce alt avantaj important oferă?

9. Folosind o soluție de virtualizare, se dorește simularea unei rețele formată din: două sisteme Windows, un gateway/firewall OpenBSD și un server Linux. Opțiunile sunt VMware Workstation, OpenVZ și Xen. Care variantă de virtualizare permite rularea unui număr cât mai mare de instanțe de astfel de rețele pe un sistem dat?

10. Un sistem de operare dat poate fi configurat să folosească un split user/kernel 2GB/2GB al spațiului de adresă al unui proces sau un split 3GB/1GB. Sistemul fizic dispune de 1GB RAM. Un proces rulează secvența:

```
for (i = 0; i < N; i++)
    malloc(1024*1024);
```

Pentru ce valori (aproximative) ale lui N *malloc* va întoarce NULL în cele două cazuri de split?

11. Un proces dispune de o tabelă de descriptori de fișiere cu 1024 de intrări. În codul său, procesul deschide un număr mare de fișiere folosind *open*. Totuși, al 1010-lea apel *open* se întoarce cu eroare, iar *errno* are valoarea *EMFILE* (maximum number of files open). Care este o posibilă explicație?

În conformitate cu ghidul de etică al Catedrei de Calculatoare, declar că nu am copiat la această lucrare. De asemenea, nu am ajutat și nu voi ajuta pe nimeni să copieze la această lucrare.

Nume:..... Grupă: .....

Semnătură: .....