Nume și grupă:

Sisteme de Operare

10 septembrie 2015

Timp de lucru: 90 de minute

Notă: Toate răspunsurile trebuie justificate

1. (**7 puncte**) Explicați de ce apelul `printf` nu este un mod robust de a face debugging unui program care primește semnalul `SIGSEGV`.

2. (**7 puncte**) Dați un exemplu in care planificatorul de procese *Shortest Job First* rezultă într-o alocare inechitabilă a resurselor procesorului.

3. (**7 puncte**) Numiți un avantaj și un dezavantaj al folosirii memoriei virtuale.

4. (**7 puncte**) Un atacator descoperă o vulnerabilitate într-un program care-i permite să execute cod arbitrar. De ce sunt interesante pentru atacator apelurile `read` și `write`?

5. (**7 puncte**) Dați două avantaje ale implementării unui server de web cu mai multe procese față de mai multe thread-uri.

6. (**10 puncte**) Explicați de ce este necesară starea *zombie* pentru un proces în sistemele Unix.

7. (**10 puncte**) Un sistem Unix are spațiu de adrese pe 32 biți și spatiu de swap de 40GB. Stiva unui thread are minim 4KB pe acest sistem.

a. Estimați numărul maxim de thread-uri ce pot fi create de un proces pe acest sistem.
b. Estimați numărul maxim de procese ce pot fi create pe acest sistem, presupunând că nu există limitări la numărul de descriptori sau identificatori de proces.

8. (**10 puncte**) Un proces este blocat in apelul de sistem `recvmsg`. Un pachet este primit la placa de rețea. Descrieți secvența de pași pe care îi execută placa de rețea și sistemul de operare până când apelul `recvmsg` se va întoarce și aplicația își va continua execuția.

9. (**10 puncte**) Explicați cum știe sistemul de operare cărui proces îi este destinat un segment TCP primit. Diferentiați între segmentele cu bit-ul de `SYN` activat sau dezactivat.

10. (**10 puncte**) Un programator folosește apelul de sistem `write` pentru a actualiza un fișier. După o pană de curent, programatorul descoperă ca unele informații au fost scrise cu `write` dar nu apar în fișier. Explicați cauza problemelor și dați o posibilă soluție.

11. (**25 puncte**) Vi se cere să implementați două programe, un sender și un receiver, care sunt capabile să transmită date între două calculatoare folosind două interfețe de rețea de 10Gbps. Datele de transmis sunt în memoria sender-ului, și se presupune că sunt infinite (e.g. dacă se ajunge la sfârșitului bufferului în care sunt stocate se va transmite de la începutul bufferului). Aplicația receiver are nevoie să stocheze datele în memorie în aceeași ordine în care au fost citite de aplicația receiver. Vi se cere să folosiți API-ul de sockets și protocolul TCP pentru a utiliza cele două interfețe disponibile la maxim (throughput total 20Gbps):

a. Detaliați protocolul pe care îl va folosi aplicația pentru a împărștia datele pe cele două căi și a le pune în ordine la receiver. (**7 puncte**)
b. Descrieți apelurile de sistem necesare pentru stabilirea conexiunii pe mai multe interfețe. (**3 puncte**)

c. Descrieți in pseudocod implementarea pentru sender, atunci când conexiunea este deja deschisă. Aveți în vedere folosirea corectă a operației send. (5 puncte)
d. Descrieți in pseudocod implementarea pentru receiver, atunci când conexiunea este deja deschisă. Aveți în vedere folosirea corectă a operației receive. (**5 puncte**)
e. Analizați soluția propusă, discutând minim 2 posibile probleme de performanță care pot apărea în practică, și oferind soluții de remediere. (**5 puncte**)

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

Nume și grupă:

Semnătură:.....