Facultatea de Automatica si Calculatoare, Universitatea Politehnica din Bucuresti

Computer Science & Engineering Department

Examen Final SPRC

Student: GLODEANU DAN **Grupa:** 342C5

Descriere curs:	**SPRC, An IV, C3, Semestrul I**
Titlu curs:	**Sisteme de Programe pentru Retele de Calculatoare**
Profesor:	**Prof.Dr.Ing. Florin POP**
Data examenului:	05.02.2022.
Durata examenului:	90 min
Tip Examen:	**"Closed Book"**
Numar pagini:	4 (patru)

Rezultate Examen	
Subiect	Punctaj
1	/2
2	/2
3	/2
4	/2
5	/2
6	/1
Σ	/11

Subiecte

2 puncte **1.** Prezentați cum se poate realiza propagarea actualizărilor pentru un depozit de date în două tipuri diferite de sisteme: a) un sistem bazat pe un server central cu n replici și b) un sistem Peer-to-Peer (1.5p). Discutați comparativ eficiența celor două metode propuse (0.5p).

2 puncte **2.** Explicați dacă este posibil ca o memorie să fie în același timp și secvențial și cauzal și FIFO consistentă (0.8p)? Dați un exemplu de utilizare pentru fiecare caz în parte (secvențial, cauzal, FIFIO) (1p).

2 puncte **3.** Explicați de ce modelul de protocoale bazate pe cvorum are nevoie de două condiții de bază (1p)? Cu notația folosită la curs, explicați ce implicație are înlocuirea condiției $N_W > N/2$ cu $N_W \geq N/2$ (1p).

2 puncte **4.** Descrieți pe scurt modelul Harrison-Ruzzo-Ullman (HRU). Ce rezolvă HRU și ce lasa nerezolvat? (1p) Discutați afirmația: dacă un subiect este proprietarul său, atunci poate acorda accesul (doar operații de *read* și *write*) altor subiecți la sine (1p).

2 puncte **5.** Descrieți conceptele de scalabilitate și elasticitate în mediile Cloud (1p). Descrieți două situații în care una din proprietăți să fie preferate celeilalte (1p).

1 puncte **6.** Putem proiecta un sistem de poștă electronică folosind RBAC? Justificați. (*bonus*).

Glodeanu Dan 342 C5

1) a) Sistem bazat pe un server central cu n replici

- Pentru a realiza un update la un astfel de sistem poate fi folosit Gossiping. Acesta este un algoritm epidimiologic, care presupune că un nod alege aleator potențiel recv. (destinații) cărora să le trimită update-ul. Informația nouă e trimisă dacă destinația nu e aflat deja de update. Interesul în a trimite update-uri scade pe măsură ce tot mai mulți vecini știu de el.

b) Un sistem Peer-to-Peer

- Sistem descentralizat Hibrid. Există o entitate centrală Server care stochează metadate despre fișierele stocate de clienți. Atunci când un client vrea să trimită un update altui Client:
  1. Cere de la server datele pentru destinație sa
  2. Serverul își verifică tabele și îi trimite datele de la client
  3. Clientu comunică cu celălalt client

Gossip poate fi eficient pentru sisteme de dimensiuni reduse, cu toate acestea el nu garantează actualizarea tuturor replicilor. Din acest motiv putem prefera un sistem Peer-to-Peer pentru a avea siguranța consistenței datelor.

Glodeanu Dan 342 C5

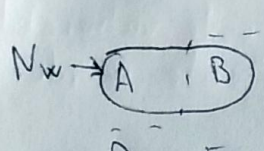
2) Secvențial - dacă operația x precede operația y în timp, atunci x este executat primul. Ex: read after write pe aceeași entitate.

Cauzal - dacă operația x precede cauzal operația y, x e executat primul indiferent de ordinea primirii. Ex: din cauza rețelei t_{i-1} a venit după t_i

FIFO consistent - tip de consistență care presupune că atunci când are loc o operație aceasta e trimisă imediat copiilor și după e executată următoarea operație

Dacă memoria e secvențială și cauzală ⇒ ea este și FIFO consistentă.

3) Protocoale Bazate pe Cvorum



1) $N_W + N_R > N$ - această condiție este necesară, deoarece presupune că există cel puțin o entitate actualizată

2) $N_W > \frac{N}{2}$ - asigură faptul că s-a executat cel puțin un write

Dacă $N_W > \frac{N}{2}$ e înlocuit cu $N_W \geq \frac{N}{2}$ pot apărea probleme la write after write.

Glodeanu Dan 342 C5

4) Harrison - Ruzzo - Ullman (HRU)

Caracteristici: S set subiecti
O set obiecte
R set drepturi
Matricea M cu intrarea M_{so} -> drepturile subiectului s asupra obiectului o

Acesta are câteva operații de bază:

- Enter right
- Delete right
- Create subject
- Delete subject
- Create object
- Delete object

Modelul asigură faptul că subiecții pot accesa obiectele pentru care au drepturi. Acesta este vulnerabil la scurgerea drepturilor, adică proprietarul poate acorda drepturile de read / write.

5) Scalabilitatea - în cloud computing scalabilitatea e infinită, datorită modelului client - server clientul poate cere oricâtă putere de calcul de la server.

Elasticitatea - acest aspect marchează posibilitatea clientului de a-și "realoca" resursele. În esență, el poate să își ajusteze puterea de calcul alocată, ca aceasta să permită ajustarea la nevoile clientului.

Preferabil:

Scalabilitatea - când avem o aplicație activă mereu sau îndelungat și știm exact de câte resurse avem nevoie pentru a maximiza eficiența

Elasticitatea - de exemplu avem un joc căruia îi crește online-ul în anumite perioade (vacanță) și scade în altele (septembrie) ne dorim să putem elibera resursele dacă nu avem nevoie de ele.

Gladcenu Dan 342 C5

6) RBAC Role Based Access Controle

Acest model presupune că avem utilizatori care au anumite roluri și au drepturi (asociate rolului) în cadrul sesiunii.

De. putem proiecte un siste de poștă electronici cu ajutorul RBAC. Fiecare client va avea un cont asociat, ier pentru a folosi platforma va fi necesară autentificarea utilizatorului. După autentificare utilizatorul obtine derpturile:

* Să citească mesajele primite
* Să șteargă mesajele primite
* Să trimită mesaje