CONTROLUL ACCESULUI

1. Ce este un sistem de protect¸ie peste o mult¸ime de drepturi? (concepte de baza ce trebuiesc definite: operatii primitive, comanda, sistem de protectie).

Un sistem de protectie peste o multime de drepturi (R) este o multime finita de comenzi peste R.
O stare finita a unui sistem de protectie este Q = (S,O,A), unde A = matrice de S linii si O coloane, A(s,o) inclus in R.

Operatii primitive:
- enter r into (Xs, Xo), adaugare de drepturi ale lui Xs asupra lui Xo, unde Xs apartine lui S (multimea de subiecti), si Xo apartine lui O (multime de obiecte).
- delete r from (Xs, Xo), stergere de drepturi.
- create / destroy Xs, adaugare / stergere de subiecti.
- create / destroy Xo, adaugare / stergere de obiecte.

Comanda:
command alfa (X1, X2, ... Xn), unde alfa reprezinta numele comenzii si X1, ... Xn sunt parametrii formali
if r1 in (Xs1,Xo1) and
...
rm in (Xsm, Xom), m >= 0
then op1
...
opn, n >= 1
end

2. In ce consta problema sigurantei sistemelor de protectie? (concepte de baza ce trebuiesc definite: stare (matrice de acces), transformare de stare, scurgere de drepturi, problema sigurantei).

Problema sigurantei pentru sistemele de protectie consta in a decide daca o stare Q apartinand lui C de mana (sistem de protectie), este sigura pentru un drept r apartinand lui R.
O stare finita a unui sistem de protectie este Q = (S,O,A), unde A = matrice de S linii si O coloane, A(s,o) inclus in R.
Transformare de stare:
Fiind date o operatie op si o substitutie sigma, definim tranzitia intre starea Q = (S, O, A) si starea Q**'** = (S', O', A') prin relatia binara „=>sigma(op)", daca si numai daca este valabila una din urmatoarele proprietati:

1. daca op = enter r into (X, Y), atunci sigma(X) apartine S, sigma (Y) apartine O, S'=S, O'=O, si

A’(s,o) = * A(s,o) reunit {r}, daca ((sigma(X),sigma(Y)) = (s,o)
* A(sigma(X),sigma(Y)), altfel

2. daca op = sterge r din (X,Y), atunci sigma(X) apartine S, sgiam (Y) apartine O, S’= S, O’ = O, si

A’ (s,o) = * A(s,o) – {r}, daca (sigma(X), sigma(Y)) = (s,o)
* A(sigma(X), sigma(Y)), altfel

3. daca op = create subject X, atunci sigma (X) nu apartine lui O, S’ = S reunit {sigma (X)}, O’ = O reunit {sigma (X)}, si

A’ (s, o) = * A(s,o), daca (s,o) apartine S x O

* multimea vida, altfel

4. daca op = create object Y, atunci sigma (Y) nu apartine O, S’ = S, O’ = O reunit {sigma (Y)}, si

A’(s,o) = * A(s,o), daca (s,o) apartine S x O
* multimea vida, altfel

5. daca op= destroy subject X, atunci sigma (X) apartine S, S’ = S – {sigma(X)}, O’ = O – {sigma (X)}, si A’ (s,o) = A(s,o), pentru orice (s,o) apartine S’ x O’;

6. daca op = destroy object Y, atunci sigma (Y) apartine O – S, S’ = S, O’ = O – {sigma(X)}, si A’ (s,o) = A(s,o), pentru orice (s,o) apartine S’ x O’.

Definim: (S,O,A) =>op(S’,O’, A’) ⇔ exista sigma : (S,O,A) => sigma(op)(S’,O’,A’).

Fiind date o comanda alfa si o substitutie sigma, definim tranzitia intre starea Q = (S, O, A) si starea Q**’** = (S’, O’, A’) prin relatia binara „=>sigma(alfa)”, daca si numai daca este valabila una din urmatoarele proprietati:

1. daca sigma(alfa) nu este satisfacut in (S,O,A), atunci (S’,O’,A’) = (S,O,A);
2. daca sigma(alfa) este satisfacut in (S,O,A), atunci exista Q0,Q1,...Qn asa incat:
   (S,O,A) = Q0 => sigma(op1)Q1 => sigma(op2) .... => sigma(opn)Qn = (S’, O’, A’).
   Unde op1, ... opn este corpul lui alfa.
   Definim:
   (S,O,A) => alfa (S’,O’,A’) ⇔ exista sigma : (S,O,A) => sigma(alfa) (S’,O’,A’) si
   (S,O,A) => (S’, O’,A’) ⇔ exista alfa : (S,O,A) => alfa(S’,O’,A’).

   Scurgerea de drepturi:
   C, R, Q, r apartine R, alfa (o comanda) apartine C;
   Alfa scurge dreptul r din Q, daca exista o subtitutie sigma astfel incat:
   1. testul comenzii sigma(alfa) este indeplinit la starea Q
   2. exista o secventa de stari Q0, ... Qi, a.i. :
      * Q = Q0 => sigma(op1) Q1 = (S1,O1,A1) ... => sigma(opi)Qi = (Si,Oi,Ai).
      * r apartine lui Ai(s,o) - Ai-1(s,o).

      C scurge dreptul r din starea Q daca exista o comanda in C care scurge r din Q.

3. Ce cunoasteti despre dificultatea rezolvarii algoritmice a problemei sigurantei sistemelor de protectie?

Problema sigurantei pentru sistemele de protectie este nedecidabila, in cazul in care folosim operatii de creare in comenzi.
Demonstratia se bazeaza pe o reducere de la prbl opririi pentru masinile Turing si foloseste comenzi cu operatii de creare. Daca nu am folosi operatii de creare, prbl ar fi PSPACE – completa. Daca folosim operatii de creare in comenzi, dar C este mono`operational (fiecare comanda are exact o operatie in corp), atunci prbl sigurantei e decidabila.
Daca permitem comenzi arbitrare, dar numai un nr finit de subiecti, putem face legatura intre prbl acoperirii pentru sisteme de adunare de vectori si prbl sigurantei. Va rezulta ca prbl sigurantei la sistemele de protectie cu un nr finit de subiecti este decidabila.
Un sistem de protectie este monotonic daca comenzile lui nu contin operatii de delete si destroy.
Prbl sigurantei la sistemele monotonice este nedecidabila.

4. Ce este o lista de control al accesului? Discutati cazul Unix si Windows NT.

O lista de control al accesului (ACL), este o coloana din matricea de control al accesului.
Dezavantaje:
* ceruta unde userii isi administreaza singuri securitatea fisierelor lor.
* mai putin ceruta unde:
    * nr userilor e mare si in continua schimbare;
    * userii vor sa`si stabileasca singuri autoritatea de a executa un prg catre un alt user pt un timp.

Avantaje:
* usor de implementat
* verificarea securitatii la Run`Time e dificila;

ACL in UNIX:
- oricare fisier si folder au asociate permisiuni de acces (read / write / execute), definite pt useri (owner al fisierului, grupul ownerului, restul).
- fiecare permisiune are 2 valori: allow / deny, specificate de un bit.

Ex: -rw-r----- , primul bit specifica ca ACL e pt un fisier, urmatorii 3: drepturile de acces ale ownerului, urmatorii 3: drepturile de acces pt grup, iar ultimii 3: drepturi de acces pt restul userilor.
Daca primul bit = `d`, atunci obiectul in cauza este un folder.

ACL pt un prg:
UNIX nu ofera o metoda directa pt asta, dar exista 2 atribute:
* suid (set user ID)
* sgid (set group ID)

ACL in Windows NT:

- 6 tipuri de permisiuni (Read / Write / Execute / Delete / Change permission / Take Ownership)
- permisiunile au 3 valori : Access denied / Access allowed / System audit si sun definite pt useri si grupuri.
- ACL`urile sunt asociate fisierelor si directoarelor, iar fiecare ACL e o lista de intrari de forma (User / Group, Permissions).

5. **Ce este o lista de capacitati?**

Lista de capacitati este un rand din matricea de control al accesului asociata unui subiect. Stocam o pereche (o, r) numita capacitate.
Fiecare capacitate reprezinta un tichet pt s, sa acceseze o(obiect), cu permisiunea r.
Capacitatile sunt tag`uri de autentificare, folosite in EROS (extremely reliable operating system).

6. **Ce intelegeti prin acces discretionar si acces mandatar?**

Accesul directionar (DAC) restrictioneaza accesul la obiecte bazandu`se pe identitatea subiectului (user`ul sau grupul de care apartine)
Subiectii pot decide singuri cine le poate accesa resursele si cu ce autoritate.

Accesul mandatar (MAC) restrictioneaza accesul la obiecte pe baza nivelului de senzitivitate al informatiilor pe care obiectul le contine si autorizarea subiectului de a accesa informatii cu acel nivel de senzitivitate.

Subiectii nu pot controla sau neglija accesul; adminul defineste nivelul de senzitivitate prin etichete de securitate si controleaza accesul userilor, specificand pt fiecare ce eticheta de securitate poate folosi.

POLITICI DE SECURITATE

1. In ce consta politica de confidentialitate Bell-LaPadula? (concepte de baza ce trebuiesc definite: latice a nivelelor de securitate, stare, regulile BLP, stare sigura)

Politica de confidentialitate Bell-LaPadula consta in definirea a trei reguli (regula de securitate simpla, proprietatea star si regula discretionara) si o latice a nivelelor de securitate.

Regula de securitate simpla: un subiect poate citi doar obiecte cu nivelul de securitate <= nivelul curent al subiectului.
Proprietatea star: un subiect poate scrie doar obiecte cu nivel de securitate >= nivelul curent al subiectului
Regula discretionara: creatorul unui obiect are drept de control al accesului altor subiecti asupra obiectului respectiv.

O stare este un 3-uplu (b,M,f), unde b = accese curente posibile, M = matricea de permisiuni si f=un 3-uplu (fs, fc, fo): fs =nivelul maxim de securitate al subiectului, fc=nivelul curent de securitate al subiectului, fo = nivelul de securitate al obiectului.

O stare este sigura daca cele trei reguli sunt respectate.

Laticea nivelelor de securitate este o multime L partial ordonata, astfel incat oricare 2 elemente a,b apartinand lui L au o limita superioara a|b si o limita inferioara a&b. Pentru doua obiecte la nivelurile a si b va exista un nivel minim de securitate a|b pentru a le accesa pe amandoua. Pentru doi subiecti la nivelurile a si b va exista un nivel maxim de securitate a&b pentru un obiect, pentru ca acesta sa poata fi citit de amandoi.

2. In ce consta politica de integritate Biba? (concepte de baza ce trebuiesc definite: latice a nivelelor de integritate, regulile Biba)

Politica de integritate Biba stabileste un set de reguli pentru accesul subiectilor la obiecte:

1) Proprietatea de integritate simpla: x=r => i(s)<=i(o) – un subiect nu poate citi obiecte cu nivelul de integritate mai mic decat al subiectului
2) Proprietatea no write-up: x=w => i(s)>=i(o) – un subiect nu poate scrie obiecte cu nivelul de integritate mai mare decat al subiectului
3) Invocation property: x=e => i(s)>=i(o) – un subiect nu poate executa programe cu nivelul de integritate mai mare decat al subiectului

Laticea nivelelor de securitate este o multime L partial ordonata, astfel incat oricare 2 elemente a,b apartinand lui L au o limita superioara a|b si o limita inferioara a&b. Pentru doua obiecte la nivelurile a si b va exista un nivel minim de securitate a|b pentru a le accesa pe amandoua.

Stabilirea cheii

1. **Descrieti si explicati rolul protocolului Shamir fara cheie.**

In protocolul Shamir fara cheie emitatorul si receptorul mesajului nu schimba chei de criptare, insa au chei private pentru criptarea si decriptarea mesajelor. Algoritmul Shamir foloseste exponentierea modulo p (unde p este un numar prim mare) pentru functiile de criptare. Astfel, $E(e,m) = m^e \bmod p$ si $D(d,m) = m^d \bmod p$. Exponentul e va fi ales de la 1 la p-1, astfel incat gcd(e,p-1) = 1. Exponentul d, corespunzator decriptarii, va fi ales astfel incat $de \equiv 1 \pmod{p-1}$. Potrivit micii teoreme a lui Fermat, $D(d,E(e,m)) = m^{de} \bmod p = m$.

Protocolul Shamir are proprietatea dezirabila de comutativitate, deoarece $E(a,E(b,m)) = m^{ab} \bmod p = m^{ba} \bmod p = E(b,E(a,m))$.

2. **Descrieti si explicati rolul protocolului Needham-Schroeder cu chei partajate.**

   Protocolul doreste stabilirea unei chei de sesiune intre doi clienti ai unei retele, de obicei pentru a proteja comunicarea ulterioara.
   Fie A, B doi clienti si S un server. A doreste initierea comunicarii cu B. A trimite un mesaj la server identificandu-se pe el insusi si pe B, clientul cu care vrea sa comunice. Server-ul genereaza cheia K_{AB} si i-o trimite lui A.

De asemenea trimite mesajul {K_{AB}, A} criptat cu K_{BS}, impreuna cu un nonce N_A si numele lui B, pentru a il asigura pe A ca mesajul este nou si ca server-ul raspunde la cererea de initiere a conversatiei. Ca urmare a acestui mesaj, A trimite mesajul criptat cu K_{BS} catre B. B decripteaza cu cheia K_{BS} mesajul, rezultand cuplul {K_{AB}, A}.

B cripteaza nonce-ul sau primit de la server (N_B) cu cheia K_{AB} si trimite mesajul astfel criptat catre A, pentru a arata ca are cheia. A face o operatie simpla pe N_B (ex. scadere), cripteaza numarul obtinut cu K_{AB} si trimite mesajul criptat {(N_B)- 1}K_{AB} catre B. In urma decriptarii, B are certitudinea ca are o conversatie sigura cu A.

3. **Descrieti si explicati rolul protocolului Needham-Schroeder cu chei publice.**

Here, Alice (A) and Bob (B) use a trusted server (S) to distribute public keys on request. These keys are:

- K_{PA} and K_{SA}, respectively public and private halves of an encryption key-pair belonging to A
- K_{PB} and K_{SB}, similar belonging to B
- K_{PS} and K_{SS}, similar belonging to S. (Note this has the property that K_{SS} is used to *encrypt* and K_{PS} to *decrypt*).

$$A \rightarrow S : A, B$$

A requests B's public keys from S

$$S \rightarrow A : \{K_{PB}, B\}_{K_{SS}}$$

S responds with public key K_{PB} alongside B's identity, signed by the server for authentication purposes.

$$A \rightarrow B : \{N_A, A\}_{K_{PB}}$$

A invents N_A and sends it to B.

$$B \rightarrow S : B, A$$

B requests A's public keys.

$$S \rightarrow B : \{K_{PA}, A\}_{K_{SS}}$$

Server responds.

$$B \rightarrow A : \{N_A, N_B\}_{K_{PA}}$$

B invents N_B, and sends it to A along with N_A to prove ability to decrypt with K_{SB}.

$$A \rightarrow B : \{N_B\}_{K_{PB}}$$

A confirms N_B to B, to prove ability to decrypt with K_{SA}

4. **Descrieti si explicati rolul protocolului Bloom.**

a,b,c – secrete distincte
f(x,y) =a+b(x+y)+cxy mod p
fie U, ru; U=multimea utilizatorilor, ru= public
f(x1,ru) = a+(b+cru)x.
$g_{ru}(x)$ = au+bux
gru este calculat de admin;
au, bu secrete ptr U.
U-> V (U comunica cu V)
$g_{ru}(rv) = K_{uv} = g_{rv}(ru) = Kvu$
le >= p are solutie unica oricare ar fi l.

6. **Descrieti si explicati rolul schemei Shamir de partajare a secretelor.**

O schema da partajare a secretelor pleaca de la un secret si deriva din el mai multe secrete partiale, care vor fi distribuite intre un grup de useri. Anumite subgrupuri de useri vor putea sa recupereze secretul initial daca membrii unui astfel de subgrup isi pun laolalta secretele.

Input: n useri si un prag k, k<=n.

TA alege un secret S!=0, un numar prim p>max{S,n} si k-1 parametri distincti a1, a2,...,ak-1 apartinand Z*p

TA calculeaza functia polinomiala f(x) = SUM(i=0,k-1)(ai * xi) de gradul k-1, unde a0 = S. TA transfera secretul partial Si = f(i) userului i, 1<=i<=n

Recuperarea secretului: orice grup de k useri distincti care isi pun laolalta secretele partiale pot recalcula polinomialul f prin interpolarea Lagrange, apoi pot reface secretul stiind ca S = f(0).

Formula de interpolare Lagrange este: f(x) = SUM(i=1, k) f(i) * PROD(j=1, k; j!=i) (x-j)/(i-j).