Controllo dell’accesso

- Valuta le richieste di accesso alle risorse/informazioni da parte degli utenti che hanno “guadagnato” accesso al sistema e decide se permetterle o negarle.
- La correttezza del controllo dell’accesso presuppone
  - Corretta identificazione/autenticazione degli utenti
    $\Longrightarrow$ Nessuno dovrebbe poter acquisire i privilegi di un altro.
  - Correttezza delle informazioni e delle regole di controllo (che devono essere protette da modifiche improprie).
- Autenticazione à anche necessaria per stabilire le responsabilità (accountability).
- Ogni login dovrebbe corrispondere ad un solo utente
  $\Longrightarrow$ no account condivisi

Politiche e meccanismi

Nello studio del controllo dell’accesso è utile separare

- Politiche Definiscono linee guida ad alto livello e regole che descrivono gli accessi autorizzati al sistema.

  Le politiche definiscono regole generali che governano l’accesso (es., politica aperta vs politica chiusa)

  Spesso il termine politica è utilizzato anche per riferirsi alle specifiche autorizzazioni e restrizioni di accesso che devono essere implementate (es., “Impiegati possono leggere il bollettino del dipartimento”).
- Meccanismi Funzioni di basso livello (hardware-software) che implementano le politiche.

Politiche e meccanismi

La separazione di politiche e meccanismi permette di:

- discutere differenti requisiti di accesso indipendentemente dalla loro implementazione
- confrontare differenti politiche per il controllo degli accessi
- confrontare differenti meccanismi che implementano la stessa politica
- sviluppare meccanismi in grado di implementare più politiche

Meccanismi per il controllo dell'accesso

Basati sulla definizione di un reference monitor che deve essere

- tamper-proof: non può essere alterato
- non-bypassabile: media tutte le richieste di accesso al sistema e alle sue risorse
- security kernel: deve essere confinato in una parte limitata del sistema (distribuire le funzioni di sicurezza in tutto il sistema implica che tutto il codice deve essere verificato)
- piccolo abbastanza da poter essere verificato con metodi rigorosi e formali

Meccanismi di sicurezza

L' implementatione di un meccanismo non è banale ed è complicata dalla necessità di evitare

- storage channel (problema del residuo) Elementi di memorizzazione quali pagine di memoria e settori di disco devono essere "ripuliti" prima di essere riallocati a un nuovo processo per prevenire "scavenging" dei dati.
- covert channel Canali che non sono intesi per trasferimento di informazioni (es., le risposte del sistema o gli effetti del carico del sistema sulla esecuzione di un programma – timing channel) che possono essere sfruttati per inferire informazione.

Assurance Come si comporta il meccanismo?

Sviluppo di controllo dell'accesso

Generalmente affrontato tramite un approccio multi-fase, dalle politiche ..... ai meccanismi

Passa attraverso la definizione di un

- Modello per il controllo degli accessi che definisce formalmente la specifica e la esecuzione del controllo dell'accesso.

  Il modello deve essere:

  - completo: Deve rappresentare tutti i requisiti di sicurezza che devono essere rappresentati.
  - consistente: Non deve contenere contraddizioni. Es., non può allo stesso tempo negare e autorizzare uno stesso accesso.

Progettazione di un sistema di sicurezza

Vantaggi di un approccio multifase:

- In ogni fase lo studio viene concentrato su aspetti particolari
- Permette di separare la valutazione di politiche, modelli e meccanismi
- Permette di dimostrare proprietà sul sistema.
  - Dimostrando che il modello è "sicuro" e che il meccanismo implementa correttamente il modello possiamo affermare che il sistema è "sicuro".

Attenzione Non dimostriamo sicurezza assoluta, ma sicurezza rispetto a come l'abbiamo definita!!!

Politiche di sicurezza

Le politiche di sicurezza possono essere distinte in politiche per:

- controllo dell'accesso: definiscono chi può (o non può) accedere alle risorse. Tre classi principali:
  - Politiche discrezionarie (DAC)
  - Politiche mandatorie (MAC)
  - Politiche basate sui ruoli (RBAC)
- amministrazione: definisce chi può specificare le autorizzazioni/regole che governano il controllo dell'accesso
  - Generalmente associata a DAC e RBAC

Politiche discrezionarie

- Controlla l'accesso sulla base dell'identità degli utenti che lo richiedono e su regole che stabiliscono chi può (o non può) eseguire azioni sulle risorse.
- Sono chiamate discrezionarie perchè agli utenti può essere data l'autorità di passare i propri privilegi ad altri utenti (la concessione e la revoca di privilegi è controllata da una politica amministrativa).

Modelli di sicurezza

Nello studio dei modelli di sicurezza valutiamo

- Entità del sistema
  - Soggetti che possono accedere agli oggetti
  - Oggetti che devono essere protetti
  - Azioni che possono essere eseguite sugli oggetti
- Stato di autorizzazione/protezione specifica gli accessi che sono permessi/negati
- Assiomi che devono essere soddisfatti

Modello a matrice di accesso

- Introdotto da Lampson (nel '71) nel contesto della protezione dei sistemi operativi. Esteso da Graham e Denning (nel '72).
- Formalizzato da Harrison, Ruzzo e Ullmann (nel '76). Spesso riferito come modello HRU
- È chiamato "a matrice di accesso" perchè lo stato di autorizzazione è rappresentato da una matrice
- Fornisce una struttura per descrivere i sistemi di protezione (molti sistemi successivi possono essere classificati come modello a matrice di accesso)

Modello a matrice di accesso – stato di protezione

Lo stato del sistema è definito da una tripla (S,O,A) dove

- S è l'insieme dei soggetti (che possono esercitare privilegi).
- O è l'insieme degli oggetti (sui quali possono essere esercitati privilegi). In alcuni casi i soggetti possono essere considerati oggetti, nel qual caso S$\subseteq$ O. Es., file, directory (in SO); relazioni, viste (in DB)
- A è la matrice di accesso, dove
  - righe corrispondono ai soggetti
  - colonne corrispondono agli oggetti
  - A[s,o] indica i privilegi del soggetto s sull'oggetto o

Lo stato può essere modificato tramite comandi che utilizzano le primitive:

enter r into $A[s,o]$, **delete** r from $A[s,o]$, **create subject** s', **destroy subject** s', **create object** o', **destroy object** o'

Access Matrix – Example

	File 1	File 2	File 3	Program 1
Ann	own read write	read write		execute
Bob	read		read write	
Carl		read		execute read

Implementazione del modello a matrice di accesso

La matrice è generalmente grande e sparsa. Memorizzare la matrice come array bi-dimensionale $\Rightarrow$ spreco di memoria

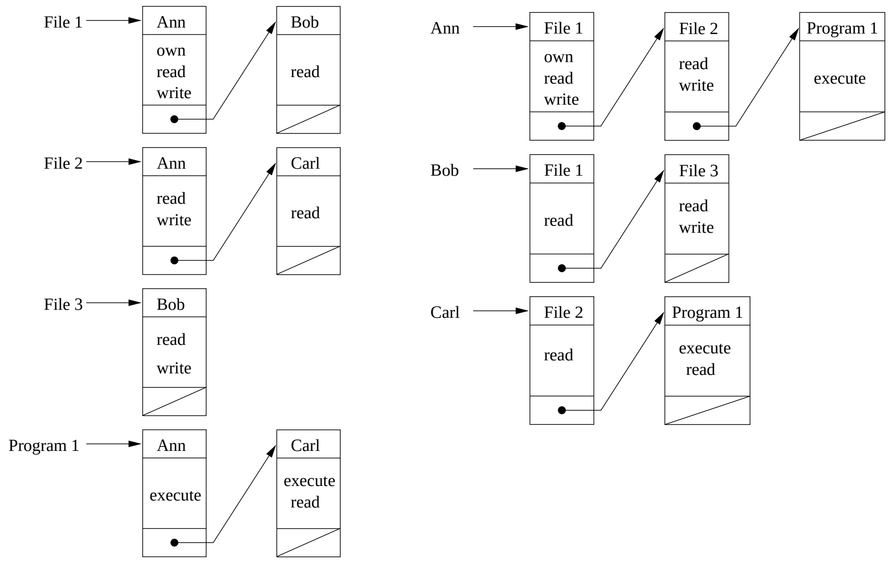
Approcci alternativi

- Tabella di autorizzazione Memorizza le triple (s,o,a) non nulle della matrice in una tabella con tre colonne. Generalmente utilizzato in DBMS.
- Access control list (ACLs) Memorizza per colonne.
- Capability list (ticket) Memorizza per righe.

Authorization Tables

User	Access mode	Object
Ann	own	File 1
Ann	read	File 1
Ann	write	File 1
Ann	read	File 2
Ann	write	File 2
Ann	execute	Program 1
Bob	read	File 1
Bob	read	File 2
Bob	write	File 2
Carl	read	File 2
Carl	execute	Program 1
Carl	read	Program 1

Access control lists vs. Capability Lists



ACL vs Capability

- ACL richiede di autenticare sempre i soggetti.
- Capability non richiede sempre autenticazione di oggetti, ma richiede *non falsificabilità* e controllo sulle propagazione di capability.
- ACL è più efficiente nel controllo dell'accesso e revoca relative a oggetti.
- Capability è più efficiente nel controllo dell'accesso e revoca relative a soggetti.
- Le operazioni per-oggetti sono considerate prevalenti $\Longrightarrow$ molti sistemi sono basati su ACL.
- Alcuni sistemi utilizzano una forma abbreviata di ACL (es., Unix 9 bits)

Comandi

I cambiamenti allo stato del sistema sono modellati tramite un insieme di comandi della forma

command $c(x_1, \ldots, x_k)$
  if r_1 in $A[x_{s_1}, x_{o_1}]$ and
    r_2 in $A[x_{s_2}, x_{o_2}]$ and
    ....
    $r_m$in $A[x_{s_m}, x_{o_m}]$
  then op_1
    op_2
    ....
    op_n
end

r_1, ..., r_m sono modi di accesso; s_1, ..., s_m e o_1, ..., o_m sono interi tra 1 e k. Se m=0, il comando non ha la parte condizionale.

Comandi – Esempi

command CREATE(subj,file)
create object file
enter Own *into* A[subj,file] **end.**

command CONFER$_{\text{read}}$(owner,friend,file)
if Own in A[owner,file]
then *enter* Read *into* A[friend,file] **end.**

command REVOKE$_{\text{read}}$(subj,exfriend,file)
if Own in A[subj,file]
then *delete* Read *from* A[exfriend,file] **end.**

Nota: Il modo di accesso non appare come parametro nei comandi $\Longrightarrow$ un comando diverso per ogni modo di accesso.

Trasferimento di privilegi

Autorizzazioni amministrative (che permettono trasferimento di privilegi) possono essere modellate associando `flag` ai modi di accesso. Es.,

- copy flag ($*$): il soggetto può concedere ad altri la autorizzazione

  command TRANSFER$_{\text{read}}$(subj,friend,file)
  if Read* in A[subj,file]
  then *enter* Read *into* A[friend,file] **end.**

- transfer-only flag (+): il soggetto può concedere ad altri la autorizzazione (e il flag su questa); ma così facendo la perde.

  command TRANSFER-ONLY$_{\text{read}}$(subj,friend,file)
  if Read+ in A[subj,file]
  then *delete* Read+ *from* A[subj,file]
  enter Read+ *from* A[friend,file] **end.**

Transizioni di stato

L'esecuzione di un comando $c(x_1, ..., x_k)$ sullo stato di sistema $Q = (S, O, A)$ causa la transizione allo stato Q' tale:

$$Q = Q_0 \vdash_{op_1^*} Q_1 \vdash_{op_2^*} ... \vdash_{op_n^*} Q_n = Q'$$

dove

- $op_1^* \ldots op_n^*$ sono operazioni primitive
- i parametri formali $(x_1, ..., x_k)$ nella definizione sono sostituiti dai parametri attuali con i quali la chiamata è effettuata

Se la parte condizionale del comando non è verificata, il comando non ha effetto e $Q = Q'$.

Problema della safety

Riguarda il problema della propagazione dei privilegi. Vuol rispondere alla domanda:

- Dato un sistema con configurazione iniziale Q_0 esiste una sequenza di richieste che eseguita su Q_0 produce uno stato Q' dove a appare in una cella $A[s, o]$ che non la aveva in Q_0?

Non tutti i flussi di privilegio sono indesiderati $\Longrightarrow$ soggetti fidati sono ignorati nella analisi.

- è un problema non decidibile (può essere ridotto al problema della terminazione di una macchina di Turing)
- è decidibile per comandi mono-operazionali (ogni comando può eseguire una sola operazione primitiva)
- decidibile quando i soggetti e gli oggetti sono finiti

Modello Take-Grant

Introdotto da Jones, Lipton e Snyder (nel '76) per studiare il problema della safety

- Considera una classe ristretta di sistemi
- Modella lo stato di protezione come un grafo
  - ogni grafo può essere rappresentato con la sua matrice di adiacenza $\Longrightarrow$ può essere interpretato come un modello a matrice di accesso
- Safety risulta polinomiale

Modello Take-Grant

Lo stato del sistema è definito da una tripla (S, O, G) dove

- S è l'insieme dei soggetti (che possono esercitare privilegi).
- O è l'insieme degli oggetti (sui quali possono essere esercitati privilegi).
- $G = (S \cup O, E)$ grafo di autorizzazione, dove
  - nodi sono soggetti e oggetti
  - archi autorizzazioni

  Modi di accesso: read, write, take, grant.
  - Take Un soggetto che ha diritto take su di un altro può prendere da questo <u>qualsiasi</u> privilegio
  - Grant Un soggetto che ha diritto grant su di un altro può dare a questo <u>qualsiasi</u> privilegio

Operazioni amministrative

- s **take** r for y from x.

  Il soggetto s prende il privilegio r su y dal soggetto x.

- s **grant** r for y to x.

  Il soggetto s garantisce il privilegio r su y al soggetto x.

Operazioni amministrative – 2

- s **create** R for **new** subject/object x.

  Il soggetto s crea un nuovo soggetto/oggetto x e prende privilegi R su questo.

- s **remove** r for x.

  Il soggetto s rimuove il privilegio r su x

Vantaggi modello Take-Grant

Il modello Take-Grant è stato introdotto nel contesto del problema della safety.

- Safety è decidibile (problemi di connettività del grafo)
  - $O(n)$ nel numero dei nodi rispetto a una specifica autorizzazione
  - $O(n^3)$ nel numero dei nodi se non si fissa il soggetto che può acquisire il privilegio)

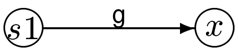
Però il contesto è differente da quello del modello a matrice di accesso.

Modello Take-Grant

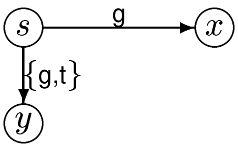
Svantaggi:

- Non selettività rispetto alle autorizzazioni al trasferimento: il take e grant si applicano a tutte le autorizzazioni di un soggetto.
- Non controllo sui diritti concessi: se viene dato a qualcuno un diritto 1) questo può darlo a tutti quelli su cui ha un grant, 2) tutti quelli che hanno un take su di lui possono prenderlo
- Reversibilità del flusso di trasporto dei privilegi: uno stato in cui ci può essere flusso di privilegi da x a y può essere trasformato in uno stato in cui può esistere flusso da y a x.

Reversibilità del flusso di privilegi



s1 **create** {g,t} for y



s1 **grant** g for y to x