12 - Protezione

Protezione

- **Protezione**: garantire che le **risorse** di un sistema di elaborazione siano accedute solo dai **soggetti** autorizzati.
- **Risorse**: fisiche e logiche (fisiche: CPU, memoria, stampanti; logiche: file, semafori, ...)
- **Soggetti**: utenti, processi, procedure.
- Servono metodologie, modelli, strumenti per la **specifica dei controlli** e la loro **realizzazione** (enforcement)
- **Obiettivo della protezione**: assicurare che ciascun componente di programma/processo/utente attivo in un sistema usi le risorse del sistema solo **in modi consistenti** con le **politiche** stabilite per il loro uso.
- **Separazione tra politiche e meccanismi**: un sistema di protezione deve essere iin grado di realizzare una varieta' di politiche

Protezione o Sicurezza?

Protezione e Sicurezza sono due termini vicini ma diversi.

- La **protezione** serve per prevenire errori o usi scorretti da parte di procesi/utenti che operano nel sistema
- La **sicurezza** serve per difendere un sistema dagli attacchi esterni

Sicurezza

Autenticazione

Verifica l'identita' dell'utente attraverso:

- Possesso di un oggetto (es., smart card)
- Conoscenza di un segreto (password)
- Caratteristica personale fisiologica (impronta digitale, venature retina)

Problema della mutua autenticazione

Si noti che **l'autorizzazioen** (protezione) serve per specificare le azioni concesse a ogni utente.

Autenticazione ≠ Autorizzazione

- **Riservatezza**: previene la lettura non autorizzata delle informazioni (es. messaggi cifrati. Se intercettati, non rivelano comunque il contenuto)
- **Integrità:** previene la modifica non autorizzata delle informazioni (Es. un messaggio spedito dal mittente e' ricevuto tale e quale dal destinatario).
- **Disponibilità**: garantire in qualunque momento la possibilita' di usare le risorse
- **Paternità**: chi esegue un'azione non puo' negarne la paternita' (per esempio un assegno firmato)

Protezione (e least privilege)

- In qualunque momento un processo (o un utente) puo' accedere solo agli oggetti per cui e' autorizzato
- Nell'informatica moderna e' importante rispettare il principio di "**least privilege"**, cioe' in ogni istante un processo deve accedere **solo a quelle risorse strettamente necessario** per compoiere la sua funzione (in questo odo di limita il danno che un processo con errori puo' creare nel sistema).

Esempi di **least privilege**:

1. Processo **P** chiama una procedura **A. A** deve poter accedere a sue variabili e parametri formali passate, e non a tutte le variabili del processo **P**.
2. Un amministratore di sistema che deve fare il backup di tutto un file system, deve avere i diritti di lettura su tutto, non quelli di scrittura

Protezione e controllo degli accessi

- Nei sistemi operativi ci sono dei componenti incaricati di verificare che i processi possano accedere alle sole risorse per cui sono autorizzati.
- Si parla di **Reference Monitor**, come il componente del sistema che media tra le richieste di accesso dei processi e le risorse
- TUTTE le richieste di accesso passano dal Reference Monitor
- E' importante che tutte le decisioni di accesso alle risorse siano concentrate in un unico componente

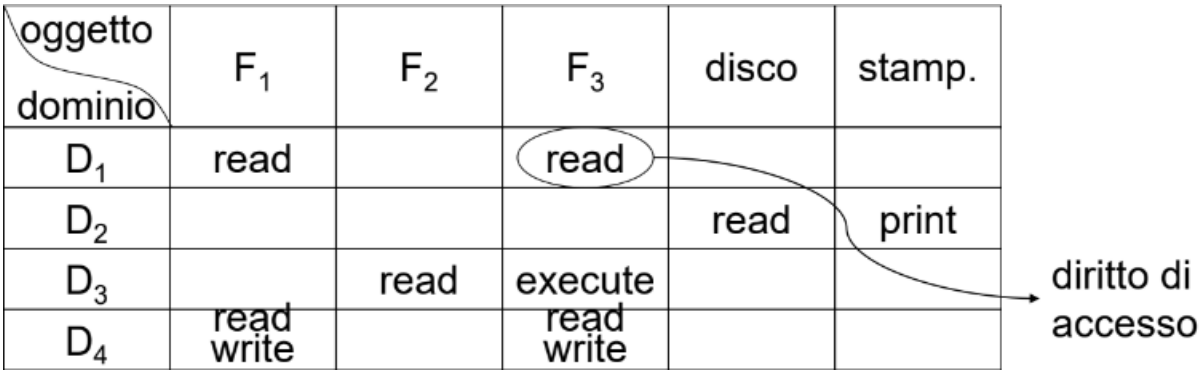
Dominio di protezione

- Quali sono le soluzioni per gestire il controllo degli accessi e garantire quindi uan corretta protezione delle risorse?
- Consideriamo il **dominio di protezione**, che definisce un insieme di risorse (oggetti) e i relativi tipi di operazione (diritti di accesso) sugli oggetti stessi che sono permesse a processi (soggetti) appartenenti a tale dominio
- Per esempio, un processo opera all'interno di un **dominio di protezione** che specifica le risorse che il processo puo' usare

Il dominio e' un concetto astratto che puo' essere realizzato in una varieta' di modi:

- un dominio per ogni **utente**. L'insieme degli oggetti che l'utente puo' accedere dipende dall'**identita' dell'utente**. Il cambio di dominio e' **legato dall'identita' dell'utente** (avviene quando cambia l'utente, es. Unix)
- un dominio per ogni **processo**. Ogni riga descrive oggetti e i diritti di accesso per un processo. Il cambio di dominio corrisponde **all'invio di un messaggio** a un altro processo
- un dominio per ogni **procedura**. Il cambio del dominio corrisponde **alla chiamata di procedura**

Matrice degli accessi (modello di protezione)



- **access(i,j)** definisce l'**insieme dei diritti di accesso** che un processo che opera nel dominio **i** puo' esercitare sull'oggetto **j**
- Si puo' realizzare come un **insieme ordinato** di triple <dominio, oggetto, insieme dei diritti> (tavola globale)
- Quando un'operazione **M** deve essere eseguita nel dominio $\mathbf{D_i}$ su $\mathbf{O_j}$, si cerca la tripla $<\mathbf{D_i}, \mathbf{O_j}, \mathbf{R_k}>$ con $\mathbf{M} \in \mathbf{R_k}$. Se esiste, l'operazione puo' essere eseguita; diversamente, si ha situazione di **errore **
- Domini **statici o dinamici**
  - caso statico: l'associazione processo/dominio non puo' cambiare durante la vita del processo
  - caso dinamico: c'e' uno "switch" per permettere a un processo di cambiare dominio di protezione
- Problemi della matrice degli accessi: dimensioni matrice troppo grandi (e sparse)
- Soluzioni meno generali ma piu' efficienti e diffuse: **access control list, capability**

Access Control List (ACL)

- **Per ogni oggetto** viene indicata la coppia ordinata <dominio, insieme dei diritti>
  limitatamente ai domini con un insieme di diritti non vuoto
- Quando **deve essere eseguita** un'operazione **M** su un oggetto $\mathbf{O_j}$ nel dominio $\mathbf{D_i}$, si cerca nella lista degli accessi
  $<\mathbf{D_i}, \mathbf{R_k}>$ con **M** e $\mathbf{R_k}$

- Se non esiste, si cerca in **una lista di "default"**; se non esiste, si ha condizione di **errore**
- Per motivi di **efficienza**, si puo' cercare prima nella lista di default e successivamente nella lista degli accessi
- Esempio: **file system**, lista degli accessi **associata al file** contiene: nome utente (il dominio) e diritti di accesso

Capability Lis

- Per ogni dominio viene indicato **l'insieme degli oggetti e dei relativi diritti di accesso** (capability list)

  D1: <O_1, diritti> , <O_2, diritti>, etc.

  D2: <O_2,diritti> , <O_5, diritti>, etc.

  etc.
- Spesso un oggetto e' identificato **dal suo nome fisico** o dal **suo indirizzo** (capability). Il possesso della capability corrisponde all'autorizzazione a eseguire una certa operazione
- Quando un processo opera in un dominio, **chiede di esercitare un diritto di accesso** su un oggetto. Se cio' e' consentito, il processo **entra in possesso di una capability** per l'oggetto e puo' eseguire l'operazione
- La lista delle capability non e' direttamente **accessibile** a un processo in esecuzione in quel dominio. E' **protetta** e **gestita** dal SO. Non puo' migrare in qualsiasi spazio direttamente accessibile a un processo utente (non puo' essere **manipolata** dai processi).