Sicurezza

Il sistema operativo è quello che tiene isolati i processi e gestisce l'hardware. Se vogliamo avere un **S.O.** sicuro dobbiamo costruirlo su una base sicura. Ma cosa si intende per sicuro? Dobbiamo tenere a mente tre concetti:

- **La segretezza,** possono accedere ai dati solo i rispettivi proprietari.
- **L'integrità**, un altro utente non può accedere a un contento del mio file neanche per sbaglio
- **La disponibilità,** il sistema deve essere sempre pronto, deve rispondere sempre.

Quando un processo fa una system call, il **kernel** deve valutare la richiesta che viene fatta. Innanzitutto controlla se è sensata, il numero deve corrispondere ad una system call e i parametri devono essere validi. Inoltre il **kernel** controlla se il processo ha il permesso di fare quella determinata cosa (dettata dalla system call). Ogni richiesta arriva da un soggetto contenente un oggetto (file o parte di memoria). Il sistema verifica se il soggetto ha i permessi di fare quella precisa cosa. Le regole cambiano da sistema operativo a sistema operativo.

Un utente non parla direttamente con il kernel, ma interagisce con il sistema e lancia i processi che faranno system call. Deve esserci un'associazione fra **utenti e processi,** e un modo per verificare l'identità degli utenti. Nei sistemi di autenticazione ci sono le cosiddette **AAA:**

- **Authentication:** verifica dell'identità mediante, per esempio, username e password
- **Authorization:** decide se accettare o rifiutare le richieste dell'utente
- **Accounting:** Tiene traccia di quello che fai

Identificazione e autenticazione sono argomenti molto vasti. Il caso più comune è che gli utenti si identificano con uno username e si autenticano con una password.

All'interno dei sistemi **unix** ci sono due file che contengono per ogni utente le sue informazioni; in particolare:

- **L'user id (UID):** numero che identifica l'utente
- **La password**

Abbiamo due file importanti: i **file passwd** che contengono le informazioni base di ogni utente. Non contiene però la password (anche se dal nome sembra) perché può essere letto da tutti gli utenti del sistema. **I file shadow** contengono le password ed è un file protetto.

UID0 corrisponde a **root**, che è amministratore del sistema; quindi, i processi che girano con **UID0** sono chiamati privilegiati.

Per l'autorizzazione ci sono due approcci:

- **Access Control List:**
    - Ogni oggetto contiene una lista di coppie soggetto/accesso

- **Capabilities:**

- Sono delle chiavi che racchiudono un oggetto in accesso e ci danno il loro privilegio

Indipendentemente dall'uso di ACL o capabilites chi decide chi può (o non) accedere ad una risorsa? Il **propietario (DAC)** o tramite un **controllo di accessi con delle regole (MAC)** usato in ambito militare.

Un principio importante è quello del **minimo privilegio:** l'idea è che in ogni momento, ciascuna identità dovrebbe avere i permessi minimi per eseguire i suoi compiti. Se un programma ha più permessi del necessario, vuol dire che un attaccante può fare più cose.

Ogni processo ha associato tre UID e quando facciamo Login coincidono:

- **Real UID:** proprietario del processo
- **Effective UID:** usato per determinare i permessi di accesso a risorse condivise
- **Saved UID:** Serve per salvare temporaneamente l'effective UID originale