

La Protezione nei Sistemi Operativi

Protezione e Sicurezza

Sicurezza: riguarda l'insieme delle **tecniche per regolamentare l'accesso** degli utenti al sistema di elaborazione.

La sicurezza impedisce accessi non autorizzati al sistema e i conseguenti tentativi dolosi di alterazione e distruzione dei dati.

Protezione: insieme di attività volte a garantire il controllo **dell'accesso** alle risorse logiche e fisiche da parte degli utenti autorizzati all'uso di un sistema di elaborazione .

Sicurezza

Le tecnologie di sicurezza di un sistema informatico realizzano meccanismi per l'**identificazione**, l'**autenticazione** e l'**autorizzazione** di utenti «fidati».

Protezione

Per ogni utente **identificato, autenticato e autorizzato** è **necessario stabilire**:

- quali siano le **risorse** alle quali può accedere
- con quali **operazioni** può accedervi

Questo è stabilito dal **sistema di protezione** tramite le tecniche di **controllo degli accessi**.

Protezione

In un sistema il **controllo degli accessi** si esprime tramite la definizione di **tre** livelli concettuali:

- **modelli**
- **politiche**
- **meccanismi**

Modelli

Un **modello di protezione** definisce i **soggetti**, gli **oggetti** ai quali i soggetti hanno accesso ed i **diritti** di accesso:

- **oggetti** costituiscono la parte **passiva**, cioè le risorse fisiche e logiche alle quali si può accedere e su cui si può operare. Ad es: i file.
- **soggetti** rappresentano la parte **attiva** di un sistema, cioè le entità che possono richiedere l'accesso alle risorse. Ad es: gli **utenti**, o i **processi** che eseguono per conto degli **utenti**.
- **diritti di accesso**: sono le **operazioni** con le quali è possibile operare sugli oggetti.

NB: Un soggetto può avere **diritti di accesso** sia per gli oggetti che per altri soggetti (ad es. un processo può controllarne un altro).

Politiche

Le **politiche di protezione** definiscono le regole con le quali i soggetti possono accedere agli oggetti.

Classificazione delle politiche:

- **Discretionary access control (DAC)**. Il soggetto creatore di un oggetto controlla i diritti di accesso per quell'oggetto (UNIX). La definizione delle politiche è decentralizzata.
- **Mandatory access control (MAC)**. I diritti di accesso vengono definiti in modo centralizzato. Installazioni di alta sicurezza (es., enti governativi).
- **Role Based Access Control (RABC)**. Ad un ruolo sono assegnati specifici diritti di accesso sulle risorse. Gli utenti possono appartenere a diversi ruoli. I diritti attribuiti ad ogni ruolo vengono assegnati in modo centralizzato.

Principio del privilegio minimo

Ad ogni soggetto sono garantiti i diritti di accesso solo agli oggetti strettamente necessari per la sua esecuzione (**POLA principle of least authority**).

Caratteristica **desiderabile** per tutte le politiche di protezione.

Meccanismi

I **meccanismi di protezione** sono gli strumenti messi a disposizione dal sistema di protezione per imporre una determinata politica.

Principi di realizzazione:

- **Flessibilità del sistema di protezione**: i meccanismi di protezione devono essere sufficientemente **generali** per consentire l'applicazione di diverse politiche di protezione.
- **Separazione tra meccanismi e politiche**. La politica definisce ***cosa va fatto*** ed il meccanismo ***come va fatto***.
E' desiderabile la massima indipendenza tra le due componenti.

Esempio: Unix

- L'utente (politica **DAC**) definisce la **politica**, ovvero il valore dei bit di protezione per ogni oggetto di sua proprietà.
- Il S.O: fornisce un **meccanismo** per definire e interpretare per ciascun file i tre bit di *read*, *write* e *execute* per il proprietario del file, il gruppo e gli altri.

Dominio di protezione

Ad ogni **soggetto** è associato un **dominio**, che rappresenta *l'ambiente di protezione* nel quale il soggetto esegue; il dominio specifica i diritti di accesso posseduti dal soggetto (es: utente) nei confronti di ogni risorsa.

Le operazioni vengono svolte da **processi** che operano per conto di soggetti (a cui sono associati i domini)

Un dominio di protezione è **unico per ogni soggetto**, mentre un processo può eventualmente **cambiare dominio** durante la sua esecuzione.

In questo caso, ad esempio:

- un processo P può eseguire inizialmente in un dominio di protezione D_i per conto del soggetto S_i ;
- in una fase di tempo successiva, P può eseguire in un altro dominio D_j per conto del soggetto S_j .

Dominio di protezione

DEF: Un dominio definisce un insieme di coppie, ognuna contenente l'identificatore di un oggetto e l'insieme delle operazioni che il soggetto associato al dominio può eseguire su ciascun oggetto (diritti di accesso):

$$D(S) = \{ \langle o, \text{diritti} \rangle \mid o \text{ è un oggetto, diritti è un insieme di operazioni} \}$$

Ogni **dominio** è associato univocamente ad un **soggetto**; il soggetto può accedere solo agli oggetti definiti nel suo dominio, utilizzando i diritti specificati dal dominio.

Domini disgiunti o domini con diritti di accesso in comune.

Possibilità per due o più soggetti di effettuare alcune operazioni comuni su un oggetto condiviso:

D1

<File1, (read, write)>
<File3, (execute) >

D2

<File1, (execute) >
<File2, (read, write) >

D3

<File2, (read) >
<File3, (read) >


Le operazioni vengono svolte da processi che operano per conto di soggetti (a cui sono associati i domini)

In ogni istante della sua esecuzione, il processo esegue in uno ed un solo dominio.

Associazione tra processo e dominio

Statica: l'insieme delle risorse disponibili ad un processo rimane fisso durante il suo tempo di vita.

Osservazioni:

- L'insieme globale delle risorse che un processo potrà usare può non essere un'informazione disponibile prima dell'esecuzione del processo.
 - L'insieme minimo (politica del minimo privilegio) delle risorse necessarie ad un processo cambia dinamicamente durante l'esecuzione.
-  L'associazione statica non è adatta nel caso si voglia limitare per un processo l'uso delle risorse a quello strettamente necessario (privilegio minimo).

Associazione tra un processo ed un dominio

Dinamica: associazione tra processo e dominio varia durante l'esecuzione del processo.

In questo modo si può mettere in pratica il principio del privilegio minimo: cambiando dinamicamente dominio, in ogni fase della sua esecuzione il processo acquisisce diritti diversi (privilegio minimo: solo quelli strettamente necessari).

→ Occorre un meccanismo per consentire il passaggio da un dominio all'altro del processo.

Cambio di dominio: esempio

Standard dual mode (kernel/user mode):

Due **domini (ring)** di protezione: quello dell'utente (user mode) e quello del kernel (monitor o kernel mode):

- Cambio di dominio associato alle system call: quando un processo deve eseguire un'istruzione privilegiata (accesso ai file, alle funzioni di rete, generazione dei thread etc.) avviene un cambio di dominio.

non realizza la protezione tra utenti, ma solo tra kernel e utente.

Esempio: Unix

Dominio associato all'utente: ogni processo è caratterizzato dall'attributo UserID (UID). Il cambio di dominio corrisponde al cambio temporaneo di identità (UID) del processo.

Possibilità di cambio di dominio:

Ad ogni file sono associati il proprietario (user-id) e un bit di dominio (set-uid).

Quando un utente A (user-id=A) lancia l'esecuzione di un file P il cui proprietario è B (user-id=B) ed il file ha set-uid=on, al processo che esegue P viene assegnato lo user-id di B.



P entra nel dominio di B

Matrice degli accessi

Un sistema di protezione può essere rappresentato a livello astratto utilizzando la **matrice degli accessi**.

	O1	O2	O3
S1	read,write	execute	write
S2		execute	read,write,

- Ogni riga è associata a un soggetto (es: utente); riga=dominio
- Ogni colonna è associata a un oggetto (es: risorsa, file)

Matrice degli accessi

La matrice consente di rappresentare il **modello** e le **politiche** valide nel sistema considerato, specificando:

- ▣ i **soggetti**
- ▣ gli **oggetti**
- ▣ i **diritti** accordati ai soggetti sugli oggetti

Le informazioni contenute nella matrice degli accessi possono **variare nel tempo**, per effetto di operazioni che ne consentono la modifica. (es: aggiunta/rimozione oggetti, aggiunta/rimozione soggetti, aggiunta/rimozione diritti).

-> Le informazioni contenute nella matrice all'istante t , rappresentano lo **stato di protezione** del sistema in t .

La matrice degli accessi offre ai **meccanismi** di protezione le informazioni che consentono di verificare il rispetto dei vincoli di accesso.

Meccanismi

Il meccanismo:

- ha il compito di verificare se ogni richiesta di accesso che proviene da un processo che opera in un determinato dominio è consentita oppure no.
- autorizza l'esecuzione delle richieste consentite e impedisce quelle vietate.
- esegue la modifica (in modo controllato) dello stato di protezione in seguito ad ogni richiesta autorizzata da parte di un processo

Verifica del rispetto dei vincoli di accesso:

Il meccanismo consente di assicurare che un processo che opera nel dominio D_i può accedere solo agli oggetti specificati nella riga i e solo con i diritti di accesso indicati.

Quando un'operazione M deve essere eseguita nel dominio D_i sull'oggetto O_j , il meccanismo consente di controllare che M sia contenuta nella casella $\text{access}(i,j)$.

In caso affermativo l'operazione può essere eseguita. In caso negativo si ha una situazione di errore.

Modifica dello stato di protezione

Chi può modificare lo stato di protezione?

MAC: entità centrale

DAC: i soggetti (es. utenti)

Modello di Graham –Denning

La modifica controllata dello stato di protezione può essere ottenuta tramite un opportuno insieme di comandi (Graham e Denning, 1972).

8 primitive:

1. create object
2. delete object
3. create subject
4. delete subject
5. read access right
6. grant access right
7. delete access right
8. transfer access right

Creazione, eliminazione di righe e colonne (1,2,3,4)

Modifica dei diritti di accesso: aggiunta, cancellazione e **propagazione (5,6,7,8)**.

Propagazione dei diritti di accesso

La possibilità di copiare un diritto di accesso per un oggetto da un dominio ad un altro della matrice di accesso è indicato con un asterisco (*) (copy flag):

- Un soggetto S_i può trasferire un diritto di accesso α per un oggetto X ad un altro soggetto S_j solo se S_i ha accesso a X con il diritto α , e α ha il copy flag.

copy flag

	O1	O2	O3	S1	S2
S1	read*, write	execute	write		
S2		execute	read, write		

Es: S1 può propagare a S2 il diritto read per O1.

L'operazione di propagazione può essere realizzata in due modi:

- Trasferimento del diritto
- Copia del diritto

copy flag

	O1	O2	O3	S1	S2
S1	write	execute	write		
S2	read*	execute	read, write		

Trasferimento del diritto: il soggetto S1 trasferisce read* e perde il diritto per l'oggetto O1.

copy flag

	O1	O2	O3	S1	S2
S1	read*, write	execute	write		
S2	read	execute	read, write		

Copia del diritto: viene copiato solo read (propagazione limitata del diritto), mentre il soggetto S1, mantiene read*.

Diritto owner

Assegnazione di un qualunque diritto di accesso su un oggetto X ad un qualunque soggetto S_j da parte di un soggetto S_i:

L'operazione è consentita solo se il diritto **owner** appartiene a A[S_i,X].

Esempio: S2 può concedere/revocare qualunque diritto su O2 a qualunque utente

	O1	O2	O3	S1	S2
S1	read*, write	execute	write		
S2		execute, <u>owner</u>	read, write		

Diritto owner

Assegnazione di un qualunque diritto di accesso su un oggetto X ad un qualunque soggetto S_j da parte di un soggetto S_i:

L'operazione è consentita solo se il diritto **owner** appartiene a A[S_i,X].

Esempio: S2 può concedere il diritto write su O2 a S1

	O1	O2	O3	S1	S2
S1	read*, write	execute write	write		
S2		execute, <u>owner</u>	read, write		

Diritto owner

Assegnazione di un qualunque diritto di accesso su un oggetto X ad un qualunque soggetto S_j da parte di un soggetto S_i:

L'operazione è consentita solo se il diritto **owner** appartiene a A[S_i,X].

Esempio: S2 può **revocare** il diritto execute su O2 a sè stesso

	O1	O2	O3	S1	S2
S1	read*, write	execute write	write		
S2		execute , <u>owner</u>	read, write		

Diritto control

Eliminazione di un diritto di accesso per un oggetto X nel dominio di S_j da parte di S_i.

L'operazione è consentita solo se il diritto **control** appartiene a A[S_i,S_j], oppure owner appartiene a A[S_i,X]

Esempio, S₁ può revocare a S₂ il diritto write su O₃.

	O1	O2	O3	S1	S2
S1	read*, write	execute write	write owner		control
S2		owner	read, write		

Cambio di dominio: switch

Cambio di dominio: un processo che esegue nel dominio del soggetto

Si può commutare al dominio di un altro soggetto S_j .

L'operazione è consentita solo se il diritto **switch** appartiene a $A[S_i, S_j]$

Esempio, un processo che esegue nel dominio di S_2 può trasferirsi nel dominio di S_1 .

	O1	O2	O3	S1	S2
S1	read*, write	execute	write owner		
S2		owner	read	switch	

Realizzazione della matrice degli accessi

La Matrice degli accessi è una notazione astratta che rappresenta lo stato di protezione.

Nella rappresentazione concreta è necessario considerare:

- Dimensione della matrice
- Matrice sparsa

-> Rappresentare le informazioni contenute nella matrice degli accessi in una struttura dati matriciale $N_s \times N_o$ risulterebbe non ottimale per l'occupazione della memoria.

Realizzazione della matrice degli accessi

La Rappresentazione concreta dello stato di protezione deve essere ottimizzata sia riguardo all'**occupazione della memoria**, sia rispetto all'**efficienza** nell'accesso e nella gestione delle informazioni di protezione.

Due approcci:

Access Control List (ACL):

Rappresentazione per colonne: per ogni oggetto è associata una lista che contiene tutti i soggetti che possono accedere all'oggetto, con i relativi diritti di accesso per l'oggetto.

Capability List:

Rappresentazione per righe: ad ogni soggetto è associata una lista che contiene gli oggetti accessibili dal soggetto ed i relativi diritti di accesso

ACL: Lista degli accessi

La lista degli accessi **per ogni oggetto** è rappresentata dall'insieme delle coppie:

<oggetto, insieme dei diritti >

limitatamente ai soggetti con un insieme non vuoto di diritti per l'oggetto.

Quando deve essere eseguita un'operazione M su un oggetto Oj da parte di Si, si cerca nella lista degli accessi

<Si,Rk>, con M appartenente a Rk

La ricerca può essere fatta preventivamente in una lista di **default** (se prevista) contenente i diritti di accesso applicabili a tutti gli oggetti.

Se in entrambi i casi la risposta è negativa, **l'accesso è negato**.

Utenti & Gruppi

Solitamente ogni soggetto rappresenta un singolo utente.

Molti sistemi hanno il concetto di **gruppo di utenti**. I gruppi hanno un nome e possono essere inclusi nella ACL.

In questo caso l'entry in ACL ha la forma:

$UID_1, GID_1: < insieme di diritti >$

$UID_2, GID_2: < insieme di diritti >$

Dove UID è lo user identifier e GID il group identifier.

Capability list

La lista delle capability, **per ogni soggetto**, è la lista di elementi ognuno dei quali:

- è associato a un oggetto a cui il soggetto può accedere
- contiene i diritti di accessi consentiti su tale oggetto.

Ogni elemento della lista prende il nome di **capability**.

La capability si compone di un identificatore (o un indirizzo) che identifica l'oggetto e la rappresentazione (es. una sequenza di bit) dei vari diritti concessi.

Quando S intende eseguire un'operazione M su un oggetto Oj: il meccanismo di protezione controlla se nella lista delle capability associata a S ne esiste una relativa ad Oj che abbia tra i suoi diritti M.

	F1	F2	F3	F4	F5	F6	PR1	PR2
Soggetto S	-	-	R	RWE	RW	-	W	-

Tipo	Diritti	Oggetto
File	R	Puntatore a F3
File	RWE	Puntatore a F4
File	RW	Puntatore a F5
Printer	W	Puntatore a stampante PR1

capability list per il soggetto S

Le liste di capability devono essere protette da manomissioni.

Ciò si può ottenere facendo in modo che ogni capability list venga **gestita solo dal S. O.**: l'utente fa riferimento ad un puntatore (capability) che identifica la sua posizione nella lista appartenente allo spazio del kernel (soluzione simile all'uso dei file descriptor in UNIX).

Ogni intervento su CL viene ottenuto tramite system call.

ACL vs CL

Un sistema di protezione realizzato esclusivamente con ACL o capability list può presentare alcuni problemi di efficienza:

ACL. L'informazione di quali diritti di accesso possieda un **soggetto** S è sparsa nelle varie ACL relative agli oggetti del sistema.

Capability list. L'informazione relativa a tutti i diritti di accesso applicabili ad un certo **oggetto** O è sparsa nelle varie CL.

Es: Revoca dei diritti di accesso

In un sistema di protezione può essere necessario **revocare** i diritti di accesso per **un oggetto**.

La revoca può essere:

- Generale o selettiva, cioè valere per tutti gli utenti che hanno quel diritto di accesso o solo per un gruppo.
- Parziale o totale, cioè riguardare un sottoinsieme di diritti per l'oggetto, o tutti .
- Temporanea o permanente, cioè il diritto di accesso non sarà più disponibile, oppure potrà essere successivamente riottenuto.

Revoca per un oggetto con ACL:

In un sistema a liste di accesso la revoca risulta semplice. Si fa riferimento alla ACL associata all'oggetto e si cancellano i diritti di accesso che si vogliono revocare.

Revoca per un oggetto con Capability List:

L'operazione risulta più complessa in un sistema a lista di capability. E' necessario infatti verificare per ogni dominio se contiene la capability con riferimento all'oggetto considerato.

Es: Cancellazione/aggiunta di un utente

In un sistema multi-user l'amministratore può applicare modifiche all'insieme degli utenti autorizzati.

- Cancellazione di un utente U esistente \rightarrow necessità di eliminare ogni traccia dell'utente U dal sistema di protezione, compresi i diritti accordati a U sugli oggetti del sistema.
- Aggiunta di un nuovo utente \rightarrow necessità di inizializzare il sistema di protezione aggiungendo U e i diritti concessi a U sulle risorse.

Cancellazione di un Utente U in un sistema con con ACL:

In un sistema a liste di accesso la cancellazione di U è complessa. E' necessario eliminare da ogni ACL gli eventuali elementi associati a U.

Cancellazione di un Utente U in un sistema con Capability List:

è sufficiente eliminare la CL associata all'utente eliminato.

ACL vs. Capability list

ACL è più conveniente nelle operazioni che riguardano singoli oggetti, mentre il metodo delle Capability List è vantaggioso quando si devono compiere azioni sui singoli soggetti.

Soluzione:

La soluzione che viene adottata nella maggior parte dei sistemi è di usare una **combinazione** dei due metodi.

Soluzione mista

ACL memorizzate in forma persistente (es. disco).

Se un Soggetto tenta di accedere ad un oggetto per la prima volta:

- Si analizza la ACL; se esiste una entry contenente il nome del soggetto e se tra i diritti di accesso c'è quello richiesto dal soggetto, viene fornita la capability per l'oggetto (in memoria volatile).
- Ciò consente al soggetto di accedere all'oggetto più volte senza che sia necessario analizzare la ACL.
- Dopo l'ultimo accesso la capability è distrutta.

Esempio: S.O. Unix.

Apertura di un file

`fd=open(<nome file>, <diritti di accesso>)`

- Si cerca il file nel direttorio e si verifica se l'accesso è consentito (ACL, v. inode e bit di protezione).
- In caso affermativo viene creata una nuova entry nella tabella dei file aperti, costituita da fd e dai diritti di accesso.
- Viene ritornato al processo fd, cioè il riferimento al nuovo file aperto.
- Tutte le successive operazioni sul file sono eseguite utilizzando direttamente fd (capability).

Protezione e Sicurezza

La protezione riguarda il controllo degli accessi alle risorse interne al sistema.

La sicurezza riguarda il controllo degli accessi al sistema.

La protezione di un sistema può essere inefficace, se un utente non fidato riesce a far eseguire programmi che agiscono sulle risorse del sistema (es. Cavalli di Troia).

Sicurezza multilivello

La maggior parte dei sistemi operativi permette a singoli utenti di determinare chi possa leggere e scrivere i loro file ed i loro oggetti. (DAC, controllo discrezionale degli accessi).

In alcuni ambiti è richiesto un più stretto controllo sulle regole di accesso alle risorse (es. ambiente militare, aziende, ospedali, ecc.). Vengono stabilite regole **generali** su «chi può accedere a che cosa» che possono essere modificate solo da un'entità centrale autorizzata a farlo.

(MAC, controllo degli accessi obbligatorio)

Sicurezza multilivello

L'organizzazione a cui appartiene il sistema definisce delle politiche **MAC** che stabiliscono **regole generali** su chi può accedere e a che cosa, tramite l'adozione di un **modello di sicurezza**.

Tali regole **si aggiungono alle politiche di protezione**.

I modelli di sicurezza più usati sono due:

- il modello **Bell-La Padula**
- il modello **Biba**.

Entrambi sono modelli **multilivello**.

Modelli di sicurezza multilivello

In un modello di sicurezza multilivello:

I **soggetti** (utenti) e gli **oggetti** (risorse, es. file) sono **classificati** in **livelli** (classi di accesso):

- Livelli per i soggetti (**clearance levels**)
- Livelli per gli oggetti (**sensitivity levels**)

Il modello fissa inoltre le **regole di sicurezza**, che controllano il flusso delle informazioni tra i livelli.

Modello Bell-La Padula

Progettato per realizzare la sicurezza in organizzazioni militari, garantendo la **confidenzialità** delle informazioni.

Associa a un sistema di protezione (matrice degli accessi) due regole di sicurezza MAC, che stabiliscono il di propagazione delle informazioni nel sistema.

Quattro Livelli di sensibilità degli oggetti (o documenti):

1. Non classificato
2. Confidenziale
3. Segreto
4. Top secret

Quattro Livelli di autorizzazione (clearance) per i soggetti:

Gli utenti sono assegnati ai livelli a seconda del loro ruolo nell'organizzazione, ovvero dei documenti che è loro consentito esaminare.

Modello Bell-La Padula

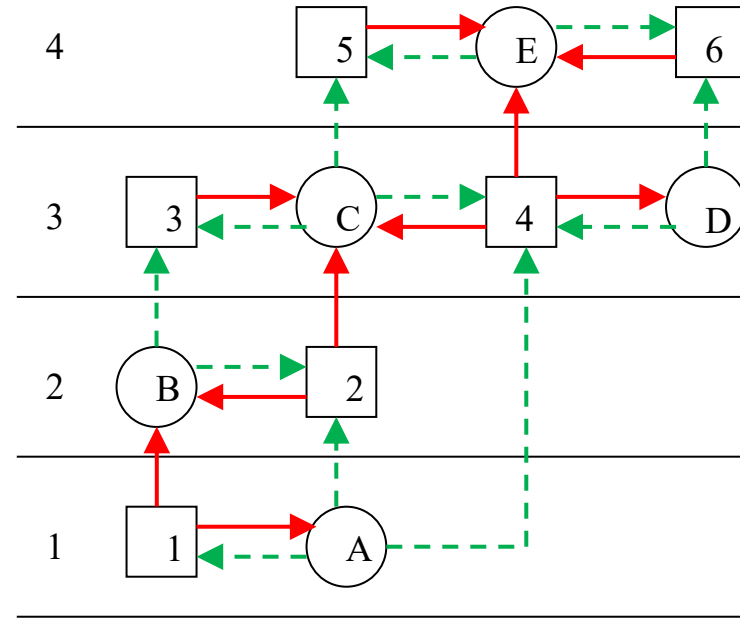
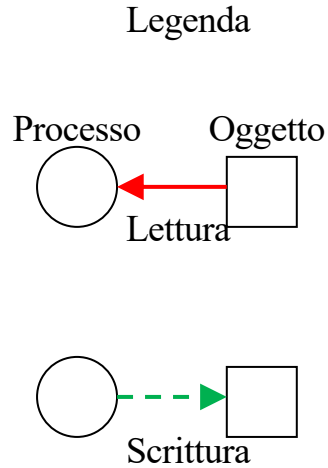
Regole di sicurezza:

- 1. Proprietà di semplice sicurezza:** un processo in esecuzione al livello di sicurezza k può leggere solo oggetti al suo livello o a livelli inferiori.
- 2. Proprietà *:** un processo in esecuzione al livello di sicurezza k può scrivere solamente oggetti al suo livello o a quelli superiori

Pertanto: i processi possono leggere verso il basso e scrivere verso l'alto, ma non il contrario.

-> Il flusso delle informazioni è dal basso verso l'alto e non viceversa.

In generale, a queste regole **si aggiungono le regole di protezione** specificate dalla matrice degli accessi.



Il modello Bell-La Padula è stato concepito per mantenere i segreti, non per garantire l'integrità dei dati. E' possibile, infatti, **sovrascrivere** l'informazione appartenente ad un livello superiore.

NB: il sistema di protezione può imporre ulteriori limitazioni.

Esempio Bell-LaPadula: Difesa dai cavalli di Troia

Nell'esempio viene usato un “*cavallo di Troia*” per aggirare un meccanismo di controllo basato sulle liste di controllo degli accessi (ACL).

Un utente, Paolo, ha creato il file **Fp**, contenente la stringa di caratteri riservati “CPE1704TKS”, con i permessi di lettura/scrittura solo per i processi che appartengono a lui.

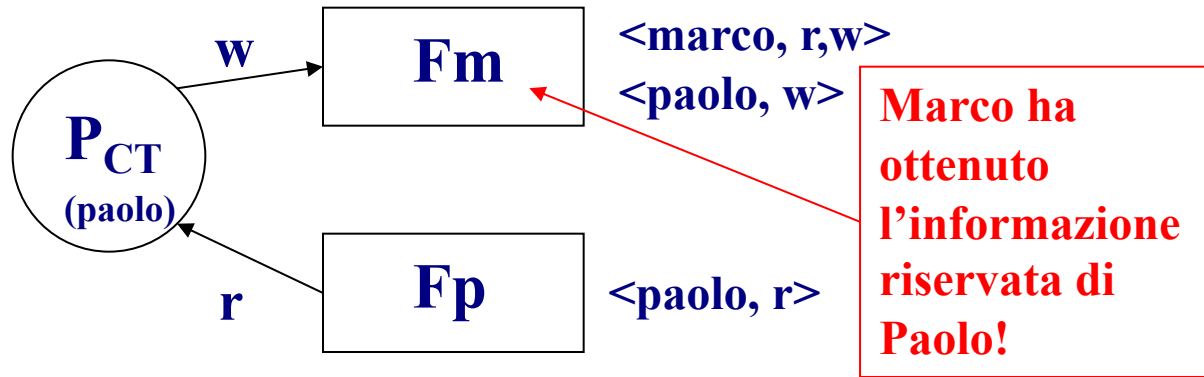
Un utente ostile, Marco, ottenuto l'accesso al sistema, installa un file eseguibile **CT** (il cavallo di Troia) e copia nel filesystem un file privato **Fm** che verrà utilizzato come “tasca posteriore”.

Regole di protezione (es. ACL):

- Marco ha permessi di lettura e scrittura per il suo file Fm.
- Marco dà a Paolo il permesso di scrittura su Fm.
- Marco concede a Paolo il diritto di esecuzione su CT
- Il file Fp (contenente i dati da proteggere) è leggibile solo da Paolo.

Marco induce Paolo ad eseguire il cavallo di Troia CT (per esempio, spacciandolo come un programma di utilità).

Il programma CT, eseguito da Paolo tramite il processo P_{CT} copia la stringa dei caratteri riservati nel file “tasca posteriore” Fm di Marco: sia l’operazione di lettura che quella di scrittura soddisfano i vincoli imposti dalle ACL.



Difesa dai cavalli di Troia mediante l'uso del modello Bell-La Padula

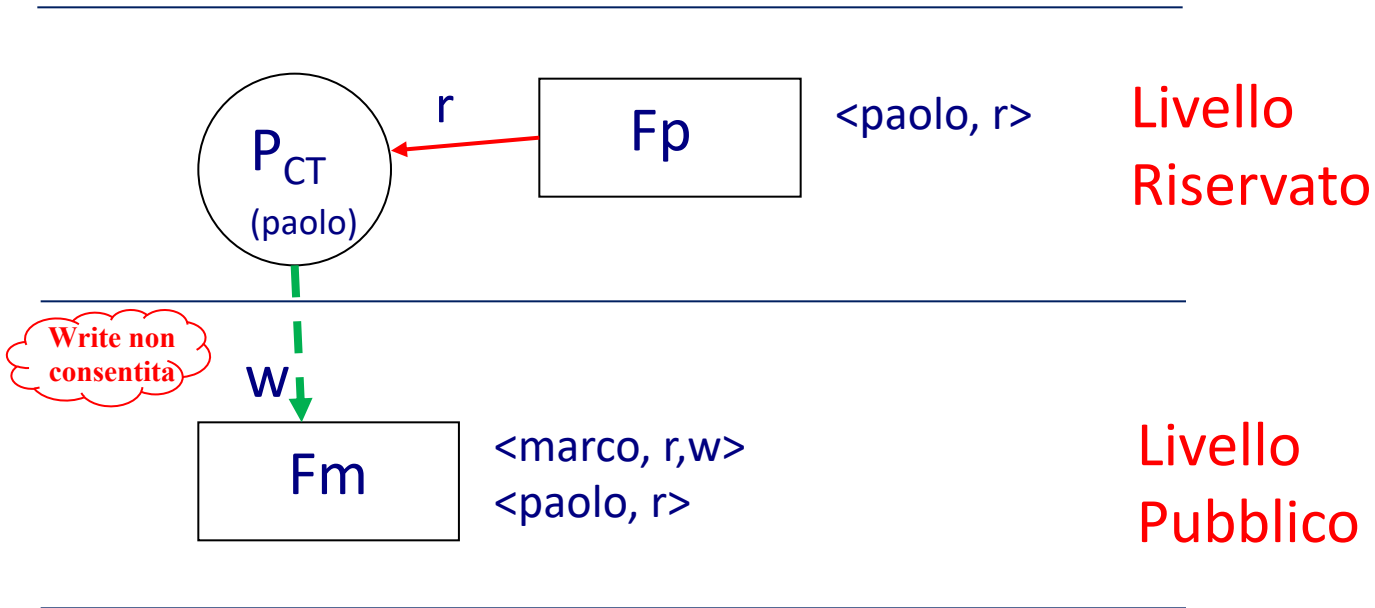
Vengono fissati **2 livelli di sicurezza**, *riservato e pubblico*:

- Ai processi ed al file dati F_p di Paolo viene assegnato il livello di sicurezza “riservato”.
- A quelli di Marco (F_m e CT) il livello “pubblico”.

Quando Paolo esegue CT , il processo P_{ct} creato acquisisce il livello di sicurezza di Paolo (riservato) e può vedere la stringa di caratteri riservata S .

Quando P_{ct} tenta di scrivere S nel file F_m pubblico (la «tasca posteriore») la proprietà * è violata ed il **tentativo viene negato dal sistema** (*v. reference monitor*), **nonostante l'ACL lo consenta**.

(La politica di sicurezza ha la precedenza sulle regole ACL)



Sicurezza Multilivello

Il modello Bell-La Padula è stato concepito per mantenere i segreti, non per garantire l'integrità dei dati.

Modello Biba

Obiettivo: integrità dei dati.

1. **Proprietà di semplice sicurezza:** un processo in esecuzione al livello di sicurezza k può scrivere solamente oggetti al suo livello o a quelli inferiori (nessuna scrittura verso l'alto).
2. **Proprietà di integrità*:** un processo in esecuzione al livello k può leggere solo oggetti al suo livello o a quelli superiori (nessuna lettura verso il basso)

B-LP vs. BIBA: I due modelli sono in conflitto tra loro e non si possono utilizzare contemporaneamente.

Architettura dei sistemi ad elevata sicurezza

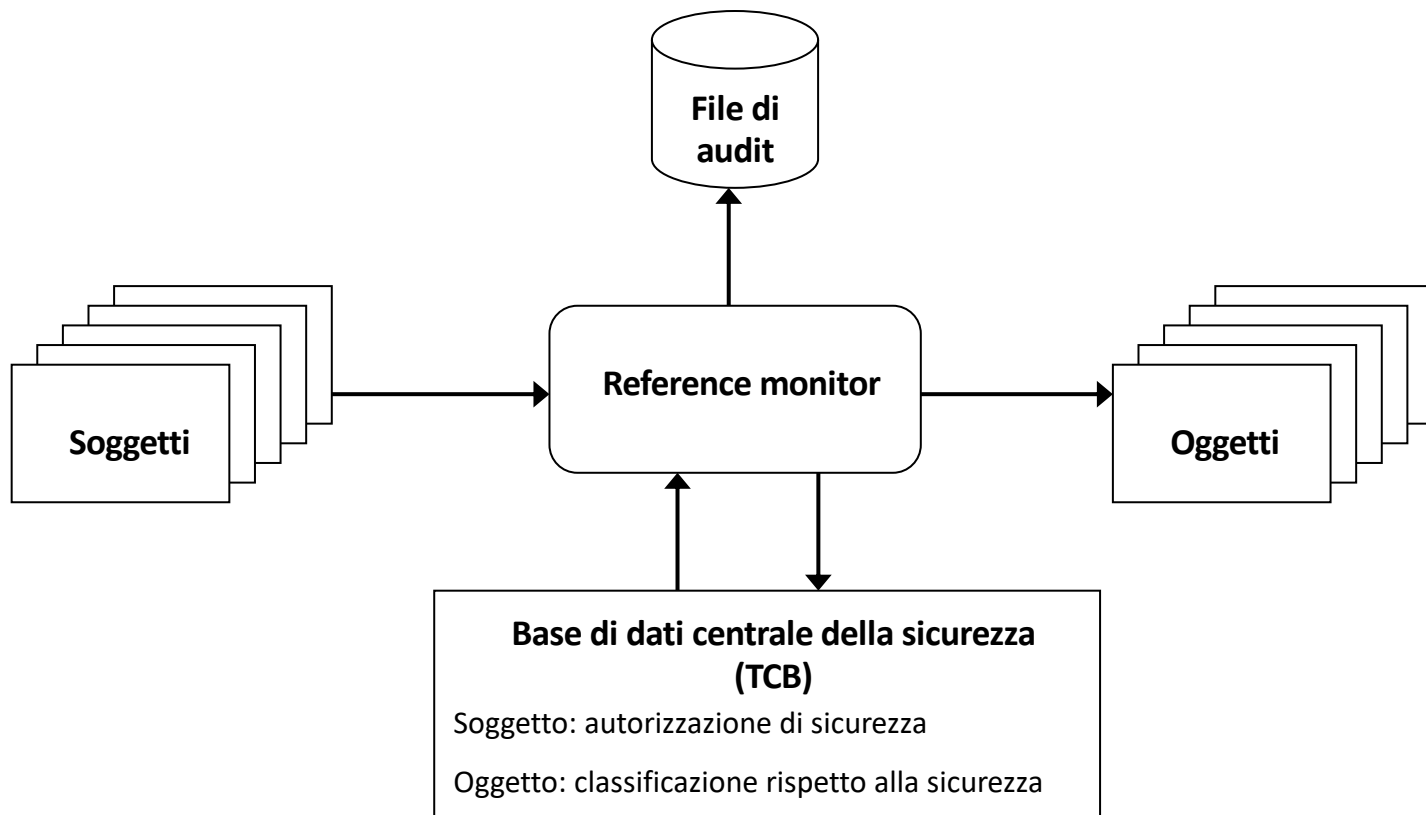
Sistemi operativi sicuri, o fidati: sistemi per i quali è possibile definire formalmente dei requisiti di sicurezza.

Reference monitor: E' un elemento di controllo realizzato dall'hardware e dal S.O. che regola l'accesso dei soggetti agli oggetti sulla base di parametri di sicurezza (es. modello Bell-La Padula).

Trusted computing base: Il RM ha accesso a una base di calcolo fidata (Trusted Computing Base, o TCB) che contiene:

- Privilegi di sicurezza (autorizzazioni di sicurezza) di ogni soggetto.
- Attributi (classificazione rispetto alla sicurezza) di ciascun oggetto.

Architettura di sistemi ad elevata sicurezza



Sistemi fidati

Il RM impone le regole di sicurezza (B-LP: no read-up, no-write down) ed ha le seguenti proprietà:

Mediazione completa: le regole di sicurezza vengono applicate ad ogni accesso e non solo, ad esempio, quando viene aperto un file.

Isolamento: il monitor dei riferimenti e la base di dati sono protetti rispetto a modifiche non autorizzate (es. kernel).

Verificabilità: la correttezza del RM deve esser provata, cioè deve esser possibile dimostrare formalmente che il monitor impone le regole di sicurezza ed fornisce mediazione completa ed isolamento

Il requisito di **mediazione completa** rende preferibile, per motivi di efficienza, che la soluzione debba essere almeno parzialmente hardware.

Il requisito dell'**isolamento** impone che non sia possibile per chi porta l'attacco, modificare la logica del reference monitor o il contenuto della base di dati centrale della sicurezza.

Il requisito della **verificabilità** è difficile da soddisfare per un sistema general-purpose.

Audit file

Vengono mantenuti in questo file gli eventi importanti per la sicurezza, come i tentativi di violazione alla sicurezza e le modifiche autorizzate alla base di dati del nucleo di sicurezza.

Classificazione della sicurezza dei sistemi di calcolo

Orange Book. Documento pubblicato dal Dipartimento della Difesa americano (D.O.D). Sono specificate quattro categorie di sicurezza: A, B, C, D (in ordine decrescente).

- **Categoria D: Minimal Protection**

Non prevede sicurezza. Esempio MS-DOS, Windows 3.1.

- **Categoria C: Discretionary Protection**

Suddivisa in C1 e C2.

C1. La TCB consente:

- Autenticazione degli utenti (password). I dati di autenticazione sono protetti, rendendoli inaccessibili agli utenti non autorizzati.
- Protezione dei dati e programmi propri di ogni utente.
- Controllo degli accessi a oggetti comuni per gruppi di utenti definiti

Esempio: Unix

- C2. La TCB consente, oltre a quanto definito per la C1, il controllo degli accessi su una base individuale. Esempio Windows.

- **Categoria B: Mandatory Protection**

B1. La TCB consente, oltre a quanto definito in C2, l'introduzione dei **livelli di sicurezza** (modello Bell-La Padula). Almeno due livelli.

B2. La TCB estende l'uso di etichette di riservatezza ad ogni risorsa del sistema, compresi i canali di comunicazione.

B3. La TCB consente la creazione di liste di controllo degli accessi in cui sono identificati utenti o gruppi cui non è consentito l'accesso ad un oggetto specificato.

Es. XTS-400 (STOP kernel), 2003

- **Categoria A: Verified Protection**

Suddivisa in A1 e classi superiori

A1. E' equivalente a B3, ma con il vincolo di essere progettato e realizzato utilizzando metodi formali di definizione e verifica (es. Honeywell SCOMP, GEMSOS, Boeing SNS Server)

Un sistema appartiene ad una classe superiore ad A1 se è stato progettato e realizzato in un impianto di produzione affidabile da persona affidabile