Debolezze del DAC

Il controllo degli accessi discrezionario controlla sono gli accessi diretti

Nessun controllo su cosa succede all'informazione una volta che è rilasciata

$\Longrightarrow$ DAC è vulnerabile a Trojan horse che <u>sfruttano privilegi del soggetto chiamante</u>

Trojan Horse: Programma con una funzione nota utile ma che contiene codice nascosto che esegue funzioni (non legittime) ad insaputa del soggetto chiamante.

Virus and bombe logiche sono generalmente trasmessi nella forma di Trojan Horse

The Trojan Horse problem

File Market

Aug. 00; product X; price 7,000
Dec. 00; product Y; price 3,500
Jan. 01; product Z; price 1,200

`owner` Jane

The Trojan Horse problem

File Market

Aug. 00; product X; price 7,000
Dec. 00; product Y; price 3,500
Jan. 01; product Z; price 1,200

`owner` Jane

John

The Trojan Horse problem

Application

File Market

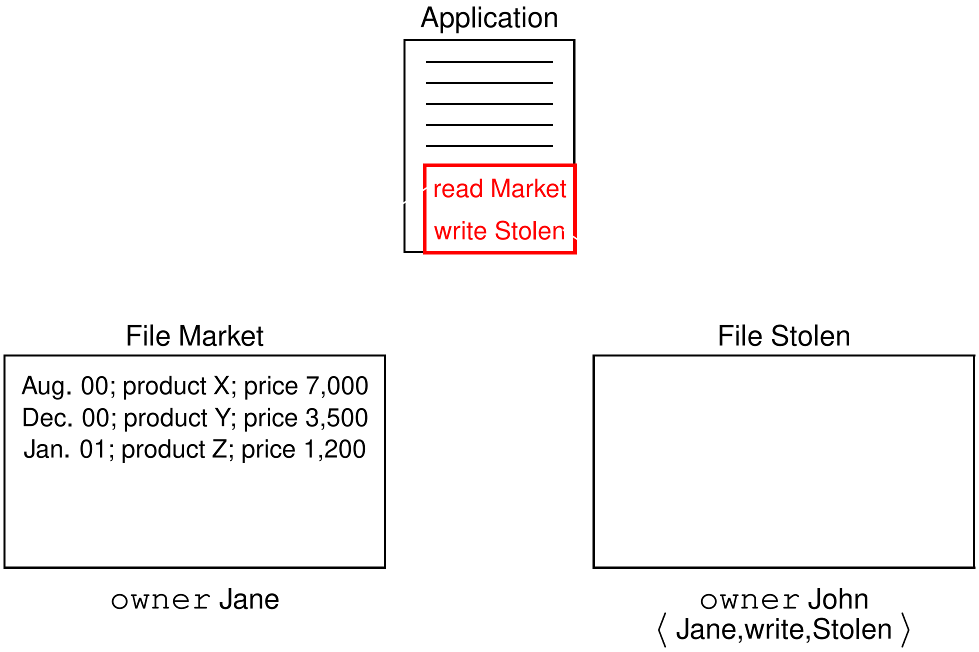
Aug. 00; product X; price 7,000
Dec. 00; product Y; price 3,500
Jan. 01; product Z; price 1,200

`owner` Jane

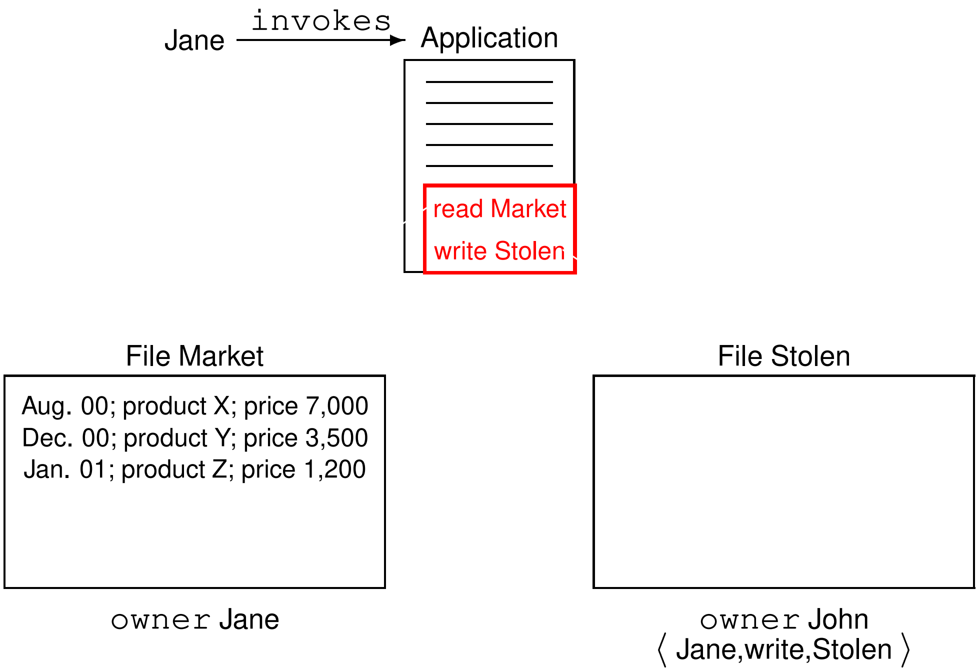
File Stolen

`owner` John
⟨ Jane,write,Stolen ⟩

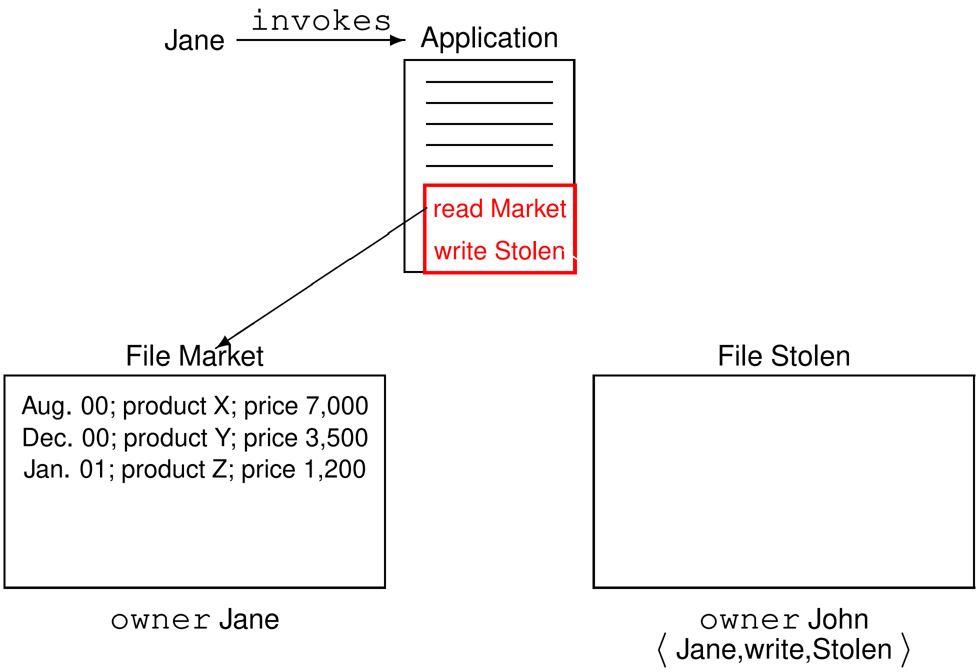
The Trojan Horse problem



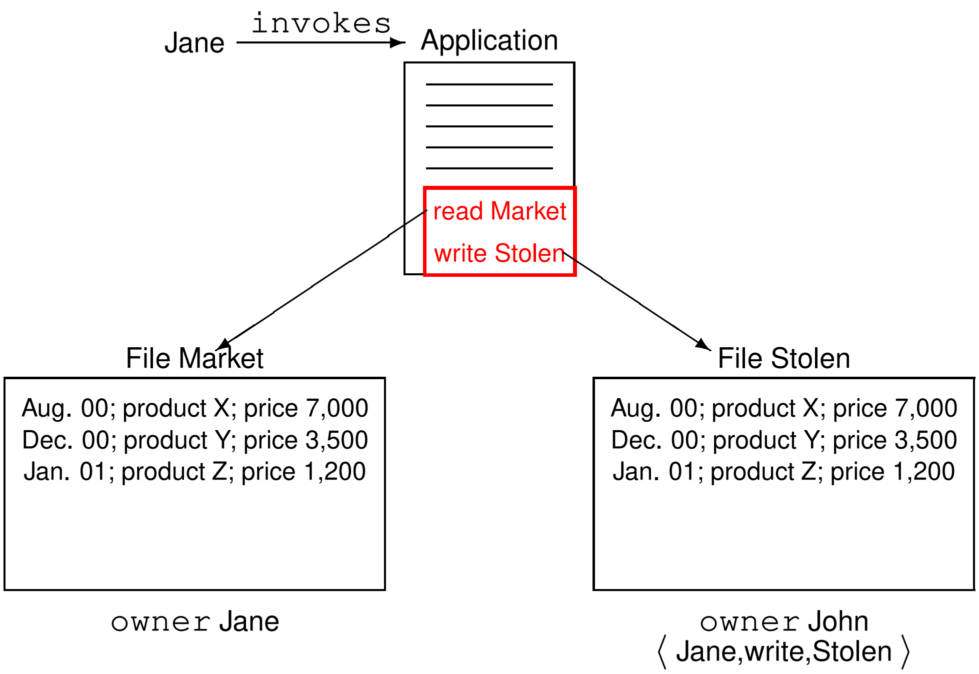
The Trojan Horse problem



The Trojan Horse problem



The Trojan Horse problem



Politiche mandatorie

Controllo dell'accesso mandatorio: Impongono restrizioni sul flusso di informazione che non possono essere bypassate da Trojan Horse.

Distinguono fra utenti e soggetti che operano per conto degli utenti.

- Utenti = Persone
- Soggetti = Processi (programmi in esecuzione). Operano per conto degli utenti.

Mentre ci può essere fiducia nel fatto che gli utenti non si comporteranno impropriamente, non c'è tale fiducia nei programmi che questi eseguono.

Politiche mandatorie

Basate sulla classificazione di soggetti e oggetti

Due classi di politiche

- Segretezza (es., modello Bell La Padula)
- Integrità (es., modello Biba)

La classificazione è un elemento di un insieme parzialmente ordinato di classi.

L'ordine parziale è definito da una relazione di dominanza, $\succeq$.

Classificazioni di sicurezza

Classi di sicurezza generalmente formate da due componenti

- Livello di sicurezza elemento di un insieme gerarchico di elementi. Es.,

  TopSecret(TS), Secret(S), Confidential(C), Unclassified(U)

  $$TS > S > C > U$$

  Crucial (C), Very Important (VI), Important (I)

  $$C > VI > I$$

- Categorie sottoinsieme di un insieme non gerarchico di elementi (es., Administrative, Financial). Può partizionare le differenti aree di competenza all'interno del sistema. Permette di implementare restrizioni "need-to-know".

Classificazioni di sicurezza

La relazione di dominanza è definita come segue:

$$(L_1, C_1) \succeq (L_2, C_2) \Longleftrightarrow L_1 \geq L_2 \wedge C_1 \supseteq C_2$$

I Criteria del DoD assumono classi di accesso di questo tipo;

Lo standard richiede la definizione di 16 livelli di classificazione e di 64 categorie.

Reticolo di classificazione

Le classi di sicurezza insieme con $\succeq$ formano un reticolo $(SC, \succeq)$

Riflessività di $\succeq$ $\forall x \in SC : x \succeq x$

Transitività di $\succeq$ $\forall x, y, z \in SC : x \succeq y, y \succeq z \Longrightarrow x \succeq z$

Antisimmetria di $\succeq$ $\forall x, y \in SC : x \succeq y, y \succeq x \Longrightarrow x = y$

Least upper bound $\forall x, y \in SC : \exists\, !z \in SC$

- $z \succeq x$ and $z \succeq y$
- $\forall t \in SC : t \succeq x$ and $t \succeq y \Longrightarrow t \succeq z$.

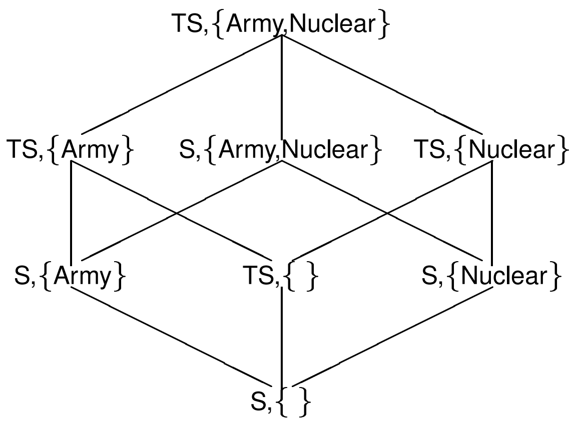
Greatest lower bound $\forall x, y \in SC : \exists\, !z \in SC$

- $x \succeq z$ and $y \succeq z$
- $\forall t \in SC : x \succeq t$ and $y \succeq t \Longrightarrow z \succeq t$.

Reticolo di classificazione – esempio

Livelli: Top Secret (TS), Secret (S)

Categorie: Army, Nuclear



- lub(⟨TS,{Nuclear}⟩,⟨S,{Army,Nuclear}⟩) = ⟨TS,{Army,Nuclear}⟩
- glb(⟨TS,{Nuclear}⟩,⟨S,{Army,Nuclear}⟩) = ⟨S,{Nuclear}⟩

Classificazione per controllo di segretezza

Ad ogni utente è associata una classe di accesso (clearance).

L'utente può connettersi al sistema ad ogni classe dominata dalla sua clearance.

I soggetti attivati in una sessione assumono la classificazione con la quale l'utente si è collegato.

Classi di segretezza

- assegnata ad un utente riflette la fiducia nel fatto che l'utente non rilasci l'informazione cui ha accesso ad altri che non hanno la clearance appropriata.
- assegnata ad un oggetto riflette la sensitività dell'informazione contenuta nell'oggetto e il potenziale danno che potrebbe risultare dalla sua impropria divulgazione.

Categorie definiscono area di competenza di utenti e dati (need to know)

Politica mandatoria per la segretezza

Goal: prevenire flusso di informazione verso classi di accesso più basse o non comparabili.

NO WRITE DOWN Un soggetto s può scrivere solo oggetti la cui classificazione domina quella di s ($\lambda(o) \succeq \lambda(s)$)

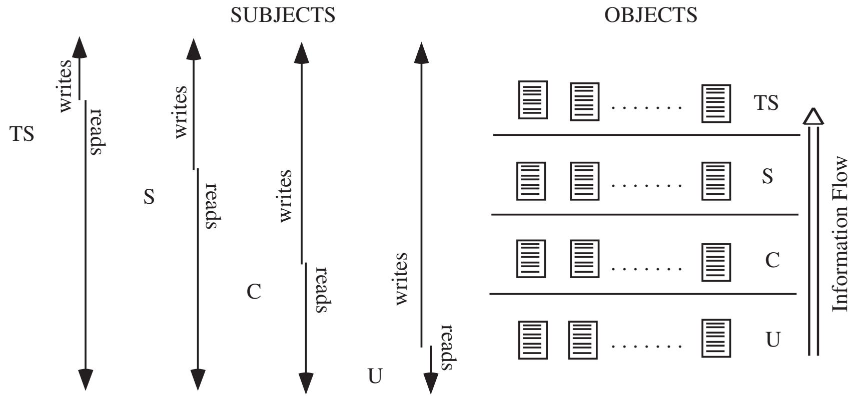
NO READ UP Un soggetto s può leggere solo oggetti o la cui classificazione è dominata da quella di s ($\lambda(s) \succeq \lambda(o)$)

Facile vedere che Trojan Horse che cercano di fare fluire informazioni lungo canali legittimi sono bloccati.

Utenti ad alto livello si possono collegare a livelli più bassi per scrivere informazioni non sensibili.

Gli oggetti creati prendono la classificazione del soggetto che li ha creati.

Flusso di informazione per segretezza



Modello di Bell e LaPadula

Definisce una politica mandatoria per la segretezza (ha introdotto i concetti di no-read-up, no-write-down).

Differenti versioni del modello (per correggere vulnerabilità o per particolari sistemi).

Il sistema è modellato come stati e transizioni di stato

Stato $v \in V$ tripla ordinata (b, M, λ)

- $b \in \wp(S \times O \times A)$: insieme degli accessi correnti nello stato v
- M: Matrice di accesso con S righe, O colonne, (entrate assumono valori in A)
- $\lambda : S \cup O \to L$: ritorna la classificazione di soggetti e oggetti

Modello di Bell e LaPadula – proprietà

simple security Stato (b, M, λ) è sicuro sse
$\forall(s, o, a) \in b, a = \texttt{read} \Rightarrow \lambda(s) \succeq \lambda(o)$

***-security** Stato (b, M, λ) è sicuro sse
$\forall(s, o, a) \in b, a = \texttt{write} \Rightarrow \lambda(o) \succeq \lambda(s)$

Uno stato è sicuro sse soddisfa security property e *-property.

Funzione di transizione di stato $T : V \times R \rightarrow V$ transforma uno stato sicuro in uno stato sicuro.

Un sistema (v_0, R, T) è sicuro sse v_0 è sicuro e ogni stato raggiungibile da v_0 eseguendo una sequenza finita di una o più richiesta da R è sicuro.

BLP – Basic security theorem

Theo Un sistema (v_o, R, T) è sicuro sse

- v_0 è uno stato sicuro
- T è tale che $\forall v$ raggiungibile da v_0 eseguendo una o più richieste da R, se $T(v, c) = v'$, dove $v = (b, M, \lambda)$, e $v' = (b', M', \lambda')$, allora $\forall s \in S, o \in O$:
  - $(s, o, \texttt{read}) \in b' \wedge (s, o, \texttt{read}) \notin b \Rightarrow \lambda'(s) \succeq \lambda'(o)$
  - $(s, o, \texttt{read}) \in b \wedge \lambda'(s) \not\succeq \lambda'(o) \Rightarrow (s, o, \texttt{read}) \notin b'$
  - $(s, o, \texttt{write}) \in b' \wedge (s, o, \texttt{write}) \notin b \Rightarrow \lambda'(o) \succeq \lambda'(s)$
  - $(s, o, \texttt{write}) \in b \wedge \lambda'(o) \not\succeq \lambda'(s) \Rightarrow (s, o, \texttt{write}) \notin b'$

Problema: nessuna restrizione è posta su T, che potrebbe essere sfruttata per far fluire informazione

System Z

Supponiamo T sia come segue:

- quando un soggetto richiede un qualsiasi accesso a un oggetto, il livello di sicurezza di tutti i soggetti e di tutti gli oggetti nel sistema è abbassato al livello più basso possibile e l'accesso è concesso.

È sicuro rispetto a BLP ma .... non è sicuro .....

BLP+Tranquility property Il livello di sicurezza non può essere cambiato.

Limitazioni della politica mandatoria

I requisiti del mondo reale spesso necessitano di eccezioni alle restrizioni della politica mandatoria. Es.,

declassificazione I dati possono poter essere declassati dopo un certo periodo

sanitizzazione Un processo può produrre dati meno sensibili di quelli a cui accede

$\Longrightarrow$ processi Trusted (fidati)

Per un processo trusted alcune restrizioni della politica mandatoria possono essere rilassate.

Limitazioni della politica mandatoria

La determinazione delle classi di accesso non è sempre facile

Associazione: L'associazione di un insieme di proprietà può avere classificazione maggiore delle proprietà singolarmente prese (es., *nome* and *stipendio*)

Aggregazione: Una aggregazione può avere una classificazione maggiore dei singoli valori che la compongono (es., la locazione di una *singola* nave militare è unclassified ma la locazione di *tutte* le navi di una flotta è secret).

Coesistenza di DAC and MAC

DAC and MAC non sono mutuamente esclusive

- Es., BLP applica anche una politica discrezionale
  DAC property $b \subseteq \{(s, o, a) \mid a \in M[s, o]\}$

Se sia DAC sia MAC sono applicate solo gli accessi che soddisfano entrambe sono permessi

DAC fornisce discrezionalità all'interno dei confini del MAC

Limitazioni delle politiche mandatorie

Le politiche mandatorie (anche con il principio di tranquillità) controllano solo canali aperti di informazione (flusso attraverso canali legittimi).

Sono vulnerabili a canali coperti.

I canali coperti sono canali non normalmente utilizzati per comunicare informazione ma che possono essere sfruttati a tale scopo.

Ogni risorsa o osservabile del sistema condivisa fra processi di diverso livello può essere sfruttata per stabilire un canale coperto.

Esempio di canali coperti e di tempo

- Un soggetto a basso livello chiede di scrivere in un file ad alto livello, che non esiste.
  - Il sistema ritorna errore (se il sistema crea un file l'utente non può essere informato di possibili errori).
- Un soggetto a basso livello richiede una risorsa (es., CPU o lock) che è occupata da un soggetto ad alto livello.
  - Il sistema non alloca la risorsa perchè occupata.
- Un processo ad alto livello può occupare risorse condivise e modificare i tempi di risposta a processi di basso livello (timing channel.

Meccanismi di locking e controllo della concorrenza devono essere ridisegnati in sistemi multilivello. (Attenzione ad evitare denial-of-service.)

Limitazioni delle politiche mandatorie

La analisi di canali coperti è solitamente fatta nella fase di implementazione (per assicurare che la implementazione non sia troppo debole).

Modelli di interfaccia cercano di eliminare la possibilità di canali coperti nella fase di modellizzazione.

- Non interferenza: attività di processi di alto livello non deve avere alcun effetto su processi a livelli inferiori o non comparabili.

Bisogna comunque sempre ricordarsi che dimostrazioni sul modello formale provano la sicurezza così come la abbiamo definita, non possono provare sicurezza assoluta.