

# modelli

# utenti e sistemi

- un sistema informatico viene tipicamente usato da molti utenti
  - non necessariamente ciascun utente ha una “utenza” (login name)
  - es. web server viene acceduto da utenti senza utenza specifica
- conflitti e convergenza di interessi
  - alcune risorse devono essere protette
  - altre risorse devono essere condivise

# confinamento (o isolamento)

- in tutti i sistemi le politiche di sicurezza impongono un confinamento
  - gli utenti (o i soggetti) possono operare solo su certe risorse e non su altre
- molti modelli sono stati sviluppati per esprimere politiche e meccanismi
- scopi
  - applicativi o teorici
  - espressione di politiche o configurazione di meccanismi

# AAA

- Authentication, Authorization, Accounting
  - IETF RFC 2903, anno 2000
  - orientato alle reti
  - radius è un “AAA protocol”
  - accounting: contabilizzazione del consumo delle risorse
  - **ACCOUNTING** ≠ gestione degli account
- il focus era sull’accesso a servizi di connettività, non su sicurezza
  - servizi dial-up, ADSL, telefonia, ecc.

# AAA e la sicurezza

- AAA è ora associato a sicurezza
- non solo reti, ma anche sistemi sw e sistemi operativi
- chi ne parla in ambito di sicurezza spesso confonde
  - Authentication, Authorization, ...
  - AUDITING: intendendo spesso logging
- infatti l'accounting non è molto rilevante in ambito sicurezza

# auditing

- auditing = controllo o verifica di “adeguatezza”
  - tipicamente manuale o semi-automatizzato
  - solo l’uomo ha l’adeguata flessibilità per adattarsi a variazioni di esigenze, strumenti, ambiente di lavoro, ecc.
- può essere fatto a vari livelli (varie accezioni diverse)
  - access auditing
  - log auditing
  - system security auditing
  - network security auditing
  - auditing di procedure (iso 27001)
  - auditing di competenze (di persone) (CISSP – SSCP e CISA – CISM)
  - ecc.

# AAA nei sistemi sw

- autenticazione
  - identifica l'utente
  - il sistema ha fiducia nell'utente e crea per lui un processo con adeguati diritti
- controllo di accesso e autorizzazione
  - l'accesso ad una risorsa può essere autorizzato o meno
  - se l'autorizzazione viene concessa l'accesso viene eseguito
- la richiesta d'accesso e il suo esito possono essere loggati

# autenticazione: trabocchetti

- ciascun utente umano dovrebbe essere rappresentato da un singolo identificatore
  - per identificatore si intende login name
  - altrimenti alcune politiche configurate sui sistemi possono venir invalidate
  - **richiede una policy esterna al sistema**
    - cioè non forzabile tecnologicamente



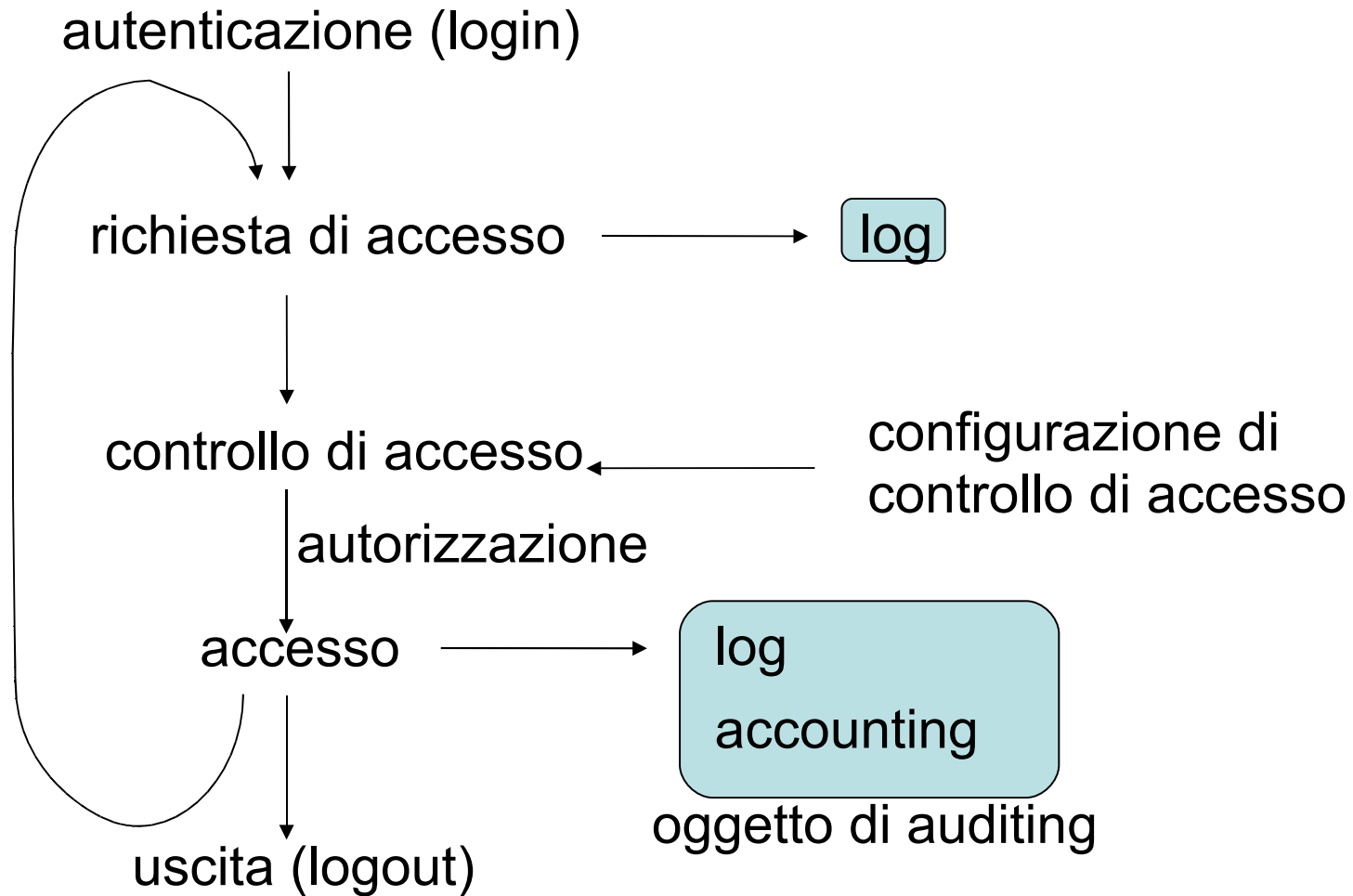
# esempi di trabocchetti

- la politica dice: se un utente viola la regola R allora il suo accesso ai sistemi deve essere revocato
  - il meccanismo usato per implementare la politica è la chiusura dell'account che ha violato R
  - se l'utente U ha due account U1 e U2 e viola R con U1 la chiusura di U1 non comporta la revoca dell'accesso ai sistemi poiché U ha ancora U2
- la politica dice: ciascun utente o può avere diritti per l'operazione op1 o l'operazione op2 ma non per entrambe.
  - l'implementazione di questa politica verifica che ciascun account non abbia la possibilità di fare entrambe le operazioni
  - se un utente U ha due account U1 e U2 dove U1 è abilitato a op1 e U2 è abilitato a op2, U ha in sostanza diritti per entrambe le operazioni in violazione della politica

# autenticazione: trabocchetti

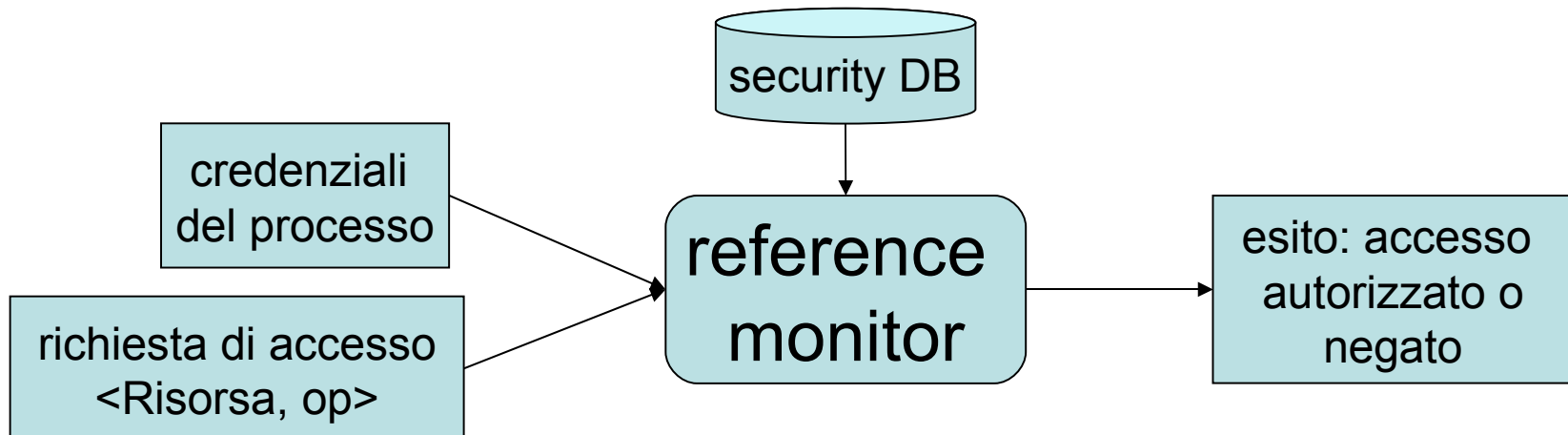
- ciascun identificatore d'utente dovrebbe far capo ad un singolo utente umano
  - necessaria per “strict accountability”
  - spesso violata perché i sistemi non posseggono adeguate funzionalità di condivisione (vedi principio di usabilità)
- esempio
  - alcuni utenti “privilegiati” conoscono la password di amministratore
  - dai log di sistema non è possibile risalire direttamente alla persona che ha compiuto una certa operazione ma solo al gruppo di persone che hanno la password di amministratore

# AAA nei sistemi sw



# reference monitor (security kernel)

- parte del s.o. che effettua il controllo di accesso
- caratteristiche:
  - invocato ad ogni richiesta di accesso
  - a prova di intrusione (nessuna vulnerabilità)
  - abbastanza piccolo da essere verificabile
- introdotto nel 1972 in James Anderson et al. - Computer Security Technology Planning Study
- richiesto in certe certificazioni: es. in TCSEC  $\geq$ B2



# reference monitor (security kernel)

- sistemi senza reference monitor
  - Windows 3.x, 95,98, Me (controllo di accesso limitato)
  - Linux (controllo di accesso efficace ma architettura senza r.m.)
- sistemi con reference monitor
  - Windows NT, 2000, XP, 2003, Vista
  - SELinux

# tipi di controllo di accesso

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)

# Discretionary Access Control

- gli utenti possono cambiare i diritti sulle risorse
  - es. il proprietario U1 di un file può dare, a sua discrezione, il diritto all'utente U2 di accedere a tale file
  - l'utente non accede direttamente al file, i suoi processi vi accedono
  - i diritti di accesso in realtà agiscono sui processi posseduti da U2
- implementazioni
  - Linux, Windows NT/2000/XP/2003/Vista

# Mandatory Access Control

- gli utenti **NON** possono cambiare i diritti sulle risorse
  - potrebbe non esistere il concetto di proprietario
  - il proprietario di una risorse è l'organizzazione (ad es. rappresentata dall'amministratore di sistema) e non un certo utente
- i diritti sono configurati dall'amministratore
- implementabile nei sistemi più diffusi
  - SELinux
  - TrustedBSD, SEBSD
  - TrustedSolaris
- Windows Vista (Windows Integrity Control, WIC)



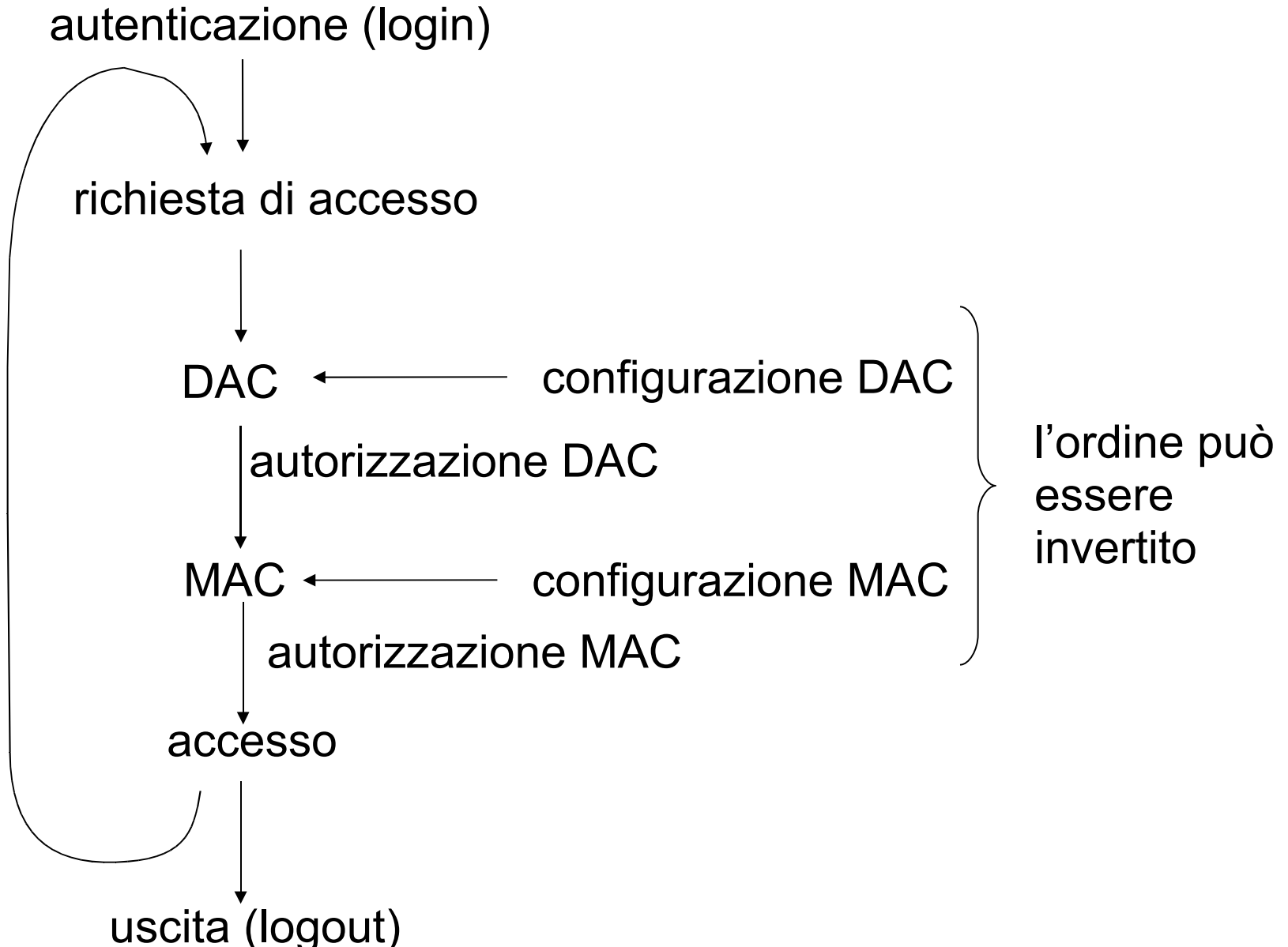
# DAC vs. MAC

- DAC il più diffuso
  - flessibile
  - sicurezza “delegata” agli utenti (tipicamente al proprietario della risorsa)
- MAC considerato il più sicuro
  - molto scomodo per gli utenti
    - se due utenti devono scambiare un file devono chiamare l'amministratore
  - usato molto in ambito militare

# MAC+DAC

- i sistemi con MAC tipicamente supportano anche DAC
- accesso consentito se sia i controlli mandatory che quelli discretionary danno autorizzazione
- permette di avere isole di “discrezionarietà” confinati da muri “mandatori”
  - es. il web server è separato dagli utenti da una configurazione MAC
  - ma MAC non isola gli utenti tra di loro, essi sono eventualmente isolati mediante DAC

# MAC+DAC



# Access matrix

# access matrix

- matrice che descrive i diritti di ciascun soggetto sugli oggetti o soggetti

objects (entities)

	$O_1$	...	$O_m$	$S_1$	...	$S_n$
$S_1$						
$S_2$						
...						
$S_n$						

subjects

- Subjects  $S = \{ s_1, \dots, s_n \}$
- Objects  $O = \{ o_1, \dots, o_m \}$
- Rights  $R = \{ r_1, \dots, r_k \}$
- Entries  $A[s_i, o_j] \subseteq R$
- $A[s_i, o_j] = \{ r_x, \dots, r_y \}$  means subject  $s_i$  has rights  $r_x, \dots, r_y$  over object  $o_j$

# i soggetti sono oggetti

- nella maggioranza dei casi i soggetti sono anche oggetti
  - cioè un soggetto può operare su altri soggetti (che svolgono il ruolo di oggetti)
  - es. un processo può essere killato

# usi della access matrix

- per esprimere politiche
- per esprimere stati di un sistema

# protection state

- è la parte dello stato di un sistema che è rilevante per la sicurezza
- una transizione di stato è un cambiamento della access matrix
  - creazione e distruzione di oggetti
    - aggiunge o cancella colonne
  - creazione e distruzione di soggetti
    - aggiunge o cancella righe e colonne
  - inserimento di un diritto in una cella
  - cancellazione di un diritto in una cella
- le transizioni di stato possono essere vincolate da particolari diritti nella matrice stessa
  - il diritto di creazione è eventualmente associato a un soggetto e non ad una cella della matrice



# rappresentazione del protection state

- **access control list**
  - ciascun oggetto ha associato una struttura dati che esprime quali soggetti possono agire sull'oggetto stesso e con che operazioni
- **capabilities**
  - ciascun soggetto ha associato una struttura dati che esprime su quali oggetti può agire e con che operazioni
- **sono modi per rappresentare una matrice sparsa**

# ownership

- $s_1$  own  $o_1$ , cioè  $\text{own} \in A[s_1, o_1]$
- $s_1$  ha il diritto di modificare a piacimento l'elemento  $A[s_1, o_1]$
- più che un diritto su  $o_1$  è un diritto su  $A[s_1, o_1]$
- vincoli tipici
  - owner non modificabile
  - un solo owner
    - come in unix e windows
    - ownership trasferibile (unix: `chown`, win: “take ownership”)
  - spesso implica il diritto grant

# grant e attenuazione del privilegio

- $s_1$  grant  $o_1$
- $s_1$  ha il diritto di modificare gli elementi della colonna  $A[ s, o_1 ]$  con  $s \neq s_1$ 
  - cioè dare ad altri soggetti diritti su  $o_1$
- più che un diritto su  $o_1$  è un diritto su  $A[ . , o_1 ]$
- vincolo tipico (principio di attenuazione del privilegio)
  - $s_1$  non può dare ad altri diritti che lui stesso non possiede

# access matrix, DAC e MAC

- una matrice di accesso può rappresentare politiche DAC o MAC
- la presenza di diritti grant rendono la politica DAC

# esempio

- Processes  $p, q$  soggetti
- Files  $f, g$  oggetti
- Rights *Read, Write, eXecute, Append, Own+grant*

	$f$	$g$	$p$	$q$
$p$	$rwo$	$r$	$rwXO$	$w$
$q$	$a$	$ro$	$r$	$rwXO$

- politica DAC

# esempio

- Procedures *inc\_ctr*, *dec\_ctr*, *manage*    soggetti
- Variable *counter*    oggetti
- Rights +, -, *call*

	<i>counter</i>	<i>inc_ctr</i>	<i>dec_ctr</i>	<i>manage</i>
<i>inc_ctr</i>	+			
<i>dec_ctr</i>	-			
<i>manage</i>		<i>call</i>	<i>call</i>	<i>call</i>

- politica MAC