Linguaggi per la specifica di autorizzazioni

Modelli DAC più recenti cercano di includere almeno

- autorizzazioni positive e negative
- propagazione basata sulla gerarchia
- politiche per la risoluzione dei conflitti e politiche di decisione
- addizionali regole di implicazione (es., se Bob può fare una cosa anche Alice la può fare)

Goal Essere flessibile e permettere di supportare politiche differenti

- Utenti/amministratori differenti possono avere differenti requisiti di protezione
- Uno stesso utente/amministratore può avere requisiti di protezione diversi su diversi oggetti
- I requisiti di protezione possono variare nel tempo

Linguaggi di autorizzazione basati sulla logica

Alcuni approcci hanno proposto l'utilizzo di linguaggi logici.

Vantaggio: maggiore espressività e flessibilità

.... ma dobbiamo stare attenti a

- bilanciare flessibilità/espressività rispetto a performance
- non perdere controllo sulle specifiche di autorizzazione
- garantire il comportamento delle specifiche (ricordiamoci che stiamo parlando di sicurezza)

Alcune proposte permettono interpretazioni multiple

$\Longrightarrow$ semantica ambigua delle specifiche di sicurezza

Esempio di linguaggio logico per autorizzazioni

cando(o,s,⟨*sign*⟩a) rappresenta le autorizzazioni.

dercando(o,s,⟨*sign*⟩a) rappresenta le autorizzazioni derivate

do(o,s,⟨*sign*⟩a) rappresenta gli accessi che devono essere permessi o negati.

done(o,s,r,a,t) storia degli accessi.

error segnala errori.

hie-predicates: predicati gerarchici

rel-predicates: predicati specifici dell'applicazione (es., $\texttt{owner}(\texttt{user}, \texttt{object})$, $\texttt{supervisor}(\texttt{user1}, \texttt{user2})$).

Esempio di regole FAF

$$\begin{aligned}
\texttt{cando}(\texttt{file1}, s, \texttt{+read}) &\leftarrow \texttt{in}(s, \texttt{Employees}) \;\&\; \neg\texttt{in}(s, \texttt{Soft-Developers}).\\
\texttt{cando}(\texttt{file2}, s, \texttt{+read}) &\leftarrow \texttt{in}(s, \texttt{Employees}) \;\&\; \texttt{in}(s, \texttt{Non-citizens}).
\end{aligned}$$

$$\begin{aligned}
\texttt{dercando}(\texttt{file1}, s, -\texttt{write}) &\leftarrow \texttt{dercando}(\texttt{file2}, s, +\texttt{read}).\\
\texttt{dercando}(o, s, -\texttt{grade}) &\leftarrow \texttt{done}(o, s, r, \texttt{write}, t) \& \texttt{in}(o, \texttt{Exams}).\\
\texttt{dercando}(\texttt{file1}, s, -\texttt{read}) &\leftarrow \texttt{dercando}(\texttt{file2}, s', +\texttt{read}) \& \texttt{in}(s, g) \& \texttt{in}(s', g).
\end{aligned}$$

Esempio di regole FAF – 2

$$
\begin{aligned}
\texttt{do}(\texttt{file1}, s, +a) &\leftarrow \texttt{dercando}(\texttt{file1}, s, +a).\\
\texttt{do}(\texttt{file2}, s, +a) &\leftarrow \neg\texttt{dercando}(\texttt{file2}, s, -a).\\
\texttt{do}(o, s, +\texttt{read}) &\leftarrow \neg\texttt{dercando}(o, s, +\texttt{read}) \,\&\\
&\quad\ \neg\texttt{dercando}(o, s, -\texttt{read}) \,\&\\
&\quad\ \texttt{in}(o, \texttt{Pblc-docs}).
\end{aligned}
$$

$$
\begin{aligned}
\texttt{error} &\leftarrow \texttt{in}(s, \texttt{Citizens}) \,\&\, \texttt{in}(s, \texttt{Non-citizens}).\\
\texttt{error} &\leftarrow \texttt{do}(o, s, +\texttt{write}) \,\&\, \texttt{do}(o, s, +\texttt{evaluate}) \,\&\\
&\quad\ \texttt{in}(o, \texttt{Tech} - \texttt{reports}).
\end{aligned}
$$

Modelli di autorizzazione per sistemi OO – 1

Controllo dell'accesso in sistemi ad oggetti sfrutta la proprietà di

- incapsulazione: l'accesso agli oggetti è sempre effettuato tramite metodi (procedure)

Può essere dato il permesso agli utenti di eseguire procedure senza necessariamente avere l'autorizzazione per tutte le azioni che la procedura compie, che possono essere controllate rispetto a

- proprietario del metodo (es., CACL)
- colui che ha garantito le autorizzazioni (OSQL)
- il codice che richiede l'accesso (Java)

Modelli di autorizzazione per sistemi OO – 2

L'incapsulazione in sistemi OO è anche stata sfruttata per applicare restrizioni di flusso.

Interpretazione discrezionale del principio no-write-down: l'informazione può fluire da un oggetto o ad un oggetto o' solo se gli utenti autorizzati a leggere o' sono un sottoinsieme di quelli autorizzati a leggere o

- troppo restrittiva

In sistemi orientati ad oggetti:

- posso controllare il flusso intercettando lo scambio di messaggi (per chiamate di metodi)
- posso specificare eccezioni alle restrizioni
  - l'informazione ritornata da un metodo (fidato) può essere rilasciata ad utenti che non sono autorizzati a leggere l'informazione che il metodo ha acceduto

Chinese wall

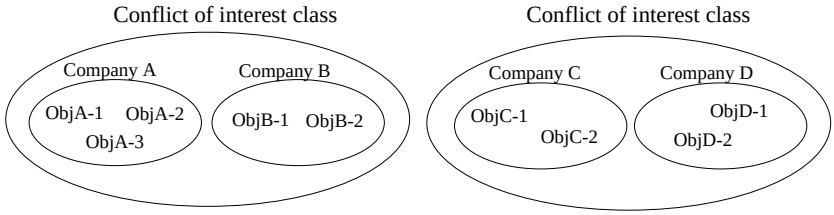
Tipo particolare di politica di stile mandatorio che applica separazione dinamica dei privilegi per proteggere segretezza in ambito commerciale

Goal prevenire flussi di informazione che causano conflitto di interesse per singoli consulenti (es. un consulente non dovrebbe avere informazioni su due banche o due compagnie petrolifere)

Chinese wall

Gli oggetti sono organizzati gerarchicamente su tre livelli

- basic object oggetti contenenti informazioni, (e.g., file), ognuno riguarda una diversa organizzazione;
- company dataset gruppi di oggetti che si riferiscono alla stessa organizzazione;
- classi di conflitto di interesse raggruppano tutti i dataset di organizzazioni che sono in competizione



Chinese wall – assiomi

Simple security rule Un soggetto s può avere accesso a un oggetto o solo se o:

- è nello stesso company dataset ("all'interno del muro") degli oggetti che s ha già acceduto, oppure
- appartiene a una classe di conflitto di interessi differente.

***-property** Accesso in scrittura è permesso solo se

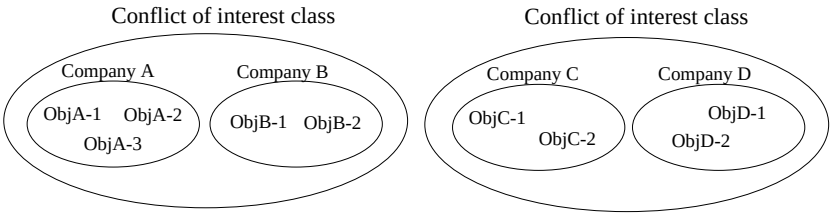
- l'accesso è permesso dalla simple security rule, e
- nessun oggetto può essere letto che è *i)* in un company dataset differente da quello per cui l'accesso in scrittura è richiesto, e *ii)* contiene informazione "non sanitizzata".

Chinese Wall

La simple security rule blocca flussi da parte di un singolo utente.

La *-property blocca flussi indiretti che potrebbero essere effettuati con la collusione di due o più utenti

Es., Alice legge ObjA-1 e scrive ObjC-1. Bob legge ObjC-1 e scrive ObjB-1.



Chinese Wall

La politica Chinese Wall non è ben formalizzata e lascia aperti alcuni problemi, quali

- gestire la storia degli accessi
- garantire accessibilità (es., se tutti gli utenti leggono lo stesso dataset il sistema è bloccato)
- sanitizzazione dei dati

Però introduce il concetto importante di separazione dei privilegi dinamica

Separazione dei privilegi

Principio di separazione dei privilegi: nessun utente (o insieme ristretto di utenti) dovrebbe avere abbastanza privilegi da poter abusare del sistema.

statica chi specifica le autorizzazioni deve assegnarle in modo da non dare "troppi privilegi" ad un singolo utente

dinamica il controllo sulla limitazione dei privilegi è dinamico: un utente non può fare "troppi" accessi ma è libero di scegliere quali fare. Il sistema gli negherà gli altri di conseguenza $\Longrightarrow$ più flessibile

Esempio: fare-ordine, spedire-ordine, registrare-fattura, pagare. Abbiamo quattro persone in segreteria. Requisito di protezione:
almeno due utenti devono essere coinvolti nel processo

statico l'amministratore assegna i compiti in modo che nessuno abbia tutte e quattro le operazioni.

dinamico ogni persona può fare qualsiasi operazione, ma non può completare il processo $\Longrightarrow$ se ne fa tre il sistema gli rifiuta la quarta.

Controlli di integrità

Il modello di Biba per la protezione dell'integrità non è sufficiente

- le restrizioni sul flusso dell'informazione a volte sono troppo restrittivi
- controlla solo compromessi di integrità dovuti a flussi verso classificazioni maggiori $\Longrightarrow$ cattura solo una parte del problema dell'integrità

Integrità

Integrità: assicurare che nessuna risorsa (inclusi dati e programmi) sia modificata in modo non autorizzato o improprio e che i dati memorizzati nel sistema riflettano il mondo reale che devono rappresentare

$\Longrightarrow$ prevenire frodi e errori

Ogni sistema di gestione dei dati ha funzionalità per garantire l'integrità

- controlli di concorrenza: per assicurare che l'esecuzione concorrente di accessi non porti a perdita di dati o a inconsistenze.
- tecniche di recovery: per ricostruire lo stato del sistema in caso di errori o violazioni
- vincoli di integrità: costringono i valori che i dati possono assumere

Integrità

Transazione: sequenza di azioni che deve assicurare le proprietà ACIDE

- Atomicità: deve essere eseguita completamente o non eseguita affatto (viene abortita)
- Consistenza: deve preservare la consistenza della base di dati
- Isolamento non deve rendere visibili le sue modifiche fino a che non completa con un commit
- Durabilità le sue modifiche, una volta che la transazione è terminata con successo, non possono essere perse per fallimenti che avvengono in seguito

Non basta! non prendono in considerazione il soggetto che esegue le azioni. Non proteggono da abusi da parte di utenti legittimi.

Clark e Wilson

Modello di Clark e Wilson definisce quattro criteri di base per salvaguardare l'integrità:

1. Autenticazione. L' identità di tutti gli utenti che accedono al sistema deve essere autenticata.
2. Audit. Tutte le modifiche devono essere registrate, cosìda poter ripristinare lo stato precedente ad esse.
3. Well-formed transaction Gli utenti non possono manipolare i dati arbitrariamente ma solo attraverso procedue che ne assicurino l'integrità (es., partita doppia in sistemi di gestione). Transazioni well-formed dovrebbero provvedere serializzabilità, recovery e gestione della concorrenza.
4. Separazione dei privilegi Il sistema deve associare ad ogni utente un insieme di programmi che questo può eseguire. I privilegi dati ad un utente devono soddisfare il principio di separazione dei privilegi.

Clark e Wilson

Basato sui seguenti concetti

- Constrained Data Item CDI sono oggetti la cui identità deve essere salvaguardata
- Unconstrained Data Item UDI sono oggetti che non sono coperti dalla politica di integrità (es., informazione immessa da tastiera)
- Integrity Verification Procedure. IVP procedure che verificano la validità di uno stato rispetto alle specifiche di integrità
- Transformation Procedure. TP sono le sole procedure (well-formed) che possono modificare i CDI o prendere input arbitrario dagli utenti e creare CDI. Preservano l'integrità.

Clark e Wilson

- Tutte le IVP devono assicurare che i CDI sono in uno stato valido quando sono eseguite
- Tutte TP devono essere certificate valide (preservare la validità dello stato)
- Assegnamento dei TP agli utenti deve soddisfare il principio della separazione dei privilegi
- Le operazioni delle TP devono essere registrate
- L'esecuzione delle TP sugli UDI deve risultare in CDI validi
- Solo TP certificate devono manipolare CDI
- Gli utenti possono accedere CDI solo attraverso le TP per le quali hanno autorizzazione
- L'identità di ogni utente deve essere autenticata

Controllo dell'accesso basato sui ruoli

Osservazione Il completamento di certe attività richiede la possibilità di eseguire un insieme di privilegi.

Garantire e concedere tali privilegi sulla base dell'identità degli utenti può essere pesante

Sfruttare i concetti di application/task.

Ruolo: insieme di azioni e responsabilità legate a una particolare attività lavorativa. Può essere generico, e riflettere un tipo di lavoro (es., segretario, direttore, ...) o specifico, riflettendo un compito (es., registrazione-fatture).

Accesso degli utenti agli oggetti è mediato da ruoli

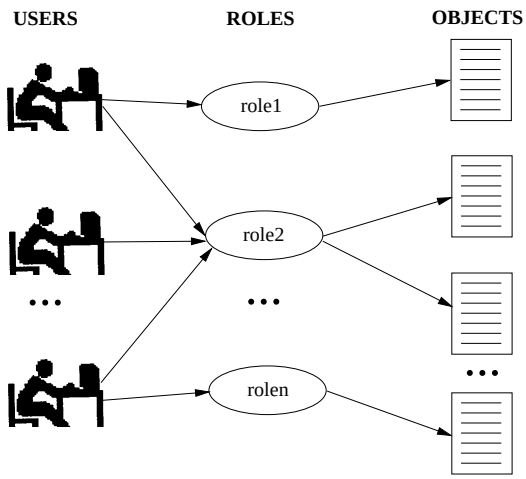
RBAC

- Ruoli sono autorizzati ad accedere agli oggetti
  Es., ⟨Direttore,scrivere,verbale ⟩
  ⟨Professore-a-lezione,usare,proiettore ⟩
- Utenti sono autorizzati ad attivare ruoli
  Es., ⟨Rossi,Professore-a-lezione⟩
- Attivando un ruolo r un utente può eseguire tutti gli accessi concessi ad r.
- I privilegi di un ruolo non sono applicabili quando il ruolo non è attivo.

Anche se ruoli e gruppi possono rappresentare lo stesso concetto, sono due cose diverse

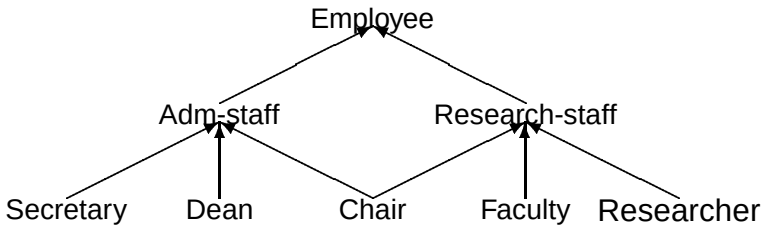
- gruppi: raggruppano utenti (“statici”)
- ruoli: raggruppano privilegi (“dinamici”)

RBAC



RBAC

Generalmente i ruoli sono organizzati in una gerarchia di specializzazione



La gerarchia può essere sfruttata per propagare autorizzazioni

- Se un ruolo r ha l'autorizzazione ad eseguire (azione,oggetto) $\Longrightarrow$ tutti i ruoli specializzazione di r possono eseguire (azione, oggetto)
- Se u ha l'autorizzazione ad attivare un ruolo $r \Longrightarrow u$ può attivare tutti i ruoli generalizzazione di r

RBAC – Vantaggi

Gestione semplificata è più facile gestire le autorizzazioni (es., è sufficiente assegnare o togliere un ruolo ad un utente per abilitarlo a tutta una serie di compiti)

Gerarchia di ruoli può essere sfruttare per le implicazioni. Semplifica ulteriormente la gestione.

Restrizioni Ai ruoli possono essere associate restrizioni quali cardinalità o restrizioni di attivazione.

Least privilege Permettono di avere in ogni momento il minimo privilegio possibile per fare un certo lavoro. $\Longrightarrow$ Limita la possibilità di abusi e danni per violazioni.

Separazione dei privilegi Permettono di applicare una politica di separazione dei privilegi.

RBAC

Controllo di accesso basato sui ruoli promettente ma non soddisfa tutti i requisiti del mondo reale

- relazioni gerarchiche possono essere limitative (es., segretario può scrivere lettere on behalf del suo manager)
- propagazione basata sulla gerarchia non sempre voluta (è contraria al principio del minimo privilegio).
- servono politiche amministrative
  - proprietà non può essere più data all'utente
- mancanza di relazioni con l'identità degli utenti (che a volte può servire per raffinare gli accessi – es., "mio paziente")

Named protection domain

Introdotti per semplificare la gestione dei privilegi nel contesto SQL.

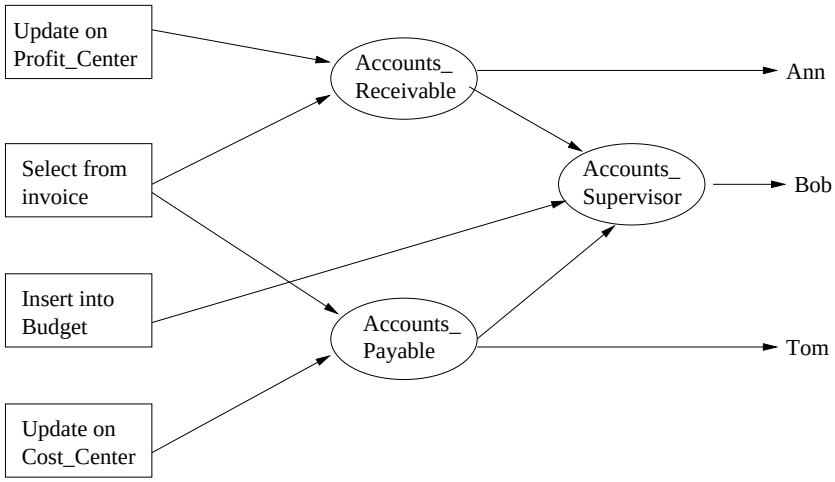
named protection domain identifica un insieme di privilegi (quelli garantiti all' NPD) che servono per eseguire un determinato compito.

Es., in una banca `Accounts_Receivable` può avere tutti i privilegi che servono per eseguire il determinato compito legato agli account.

NPD possono essere garantiti a NPD o ad altri utenti (catena di grant).

Gli NPD possono essere garantiti con la "grant option" (il ricevente può garantirli ad altri).

Named protection domain



Named protection domain

Gli utenti possono accedere agli oggetti solo attivando NPD che hanno privilegi di accesso.

Gli utenti possono attivare solo NPD per i quali sono (direttamente o indirettamente autorizzati).

Gli utenti possono attivare solo un NPD alla volta (minimo privilegio).

NPD correspondono ai ruoli in SQL:1999.

Controllo degli accessi in sistemi aperti

Siamo partiti dall'assunzione che l'autenticazione è prerequisito per il controllo dell'accesso. Oggi non sempre si può applicare

- richiesta di transazioni anonime
- in un sistema aperto come internet utenti nuovi (non conosciuti al server) possono presentare richieste di accesso
  - chiedere ad un server di mantenere informazioni su tutti gli utenti può non essere gestibile

$\Longrightarrow$ controllo dell'accesso basato su credenziali (certificati digitali)

Controllo dell'accesso basato su credenziali

Un utente può presentare certificati digitali per certificare

- identità
- accreditamenti (es., appartenenza ad associazioni o gruppi)
- la sua chiave

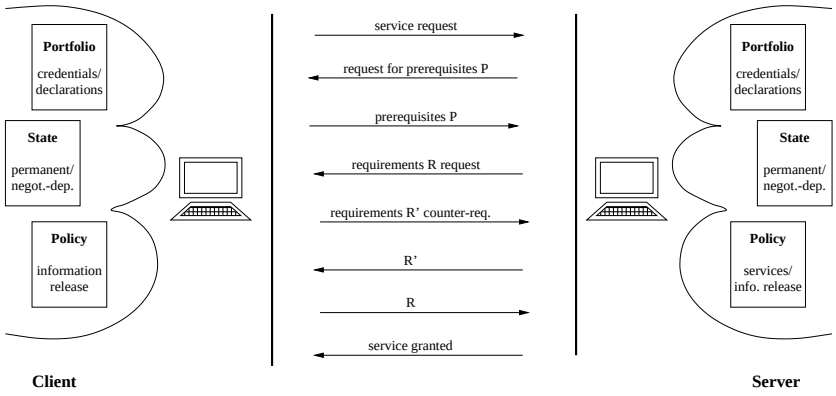
Il server può sfruttare i certificati per decidere se dare o meno l'accesso

Una ulteriore complicazione: il server deve chiedere all'utente

$\Longrightarrow$ le richieste devono garantire la privatezza della politica del server

- Sistemi di autorizzazione al lato server indicano le restrizioni di accesso e come comunicarle al client
- sistemi di autorizzazione al lato client restringono le informazioni e le credenziali che questi può rilasciare

Esempio di negoziazione



Sistemi emergenti per il controllo dell'accesso

Utilizzo di XML per la specifica di autorizzazioni

- XACML (OASIS eXtensible Access Control Markup Language)
  - Linguaggio basato su XML per la specifica di autorizzazione
  - in corso di definizione all'interno di OASIS (Organization for the Advancement of Structured Information Standards)
  - disegnato all'interno di SAML per la gestione delle proprietà degli utenti
  - Cerca di bilanciare interoperabilità e espressività con requisiti di applicabilità ed efficienza
    - richieste sono presentate con asserzioni SAML che certificano le proprietà degli utenti
    - soggetti e oggetti di autorizzazione definiti tramite formule booleane di predicati