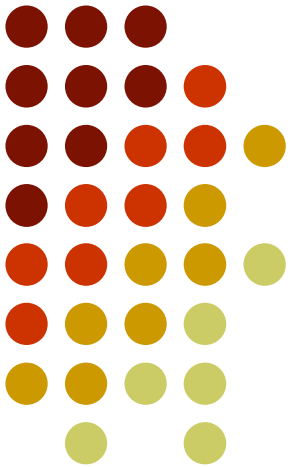
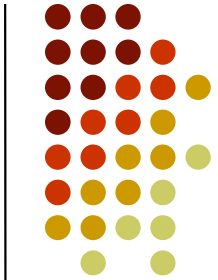
Privacy e Sicurezza dei Dati

Elena Ferrari

DiSTA, Università dell'Insubria
elena.ferrari@uninsubria.it
http://www.dista.uninsubria.it/~elena.ferrari

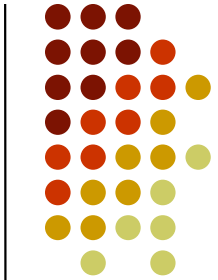


Obiettivo corso



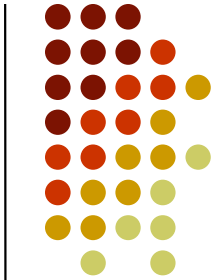
- Apprendere i concetti fondamentali connessi alla protezione dei dati con un particolare focus ai meccanismi di controllo dell'accesso e alla privacy … senza trascurare le più innovative linee di ricerca

Programma del corso



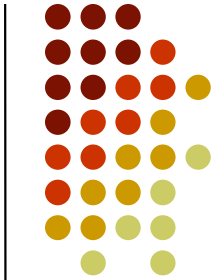
- Sicurezza/privacy nei sistemi di gestione dati: concetti introduttivi
- Controllo dell'accesso nei DBMS:
  - Concetti base;
  - Modelli di controllo dell'accesso tradizionali: DAC, MAC, RBAC;
  - Modelli di controllo dell'accesso innovativi: content-based, time-based, location-based, ecc.
  - Controllo dell'accesso in SQL;
  - Oracle VPD, OLS, Oracle Vault.

Programma del corso



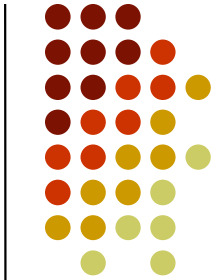
- Protezione della privacy nei DBMS:
  - Introduzione al concetto di privacy e alla legislazione associata;
  - Controllo dell'accesso purpose-based;
  - IBM Hyppocratic DBMS;

Programma del corso



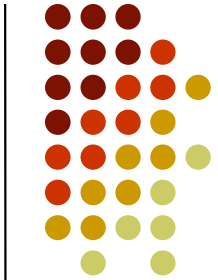
- Controllo dell'accesso e privacy, linee di tendenza:
  - data outsourcing;
  - trust e privacy in reti sociali e social web;
  - gestione delle emergenze;
  - privacy assessment.

Libri consigliati



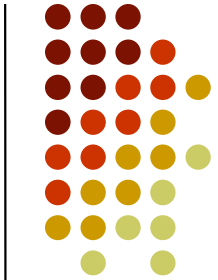
- Per la parte di controllo dell'accesso
  - E. Ferrari. Access Control in Data Management Systems, Synthesis Lectures on Data Management, Morgan & Claypool, 2010.
  - B. Catania, E. Ferrari, e G. Guerrini. Sistemi di Gestione Dati: Concetti e Architetture, CittàStudi Edizioni, 2006 (limitatamente al capitolo sulla protezione dei dati)
- Più lucidi su moodle e materiale di approfondimento consigliato durante il corso.

Orari



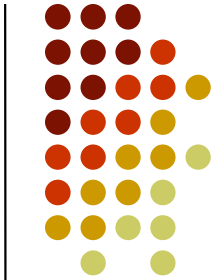
- Mercoledì: 14/18 aula 2TM
- Giovedì: 11/12.30 aula 2 TM

Organizzazione corso



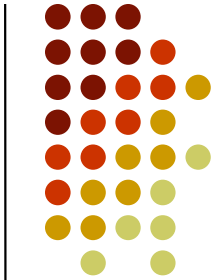
- Il corso è 9 crediti:
  - 6 CFU didattica frontale (DF) – Prof.ssa Elena Ferrari
  - 3 CFU minicorso Big Data Analytics (LAB) – Dott. Cuneyt Akcora
    - Si veda: https://class.coursera.org/sna-2012-001/class/index
    - Calendario: 31 Ott., 7/14 Nov., 5/19 Dic, 9 Gen.

Modalità d'esame

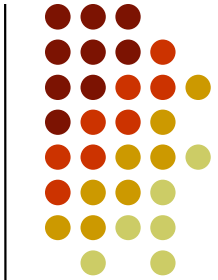


- Prova DF:
  - scritto suddiviso in parte A e parte B:
    - parte A: domande di teoria
    - parte B: esercizi
    - voto finale: 1/3 parte A + 2/3 parte B
    - il voto della parte A e B deve essere >= 18 nella stessa sessione
    - non è possibile la consultazione di libri/appunti durante l'esame

Modalità d'esame

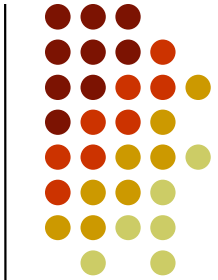


- Prova LAB:
  - progettino
- Voto: 2/3 DF * 1/3 LAB

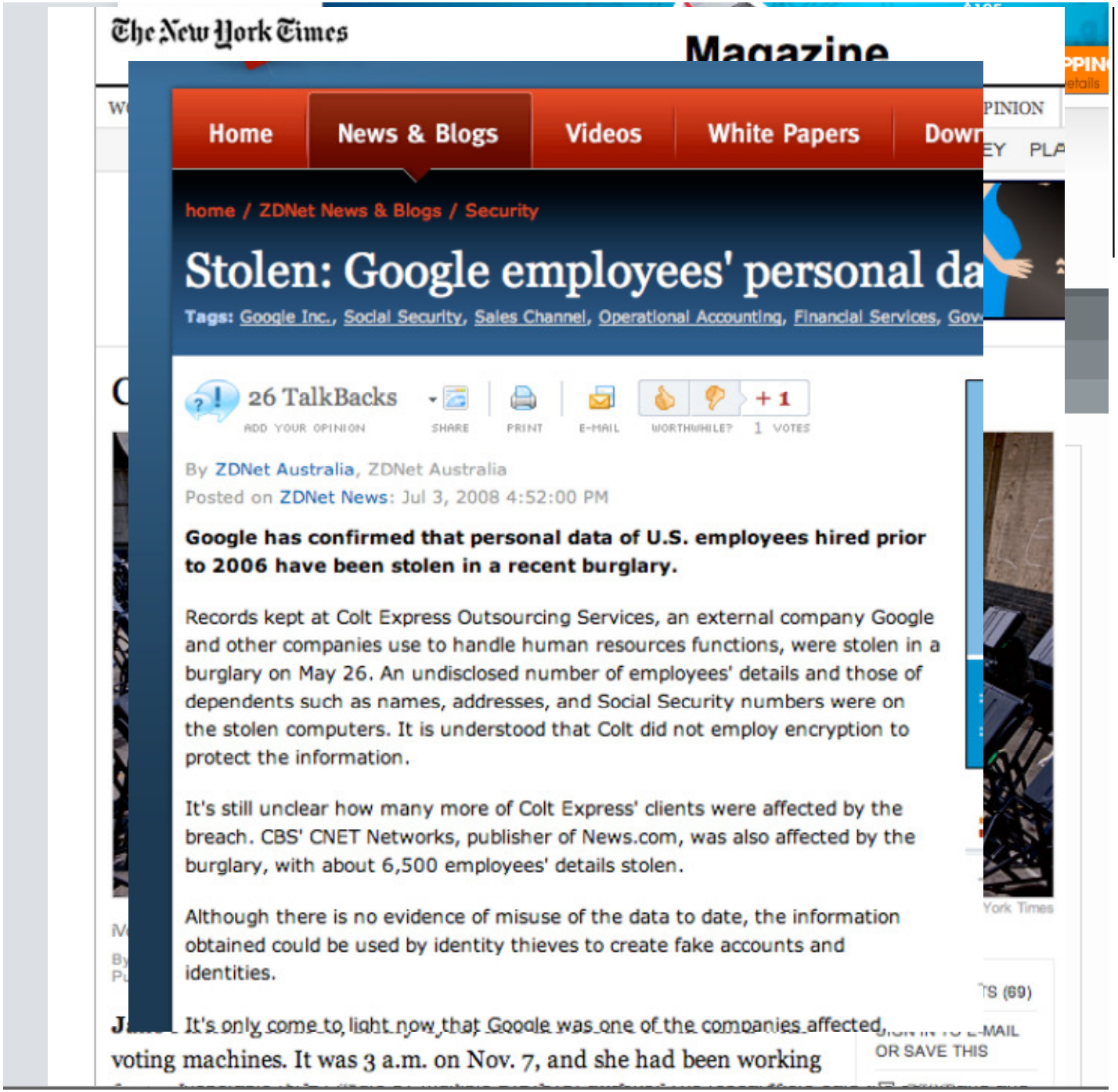


SICUREZZA/PRIVACY DEI SISTEMI DI GESTIONE DATI: CONCETTI INTRODUTTIVI

Protezione dati – perchè?



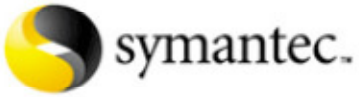
- La protezione delle informazioni è un'aspetto di importanza strategica per ogni organizzazione
- Un danno al patrimonio informativo si ripercuote non solo sul singolo utente/applicazione ma può avere un impatto notevole sull'intera organizzazione
- L'avvento di Internet e, più recentemente, delle tecnologie del Web 2.0 ha reso ancora più evidente questo problema


The New York Times
Magazine
Home
News & Blogs
Videos
White Papers
home / ZDNet News & Blogs / Security
Stolen: Google employees' personal da
Tags: Google Inc., Social Security, Sales Channel, Operational Accounting, Financial Services, Gov
26 TalkBacks
ADD YOUR OPINION
SHARE
PRINT
E-MAIL
WORTHWHILE?
+1
1 VOTES
By ZDNet Australia, ZDNet Australia
Posted on ZDNet News: Jul 3, 2008 4:52:00 PM
Google has confirmed that personal data of U.S. employees hired prior to 2006 have been stolen in a recent burglary.
Records kept at Colt Express Outsourcing Services, an external company Google and other companies use to handle human resources functions, were stolen in a burglary on May 26. An undisclosed number of employees' details and those of dependents such as names, addresses, and Social Security numbers were on the stolen computers. It is understood that Colt did not employ encryption to protect the information.
It's still unclear how many more of Colt Express' clients were affected by the breach. CBS' CNET Networks, publisher of News.com, was also affected by the burglary, with about 6,500 employees' details stolen.
Although there is no evidence of misuse of the data to date, the information obtained could be used by identity thieves to create fake accounts and identities.
It's only come to light now that Google was one of the companies affected
voting machines. It was 3 a.m. on Nov. 7, and she had been working

Cyber Attacks Against SCADA and Control Systems - Real World Trends and Real World Solutions

Featuring: Eric Byres and Alan Paller

Sponsored by:



Username (email):

Password:

share my info with sponsor

click here to proceed

FAQ For Trouble Shooting

You need to register with the SANS portal to be able to sign in.

Webcast Overview:

Cyber Attacks Against SCADA and Control Systems - Real World Trends and Real World Solutions

Featuring: Eric Byres, Alan Paller, Bryan Giovanni Geraldo

Over the past few years the popular press has been filled with stories of terrorists using the net to attack SCADA and control systems and bring our nation to it knees. Yet the lights keep burning and the factories keep running - is the need to secure SCADA systems a myth?

Based on a careful statistical analysis of validated control system incidents at 22 major corporations, the answer is that the need to secure SCADA and Control Systems is no myth. In fact, the incidents are far more widespread than commonly believed, the targets more wide ranging and attackers are not who we think they are. Even more ominous, the data shows that getting into most control systems is surprisingly easy. The good news is data also shows that there are effective solutions for SCADA systems security. The webcast will close with a discussion of these practical and cost effective measures, particularly with respect to policy creation and management for new industry regulations like NERC CIP -002-009.

Speaker Bios:

» Corriere della Sera › Esteri › *Piratata la pagina Facebook di Sarkozy*

Share Tweet 6 Consiglia 53

DISAVVENTURA INFORMATICA PER IL CAPO DELLO STATO FRANCESE

Piratata la pagina Facebook di Sarkozy

Il falso messaggio, poi cancellato: «Alle presidenziali non mi ricandido». Il presidente: «Nessun sistema è sicuro»

Il post scritto da Sarkozy su Facebook dopo la cancellazione del falso messaggio postato in bacheca

MILANO - Disavventura informatica per Nicolas Sarkozy. La pagina ufficiale su Facebook del presidente francese è stata hackerata e sulla bacheca del social network è stato postato un falso messaggio, in cui il capo dello Stato affermava di «non volersi ripresentare alle prossime elezioni presidenziali del 2012».

IL FALSO MESSAGGIO - «Cari compatrioti - recitava il post "piratato" -, tenuto conto delle circostanze eccezionali che vive il nostro Paese, ho deciso in animo e coscienza di non ripresentarmi al termine del mandato nel 2012. Per spiegarvi questo gesto, vi invito a una grande manifestazione popolare». Un messaggio pieno di errori ortografici, in cui l'autore inseriva anche un link a un'altra pagina di Facebook. Nella pagina «Brindisi per l'addio di Nicolas Sarkozy», l'amministratore dava appuntamento a «domenica 6 maggio 2012 tra le 19 e le 23.30 davanti al caffè Le Fouquet's di Parigi», ristorante sugli Champs-Elysées dove Sarkò ha festeggiato la vittoria nel 2007.

IL VERO POST DEL PRESIDENTE - Dopo la cancellazione del falso messaggio è stato inserita una dichiarazione di Sarkozy. «La mia pagina di Facebook - conferma il presidente francese - è stata piratata per ricordarmi forse che nessun sistema è infallibile». «Prendo nota della lezione di scrittura e di ortografia - si legge ancora nel post del capo di Stato - ma non sottoscrivo le conclusioni affrettate del messaggio».

Redazione online
24 gennaio 2011

© RIPRODUZIONE RISERVATA

Sicurezza Facebook – Hackerato il profilo di Mark Zuckerberg!

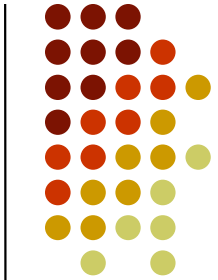
Pubblicato il gennaio 26, 2011 da **strano**

Mark Zuckerberg: pochi non lo conoscono. È l'inventore di Facebook. Ma il suo profilo non era a prova di hacker! **Un hacker se ne è infatti impossessato lasciando un messaggio un po'....strano** 😀

Il messaggio ha lasciato un po' interdetti tutti quanti...

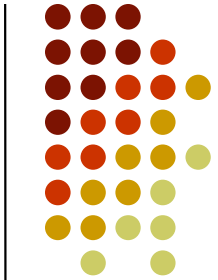
> *Che l'hacking abbia inizio: Se Facebook ha bisogno di soldi, invece di andare dalle banche, perché non permette ai suoi utenti di investirvi in modo sociale? Perché non trasformare Facebook in un "social business", nel modo in cui lo ha descritto il vincitore del premio Nobel Muhammed Yunus? Che ne pensate?*

Alcuni dati



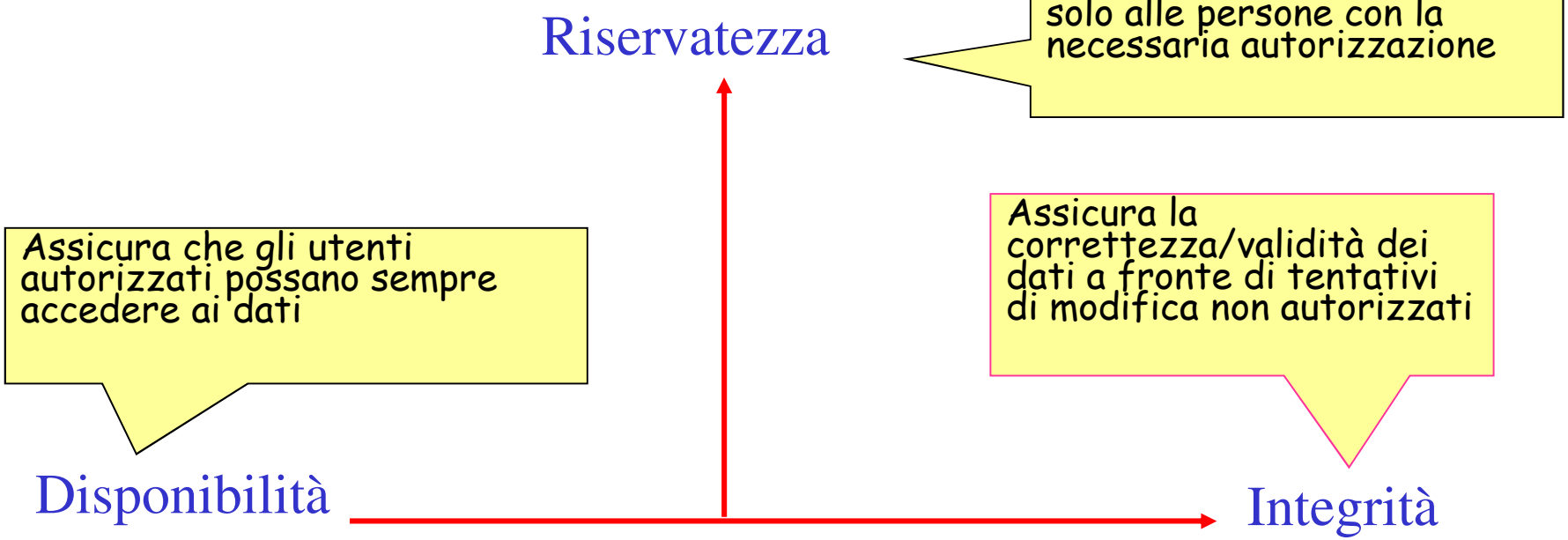
- 2011 CyberSecurity Watch Survey (*) (CSO Magazine in collaborazione con US Secret Service, CMU CERT e Deloitte)
- Sulla base delle risposte ai questionari:
  - 21% degli attacchi provengono dall'interno (58% dall'esterno, 21% non si sa)
  - Gli attacchi più frequenti sono:
    - Accesso non autorizzato a dati, sistema, reti 23%
    - Furto della proprietà intelletuale 16%
    - *Involontaria rivelazione di informazioni private/sensibili 29%*
  - (*) http://www.cert.org/insider_threat/

Attacchi alla sicurezza

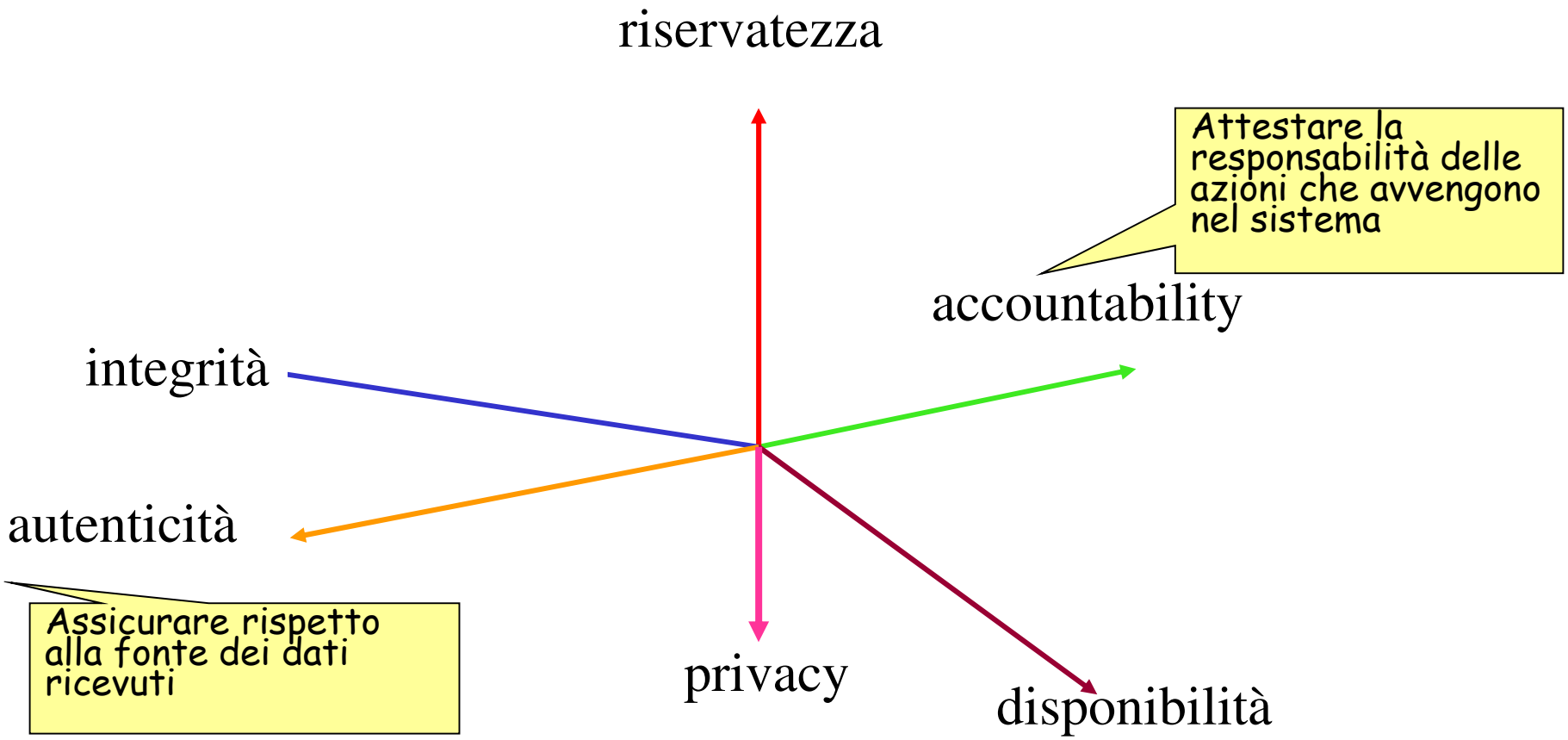


- Possono essere classificati in:
  - **Attacchi passivi:**
    - accedere ai dati di un sistema informativo senza modificarli
    - monitorare o ascoltare la trasmissione di messaggi per:
      - Intercettazione del contenuto del messaggio
      - Analisi del traffico
  - **Attacchi attivi:**
    - accedere e modificare i dati di un sistema informativo
    - modificare il flusso dei dati per:
      - mascheramento/replicazione
      - modificare un msg durante la trasmissione
      - Denial Of Service (DOS)

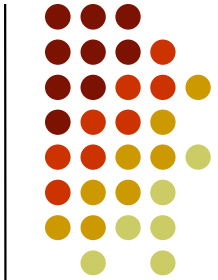
Requisiti base di sicurezza: "the CIA triad"



Altri requisiti

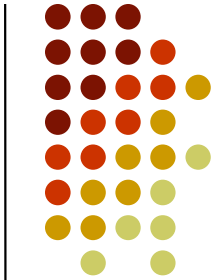


Altri requisiti



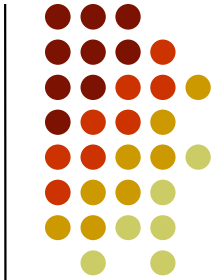
- Privacy (definizione molto imprecisa): confidentialità dell'informazione personale
  - …. Ma vedremo che proteggere la privacy richiede molto di più
- Nota: Il concetto di privacy non può di per se essere definito precisamente perchè si riferisce a "diritti" che dipendono dalla legislazione vigente

Architettura a livelli



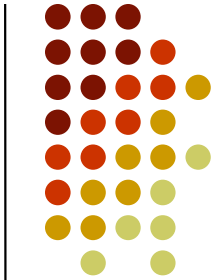
- I computer hanno un'architettura a livelli
  - Hardware
  - Sistema operativo
  - DBMS
  - Applicazioni
- Inoltre sono interconnessi tramite reti e … sono utilizzati da esseri umani
- La protezione dati è richiesta ad:
  - Ogni livello
  - Da utenti/applicazioni
  - Quando i dati sono trasmessi in rete

Meccanismi di sicurezza



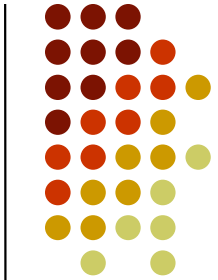
- Ogni meccanismo sviluppato per prevenire, scoprire o ripristinare il sistema dopo un attacco alla sicurezza:
  - tecniche di cifratura, firma digitale, protocolli di autenticazione, controllo dell'accesso, meccanismi di auditing, procedure di recovery, ecc.

Requisiti di sicurezza – come?



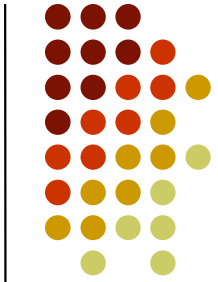
- La confidenzialità è assicurata da:
  - meccanismo di controllo dell'accesso
- L'integrità è assicurata da:
  - meccanismo di controllo dell'accesso
  - vincoli di integrità
  - tecniche di cifratura

Requisiti di sicurezza – come?



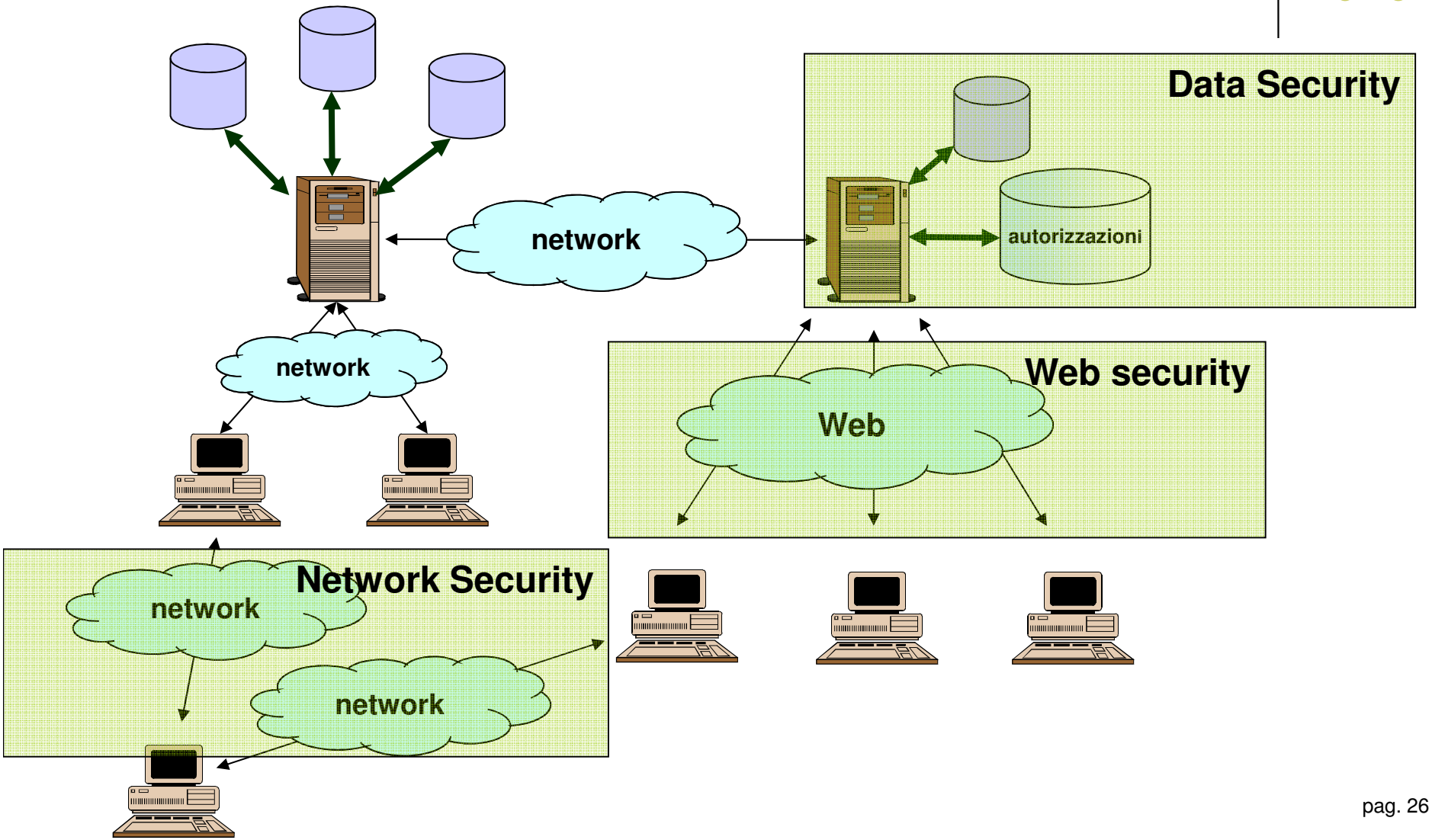
- La disponibilità è assicurata da:
  - tecniche di recovery
  - sistemi anti denial of service
- L'autenticità è assicurata dalle tecniche di firma digitale e protocolli di autenticazione

Requisiti di sicurezza – come?



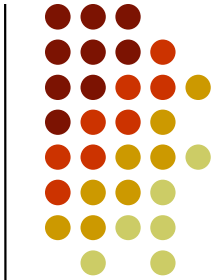
- Bisogna distinguere tra:
  - Sicurezza delle informazioni durante la trasmissione
  - Sicurezza delle informazioni quando risiedono nel sistema informativo

Protezione dei dati



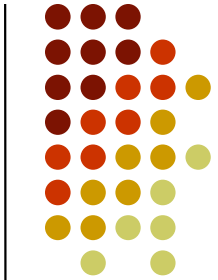
Controllo dell'accesso nei DBMS

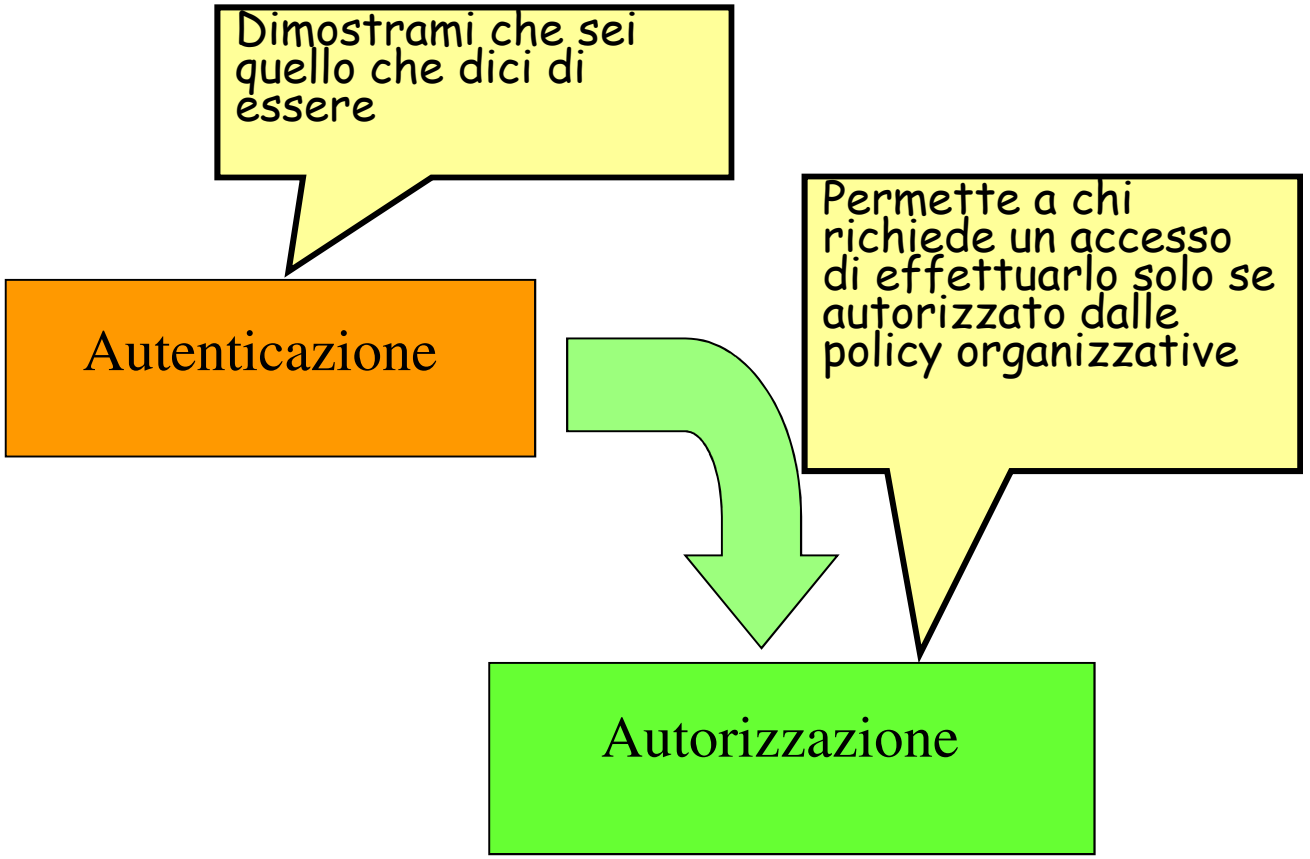
Concetti fondamentali



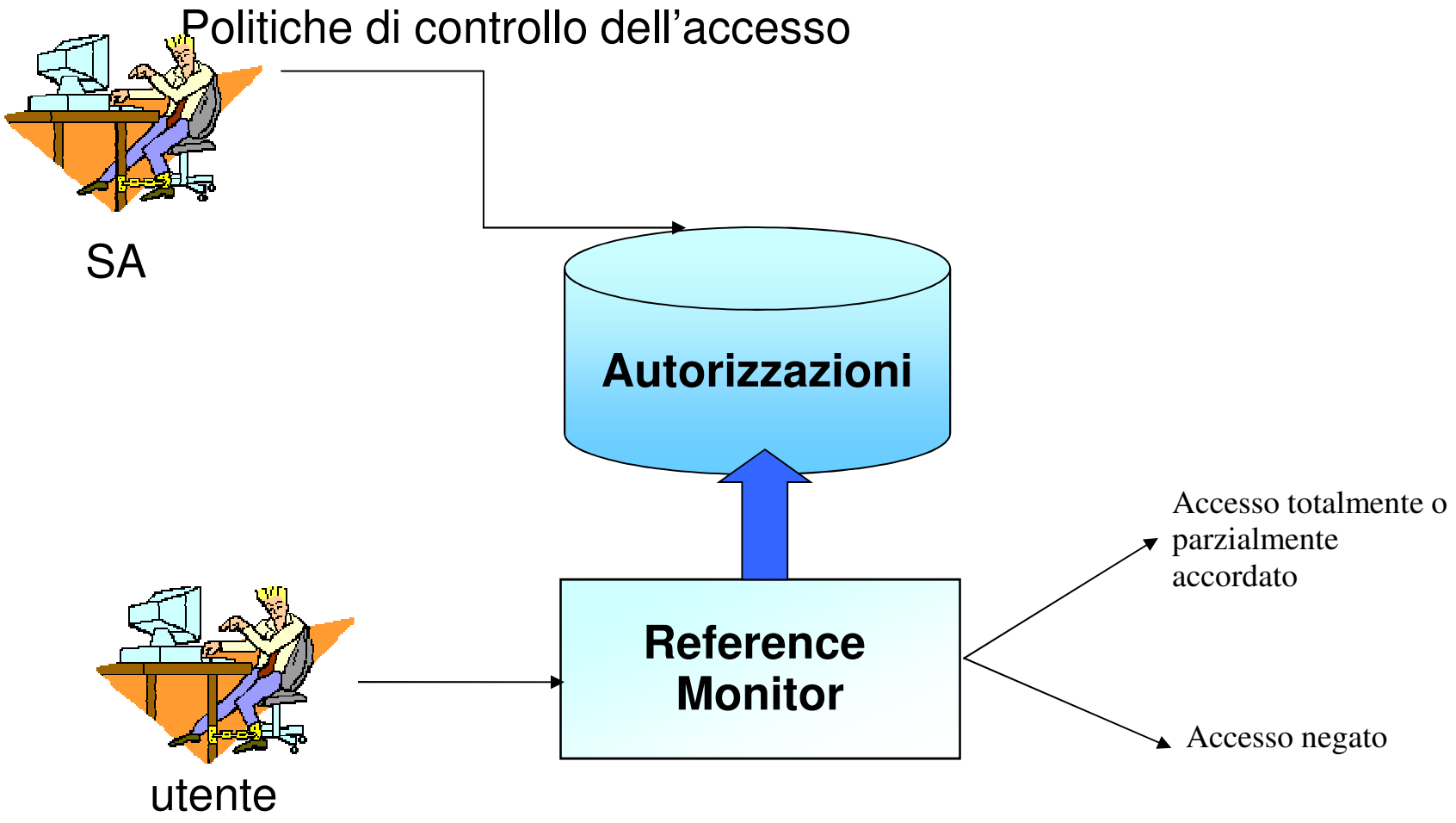
- Regola le operazioni (letture/scritture/update) che si possono compiere sui dati all'interno di un DBMS
- Lo scopo è limitare e controllare le operazioni che gli utenti effettuano, prevenendo azioni accidentali o deliberate che potrebbero compromettere la correttezza e la sicurezza dei dati

Meccanismi di sicurezza

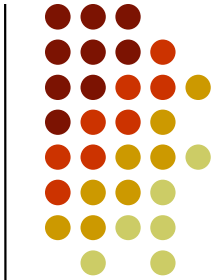




Concetti fondamentali

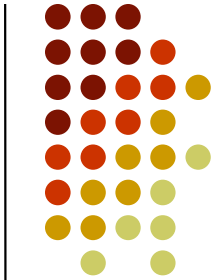


Politiche di controllo dell'accesso



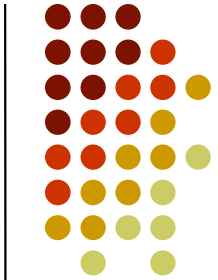
- Regole e principi secondo cui l'organizzazione vuole che siano protette le proprie informazioni:
  - rappresentano un insieme di direttive ad alto livello che esprimono le scelte compiute da un organizzazione in merito alla protezione dei propri dati
  - Es: le valutazioni psicologiche di un impiegato possono essere viste solo dal suo responsabile
  - Sono solitamente espresse in termini di soggetti, oggetti e privilegi

Concetti fondamentali



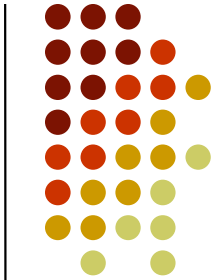
- Oggetti: le risorse a cui vogliamo garantire protezione
- Soggetti: le entità “attive” che richiedono di poter accedere agli oggetti
- Privilegi: le operazioni che i soggetti possono fare sugli oggetti

Concetti fondamentali



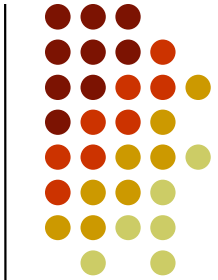
- I soggetti possono essere classificati in:
  - utenti: singoli individui che si connettono al sistema
  - gruppi: insiemi di utenti
  - ruoli: funzioni aziendali a cui assegnare un insieme di privilegi per lo svolgimento delle loro mansioni
  - processi: programmi in esecuzione per conto di utenti
- Eventualmente organizzati in gerarchie

Concetti fondamentali



- Le politiche di controllo dell'accesso sono tradotte in un insieme di *autorizzazioni* che stabiliscono gli specifici diritti che i vari soggetti, abilitati ad accedere al sistema, possono esercitare sugli oggetti:
  - le autorizzazioni nel loro formato base possono essere rappresentate mediante una tupla (*s*,*o*,*p*) dove:
    - *s* è il soggetto a cui l'autorizzazione è concessa
    - *o* è l'oggetto sui cui è concessa l'autorizzazione
    - *p* è il privilegio che s può esercitare su *o*

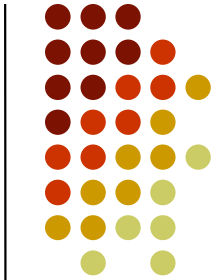
Concetti fondamentali



- Supponiamo che Mario Rossi sia il responsabile di Giovanni Bianchi
- Un'autorizzazione conforme alla politica d'esempio è:

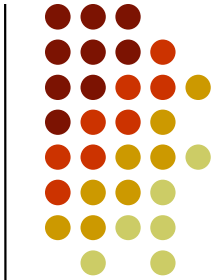
(mario rossi, ValutazionePsicologica(giovanni bianchi), lettura)

Concetti fondamentali



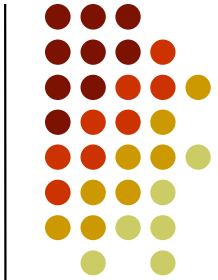
- Il controllo dell'accesso è effettuato mediante il meccanismo di controllo dell'accesso, detto anche reference monitor*:*
  - intercetta ogni comando inviato e stabilisce, tramite l'analisi delle autorizzazioni, se il soggetto richiedente può essere autorizzato (totalmente o parzialmente) a compiere l'accesso richiesto

Sistemi aperti e chiusi



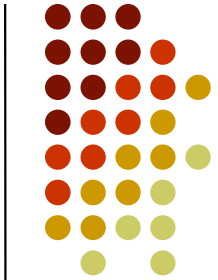
- I sistemi per il controllo dell'accesso possono essere inoltre distinti in **sistemi aperti** e **sistemi chiusi**:
  - in un sistema chiuso l'accesso è permesso *solo se* esplicitamente autorizzato
  - in un sistema aperto l'accesso è permesso *a meno che* non sia esplicitamente negato

Sistemi aperti e chiusi



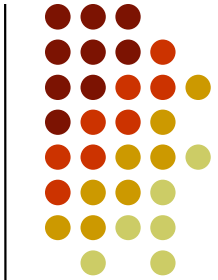
- In un sistema chiuso le autorizzazioni indicano per ogni soggetto i privilegi che egli *può* esercitare sugli oggetti del sistema
- In un sistema aperto le autorizzazioni stabiliscono, per ogni soggetto, i privilegi che egli *non può* esercitare sugli oggetti del sistema

Sistemi aperti e chiusi



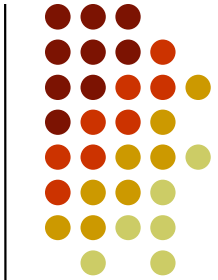
- Un sistema chiuso offre maggiori garanzie di sicurezza rispetto ad un sistema aperto
- Per questa ragione la maggior parte dei DBMS commerciali si comporta come un sistema chiuso

Politiche per il controllo dell'accesso



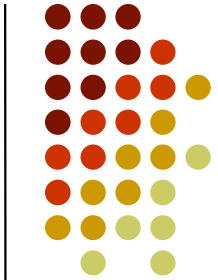
- Le politiche per controllo dell'accesso stabiliscono i criteri in base a cui i soggetti possono accedere agli oggetti nel sistema e se e come i diritti d'accesso possono venire trasmessi a terzi
- Possono essere classificate in tre categorie principali:
  - Politiche discrezionali
  - Politiche mandatorie
  - Politiche basate sui ruoli

Politiche discrezionali



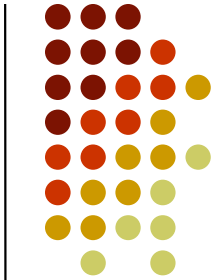
- Consente a determinati utenti nel sistema di specificare chi è autorizzato a compiere quali operazioni
- Nei sistemi che adottano tale politica, esiste un insieme di autorizzazioni che stabiliscono esplicitamente, per ogni soggetto, i privilegi che questo può esercitare sugli oggetti del sistema

Politiche discrezionali



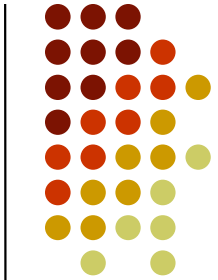
- Il meccanismo di controllo esamina le richieste di accesso accordando solo quelle che sono concesse da un'autorizzazione
- Vengono dette discrezionali, in quanto permettono agli utenti di concedere o revocare dei diritti di accesso sugli oggetti, a loro discrezione

Politiche discrezionali



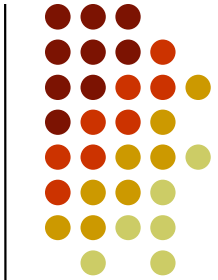
- Vantaggio:
  - sono estremamente flessibili e adatte a numerosi contesti applicativi
- Svantaggio:
  - non forniscono alcun meccanismo di controllo sul flusso di informazioni nel sistema

Politiche discrezionali



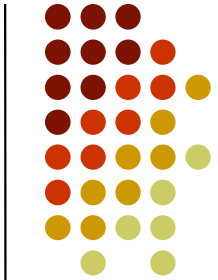
- I meccanismi discrezionali sono vulnerabili ad attacchi, quali quelli perpetrati tramite *cavalli di Troia*
- I cavalli di Troia sono programmi che apparentemente svolgono un compito utile ma che al loro interno contengono delle istruzioni nascoste che utilizzano fraudolentemente le autorizzazioni degli utenti che li eseguono per *trasferire illegalmente informazioni* non violando le limitazioni imposte dalla politica discrezionale

Cavalli di Troia: esempio



- Supponiamo che Anna e Paolo siano due dipendenti di una videoteca e che Anna sia la responsabile di Paolo
- In base alle politiche di controllo dell'accesso:
  - Anna può accedere sia in lettura sia in scrittura a tutte le relazioni della base di dati relativa alla videoteca
  - Paolo non può vedere i titoli dei film noleggiati dai clienti della videoteca

Cavalli di Troia: esempio

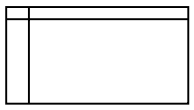


- Paolo regala ad Anna un programma per la gestione dell'agenda, in cui ha fraudolentemente inserito alcune istruzioni (il cavallo di Troia) che:
  - creano una tabella `FilmClienti` contenente, per ogni cliente, i titoli dei film che ha noleggiato
  - concedono a Paolo l'autorizzazione ad effettuare interrogazioni su di essa

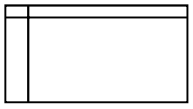
Cavalli di Troia: esempio

Programma per la gestione dell'agenda eseguito da Anna

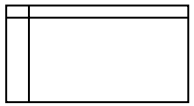
```
.............
CREATE TABLE FilmClienti(
codCli DECIMAL(4),
titolo VARCHAR(30),
regista VARCHAR(30),
dataNol Date,
PRIMARY KEY (codCli,titolo,regista,dataNol));

INSERT INTO FilmClienti
          SELECT  codCli, titolo, regista, dataNol
FROM  Cliente NATURAL JOIN Noleggio NATURAL JOIN Video;

GRANT ALL PRIVILEGES ON  FilmClientI TO Paolo WITH GRANT OPTION;


.............
```



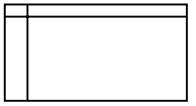
(Anna,r,Cliente)



(Anna,r,Noleggio)

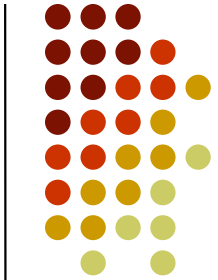


(Anna,r,Video)



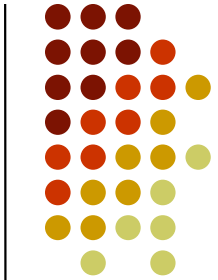
(Paolo,all,FilmClienti)

Cavalli di Troia: esempio



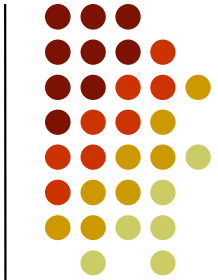
- Quando Anna esegue il programma, il meccanismo di controllo dell'accesso autorizzerà o meno le operazioni effettuate dal programma in base alle autorizzazioni possedute da Anna
- In questo modo Paolo, senza violare i controlli della politica discrezionale, potrà accedere ad informazioni per cui non ha le necessarie autorizzazioni

Politiche mandatorie



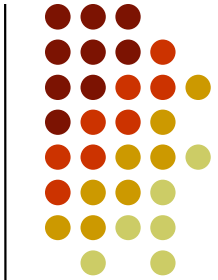
- Gli utenti non possono modificare lo stato delle autorizzazioni nel sistema
- Regolano l'accesso ai dati mediante la definizione di *classi di sicurezza*, chiamate anche etichette, per i soggetti e gli oggetti del sistema
- Le classi di sicurezza sono ordinate parzialmente (o totalmente) da una relazione d'ordine

Politiche mandatorie



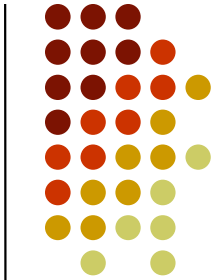
- La classe di sicurezza assegnata ad un oggetto è una misura della sensibilità dell'informazione che l'oggetto contiene:
  - > è la classe assegnata ad un oggetto, più ingente è il danno derivante dal rilascio delle informazioni in esso contenute a soggetti non autorizzati
- La classe di sicurezza assegnata ad un soggetto è una misura del grado di fiducia che si ha nel fatto che tale soggetto non commetta violazioni della politica ed, in particolare, non trasmetta ad altri informazioni riservate

Politiche mandatorie



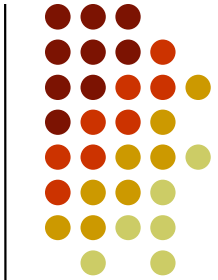
- Il controllo dell'accesso è regolato da una serie di *assiomi di sicurezza* che stabiliscono la relazione che deve intercorrere fra la classe di un soggetto e quella di un oggetto affinché al primo sia concesso di esercitare un privilegio sul secondo
- La relazione che deve essere soddisfatta dipende dal tipo di privilegio considerato

Politiche mandatorie



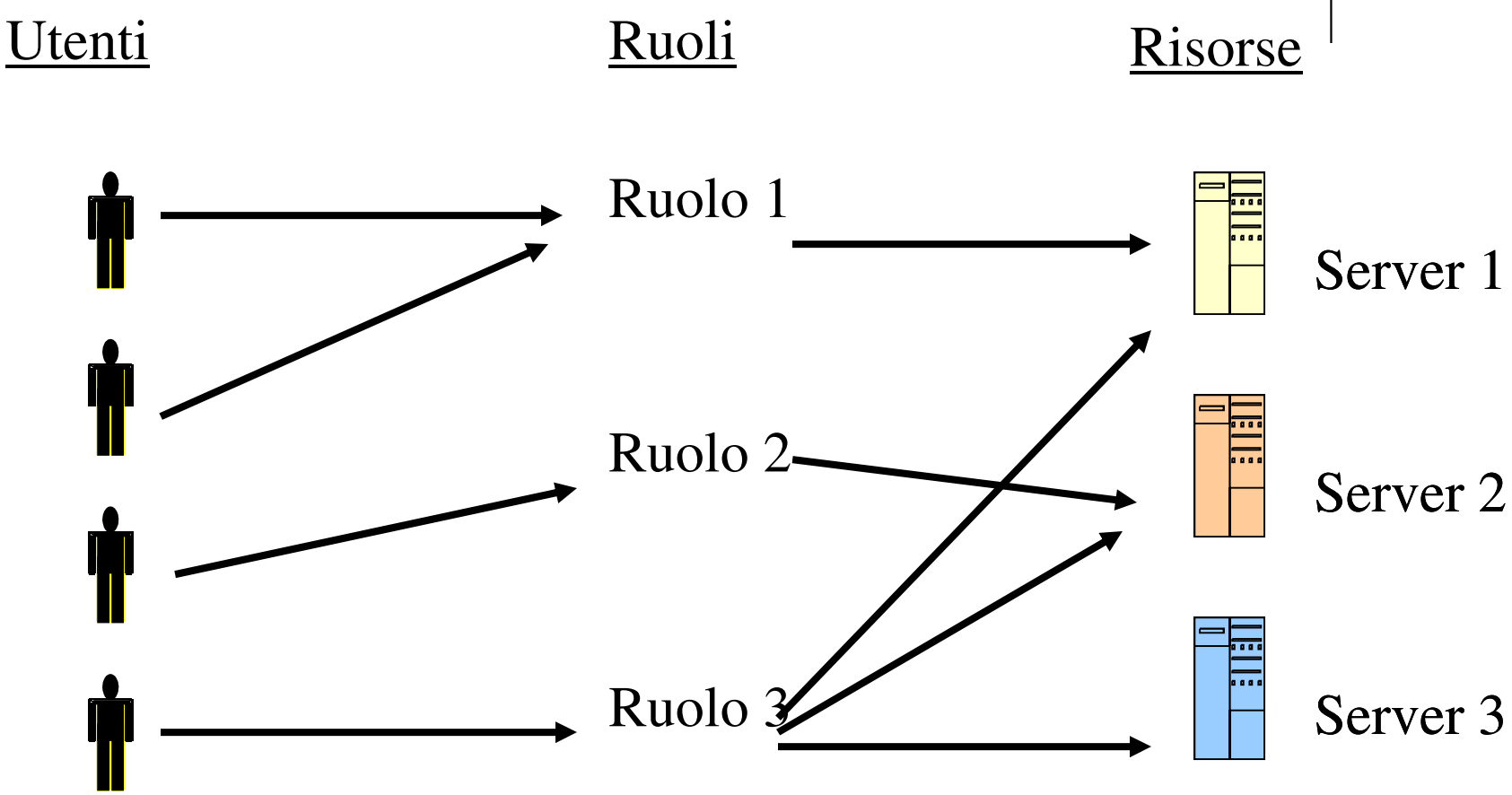
- Le politiche mandatorie sono state inizialmente applicate in ambienti, quali quello militare, dove ci sono forti esigenze di protezione ed è possibile classificare rigidamente gli elementi del sistema
- I sistemi che adottano politiche mandatorie sono spesso indicati come *sistemi multi-livello*

Politiche basate sui ruoli

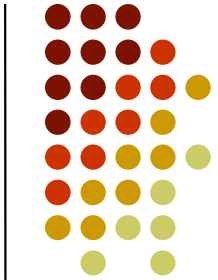


- In base a tali politiche, i privilegi non sono direttamente assegnati agli utenti ma sono mediati dal concetto di ruolo
- Un ruolo rappresenta una funzione all'interno di un'azienda od organizzazione (esempio: *direttoreVideoteca*, *commesso* e *cliente*)
- Le autorizzazioni non sono concesse ai singoli utenti ma ai ruoli
- Ogni utente è poi abilitato a ricoprire uno o più ruoli ed in questo modo acquisisce le autorizzazioni ad essi associate

Politiche basate sui ruoli

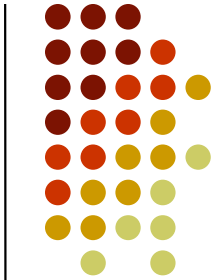


Politiche basate sui ruoli



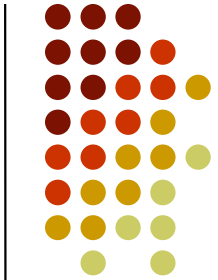
- Il concetto di ruolo semplifica notevolmente la gestione delle autorizzazioni
- In un'azienda/organizzazione spesso i privilegi esercitabili non sono legati all'identità dei singoli ma piuttosto al ruolo che essi ricoprono all'interno della stessa

Politiche basate sui ruoli



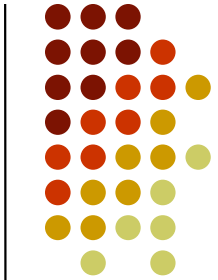
- Vantaggi:
  - un ruolo di solito raggruppa un cospicuo numero di privilegi:
    - Quando un utente deve ricoprire la mansione associata ad un certo ruolo, invece di specificare per lui tutte le autorizzazioni associate al ruolo, basta abilitarlo a ricoprire quel ruolo

Politiche basate sui ruoli



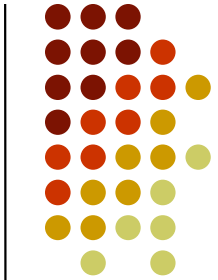
- Vantaggi:
  - un ruolo è più stabile rispetto agli utenti, che possono cambiare frequentemente e/o cambiare le loro mansioni
    - Un cambio di mansioni in un sistema che supporta i ruoli viene gestito molto facilmente
    - configurando opportunamente il sistema dei ruoli, le politiche basate suoi ruoli possono essere usate per realizzare sia politiche discrezionali sia mandatorie

Ulteriori varianti



- Controllo dell'accesso basato su attributi (ABAC Attribute-based Access Control)
- Controllo dell'accesso basato sul contenuto
- Controllo dell'accesso basato sulle finalità (PBAC Purpose-based Access Control)
- Controllo dell'accesso time-based
- Controllo dell'accesso basato sulla localizzazione

ABAC

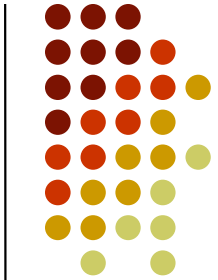


- Soggetti ed oggetti sono caratterizzati da attributi che denotano proprietà rilevanti ai fini del controllo dell'accesso (es. età, qualifica)
- I soggetti e gli oggetti in una autorizzazione sono denotati implicitamente ponendo delle condizioni sugli attributi (es. età > 18)

Controllo dell'accesso basato sul contenuto

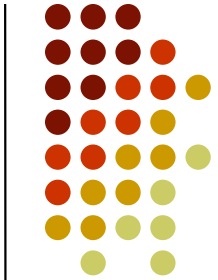
- Le autorizzazioni vengono specificate ponendo dei vincoli sul contenuto degli oggetti (es. Le informazioni su coloro che guadagnano più di 100.000 euro possono essere accedute solo dal responsabile della divisione risorse umane)
- In SQL questo è ottenuto mediante il meccanismo delle viste
- Oracle VPD implementa il controllo dell'accesso in base al contenuto

PBAC



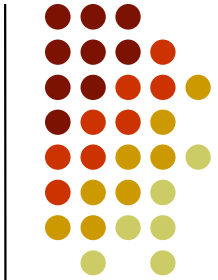
- Un'aspetto importante per la protezione della privacy riguarda la finalità per cui viene richiesto/memorizzato un dato:
  - L'indirizzo di un cliente viene memorizzato solo ai fini della spedizione della merce
- In PBAC le autorizzazioni hanno un ulteriore parametro per tenere traccia della finalità dell'accesso autorizzato

Controllo dell'accesso time-based



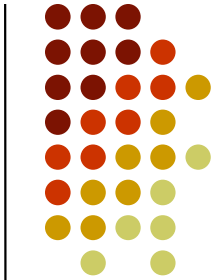
- Per ottenere una maggiore protezione le autorizzazioni sono attivabili solo in determinati momenti o periodi di tempo:
  - L'amministratore di sistema preposto a fare il back-up può accedere i file di progetto solo Venerdì pomeriggio dalle 15 alle 18.

Controllo dell'accesso basato sulla localizzazione



- Permette di condizionare gli accessi al posto dove sono effettuati:
  - Solo in locali videosorvegliati
  - Solo all'interno di un parco per cui ho pagato il biglietto di ingresso

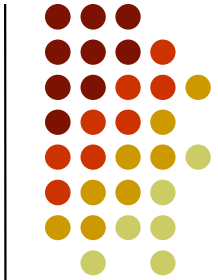
Politiche di amministrazione



Definisce chi concede e revoca i diritti di accesso

- **Centralizzata**
  - un unico autorizzatore (o gruppo), detto amministratore della sicurezza, può specificare o revocare le autorizzazioni
- **Basata su ownership**
  - l'utente che crea un oggetto, detto *owner*, gestisce anche la concessione e la revoca di autorizzazioni sull'oggetto

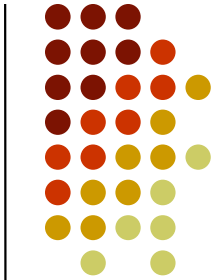
Politiche di amministrazione



- **Decentralizzata**
  - l'owner o l'amministratore della sicurezza possono delegare ad altri la facoltà di specificare e revocare autorizzazioni. Molti DBMS commerciali adottano la politica basata su ownership con delega
- **Cooperativa**
  - le autorizzazioni non possono essere concesse da un singolo utente, ma richiedono il consenso di più utenti

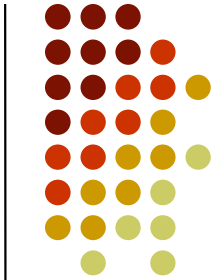
Modelli di controllo dell'accesso

Modelli per il controllo dell'accesso



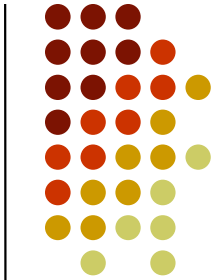
- Derivano dai modelli per la protezione di risorse in un sistema operativo
- Differenze fondamentali tra i due contesti:
  - diversi livelli di granularità (relazione, tupla, o singolo attributo)
  - protezione di strutture completamente logiche (viste, ecc.) invece di risorse reali (stampanti, ecc.)

Modello di Bell e LaPadula



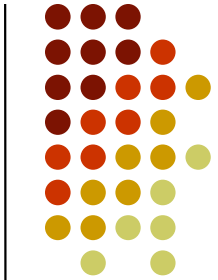
- Modello di riferimento per sistemi che adottano una politica di tipo mandatorio
- I privilegi previsti sono:
  - read: leggere ma non modificare
  - append: modificare ma non leggere
  - write: sia modificare che leggere
  - execute: eseguire ma non leggere né modificare direttamente
- Nel seguito considereremo solo i privilegi read, write ed append in quanto strettamente pertinenti alla gestione dati

Modello di Bell e LaPadula



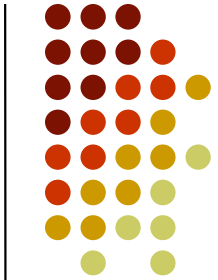
- I soggetti e gli oggetti del sistema vengono classificati mediante l'assegnazione di una classe di sicurezza
- Una classe di sicurezza è costituita da due componenti: un livello di sicurezza ed un insieme di categorie

Modello di Bell e LaPadula



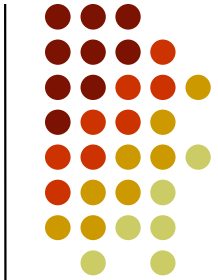
- Il livello di sicurezza è un elemento di un insieme totalmente ordinato
  - ad esempio: Top Secret (TS), Secret (S), Confidential (C) e Unclassified (U), dove: TS > S > C > U
- L'insieme di categorie è un insieme non ordinato di elementi, anche vuoto, che dipendono dal tipo di ambiente considerato e denotano aree di competenza all'interno dell'organizzazione
  - ambito militare: Army, Navy, Air Force, e Nuclear
  - ambito commerciale: Finanza, Vendite, Ricerca, e Sviluppo
  - un file contenente informazioni finanziarie riservate potrebbe essere classificato (Confidential, {Finanza})

Relazione di dominanza



- Una classe di sicurezza cs1 = (L1, Cat1) domina una classe di sicurezza cs2 = (L2, Cat2), (cs1 ≥ cs2), se entrambe le seguenti condizioni sono verificate:
  1. il livello di sicurezza di cs1 è maggiore o uguale al livello di sicurezza di cs2 (cioè L1 ≥ L2)
  2. l'insieme di categorie di cs1 include l'insieme di categorie di cs2 (cioè Cat1 ⊇ Cat2)
- Se L1 > L2 e Cat1 ⊃ Cat2 , si dice che cs1 *domina strettamente* cs2 (cs1 > cs2)
- cs1 e cs2 si dicono *incomparabili* (cs1 <> cs2) se né cs1 ≥ cs1 né cs2 ≥ cs1 valgono

Esempio

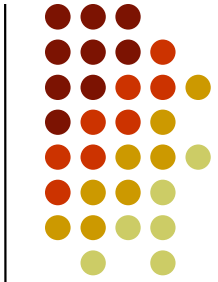


- Classi di accesso:
  - cs1 = (TS,{Nuclear,Army})
  - cs2 = (TS,{Nuclear})
  - cs3 = (C,{Army})
- cs1 $\geq$ cs2
  - L1=L2 e Cat1 $\supseteq$ Cat2
- cs1 > cs3
  - L1> L3 (TS > C) e Cat3 $\subset$ Cat1
- cs2 <> cs3
  - cs2 $\not\geq$ cs3 perché {Nuclear} $\not\supseteq$ {Army} e cs3 $\not\geq$ cs2 perché TS > C

Modello di Bell e LaPadula

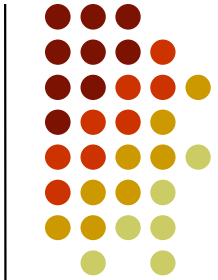
- Lo stato del sistema è descritto dalla coppia (A, $\mathcal{L}$), dove:
  - A è l'insieme degli accessi correnti. A è composto da triple della forma (*s*,*o*,*p*) che indicano che il soggetto *s* sta correntemente esercitando il privilegio *p* sull'oggetto *o*
  - $\mathcal{L}$ è la funzione di livello che associa ad ogni elemento del sistema la sua classe di sicurezza
    - $\mathcal{L}: O \cup S \rightarrow CS$

Modello di Bell e LaPadula



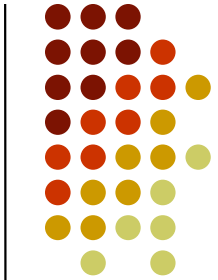
- Ogni modifica di stato è causata da una *richiesta*
- Data una richiesta ed uno stato corrente, la decisione ed il nuovo stato sono determinati in base a degli *assiomi di sicurezza*
- Tali assiomi stabiliscono le condizioni che devono essere soddisfatte nel sistema per effettuare una transizione di stato
- Se le richieste sono soddisfatte solo se verificano gli assiomi, allora il sistema è *sicuro*

Assiomi



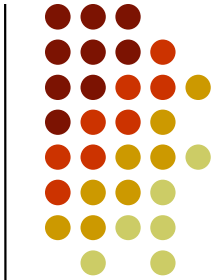
- Proprietà di sicurezza semplice
  - uno stato (A,$\mathcal{L}$) soddisfa la proprietà di sicurezza semplice se per ogni elemento (*s*, *o*, *p*) in A tale che *p=read* oppure *p=write*: $\mathcal{L}(s) \geq \mathcal{L}(o)$

Assiomi



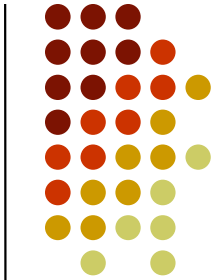
- La proprietà di sicurezza semplice (*proprietà del no read-up*) assicura che i soggetti abbiano accesso solo alle informazioni per cui hanno la necessaria classificazione, prevenendo letture di oggetti con classe di sicurezza maggiore o incomparabile
- Esempio:
  - un soggetto con classe di sicurezza (C, {Army}) non può leggere oggetti con classe di sicurezza (C,{Navy, Air Force}) o (U,{Air Force}), mentre può leggere oggetti con classe di sicurezza (U,{Army})

Assiomi



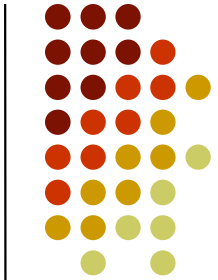
- La proprietà di sicurezza semplice non evita però che una combinazione di accessi possa compromettere la sicurezza dei dati:
  - Esempio: un soggetto con classe di sicurezza (TS,{Army}) potrebbe estrarre informazioni da un oggetto con classe di sicurezza (TS, {Army}) ed inserirle in uno con classe di sicurezza (U, {Army}), senza violare la proprietà di sicurezza semplice, che regola principalmente le operazioni di lettura

Assiomi



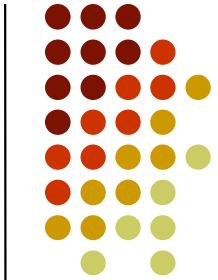
- Per evitare tali situazioni, è stato definito un ulteriore assioma, riferito specificatamente alle operazioni di scrittura

Assiomi



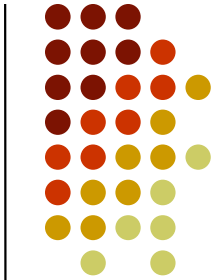
- Proprietà Star (*)
  - uno stato (A,$\mathcal{L}$) soddisfa la proprietà *, se per ogni elemento (*s, o, p*) in A tale che *p=append* o *p=write:* $\mathcal{L}(s) \leq \mathcal{L}(o)$

Assiomi



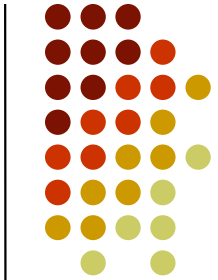
- La proprietà * (*proprietà del no write-down*) è definita per prevenire il flusso di informazioni dovute a scritture verso classi di sicurezza minori o non comparabili
- Esempio
  - un soggetto con classe di sicurezza (C,{Army,Nuclear}) non può scrivere informazioni in oggetti con classe di accesso (U, {Army,Nuclear})

Assiomi



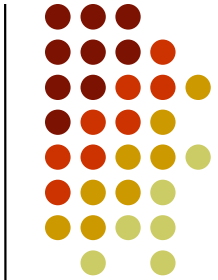
- Un sistema è definito sicuro se ogni elemento aggiunto all'insieme degli accessi correnti verifica la proprietà di sicurezza semplice e la proprietà *
- Questo implica che, per il privilegio *write,* devono essere applicati entrambi gli assiomi
  - Un soggetto *s* può esercitare il privilegio *write* sull'oggetto *o* solo se $\mathcal{L}(s) = \mathcal{L}(o)$.

Osservazioni



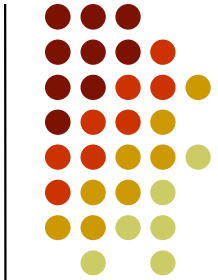
- L'applicazione delle due proprietà può comportare un'eccessiva rigidità del sistema
- Esempio:
  - Generale con classe di sicurezza (TS, {Army Nuclear}), mentre suo colonnello ha classe di sicurezza (C, {Army}).
  - Il colonnello può potenzialmente comunicare con il generale, in quanto gli è concessa la scrittura in modalità append di oggetti con classe di sicurezza maggiore,
  - Il generale non può comunicare con il suo colonnello in quanto, in base alla proprietà *, il generale non può scrivere in oggetti con classe di sicurezza minore della sua

Osservazioni



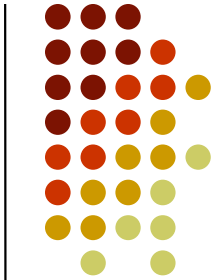
- Per ovviare a questi problemi, un utente può connettersi al sistema con una qualsiasi classe di sicurezza dominata da quella a lui assegnata
- Esempio
  - Il generale può connettersi al sistema con classe di sicurezza (C,{Army}) e comunicare quindi con il suo colonnello

Osservazioni



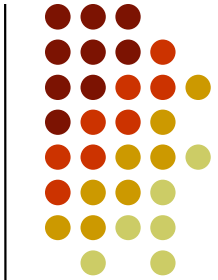
- Il modello di Bell e LaPadula da buone garanzie di sicurezza solo in presenza di una *classificazione statica* di soggetti ed oggetti in classi di sicurezza (tale principio è noto come *strong tranquillity principle*)
- Se tale principio non è soddisfatto, la sicurezza o meno del sistema dipende dal modo con cui possono essere modificate le classi di sicurezza

Osservazioni



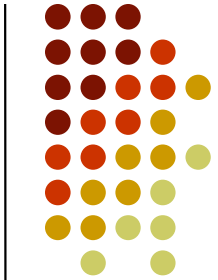
- Si consideri un sistema con la seguente politica di controllo dell'accesso:
  - quando un soggetto s richiede un qualsiasi accesso ad un oggetto o, l'accesso è concesso e la classe di sicurezza di tutti gli oggetti/soggetti diventa la più bassa possibile
- Il sistema soddisfa la definizione di sistema sicuro del modello BLP …. Anche se presenta ovvi problemi di sicurezza!

Osservazioni



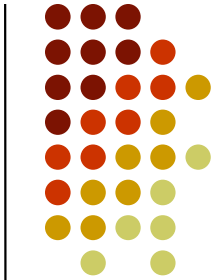
- Nella pratica, il principio di tranquillità è troppo restrittivo per essere applicato, in quanto ci possono essere situazioni in cui è necessaria una riclassificazione di soggetti ed oggetti rispetto alle classi di sicurezza
- Per questo sono stati definiti principi meno restrittivi, che consentono la modifica delle classi di sicurezza sotto opportune condizioni
- Viene inoltre introdotta la nozione di *soggetto fidato* (trusted), cioè un soggetto che può violare alcune delle restrizioni imposte dal modello

Osservazioni



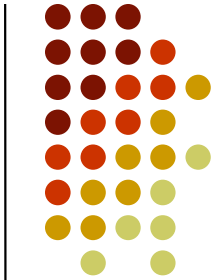
- Ad esempio, *Oracle Label Security*, prevede i seguenti privilegi speciali:
  - `READ`, che consente di evitare i controlli in lettura degli assiomi mandatori
  - `FULL`, che consente di evitare sia i controlli in lettura sia in scrittura degli assiomi mandatori
  - `WRITEDOWN`, che consente di diminuire il livello di sicurezza di una classe di sicurezza
  - `WRITEUP`, che consente di aumentare il livello di sicurezza di una classe di sicurezza

Osservazioni



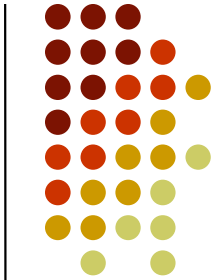
- Il modello di Bell e LaPadula permette ad un soggetto di esercitare il privilegio `append` su oggetti con una classe di sicurezza superiore alla sua
- Questo può essere utile in certi contesti e pericoloso in altri
  - Un documento a cui possono accedere in scrittura soggetti non autorizzati a leggerlo, può essere reso inconsistente da tali soggetti

Osservazioni



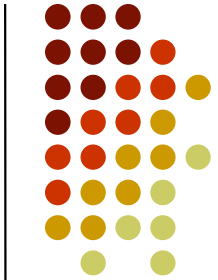
- Un approccio drastico al problema è quello di modificare il modello di Bell e LaPadula vietando le scritture verso classi di sicurezza maggiori o non comparabili, siano esse dovute all'esercizio del privilegio `append` o `write`

Osservazioni



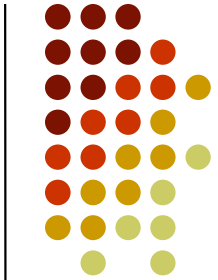
- E’ un modello usato in alcuni sistemi operativi ed alcuni DBMS
- Alcune critiche:
  - Non supporta politiche per modificare i diritti di accesso
  - Non è sempre facile classificare soggetti ed oggetti in classi di sicurezza
  - Se le classi di sicurezza possono essere dinamicamente modificate la sicurezza del sistema non è garantita
  - Può dare origine a problemi di sicurezza/integrità quando applicato a DBMS

Polistanziazione



- BPL è stato originariamente definito per sistemi operativi
- Un DBMS contiente oggetti a vari livelli di granularità (relazioni, tuple, attributi):
  - Ad esempio, il noleggio di film diversi può avere classi di sicurezza diverse
- Esiste quindi la necessità di gestire differenti versioni della stessa entità a diversi livelli di sicurezza senza violare l'integrità del db (polistanziazione)

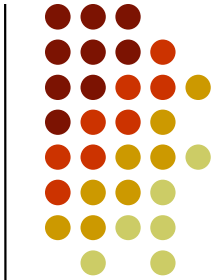
Esempio



Nome	L	Età	L	Stipendio	L
Leo	U	28	U	50k	U
Ann	U	35	S	100K	U
Marc	S	40	S	95	S

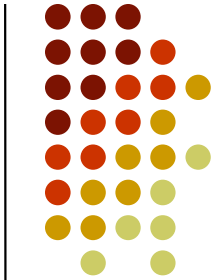
- **un utente con livello U richiede l'inserimento di (Marc,40,100)**

Esempio



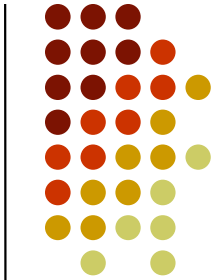
Nome	L	Età	L	Stipendio	L
Leo	U	28	U	50k	U
Ann	U	35	S	100K	U
Marc	S	40	S	95	S
Marc	U	40	U	100k	U

Polistanziazione



- Non è facile gestire l'integrità del db
- Soluzione: polistanziazione parziale e la verifica di un insieme di vincoli di sicurezza (ad esempio granularità minima = tupla)

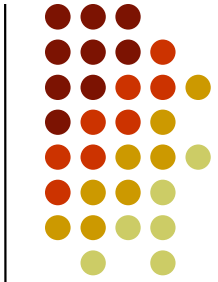
MAC



- MAC non è supportato da SQL
- I primi MLS/DBMS non hanno avuto successo
- Oggi si assiste alla cosidetta “multilevel security reprise”, (si veda Oracle, SELinux, Windows Vista) dovuta ai sempre maggiori requisiti di sicurezza che si hanno in domini anche diversi da quello militare
- Oggi il trend è combinare controllo dell’accesso mandatorio e discrezionale

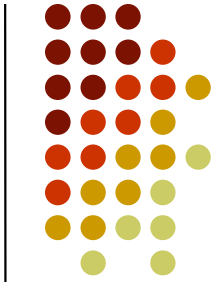
Modelli discrezionali

System R



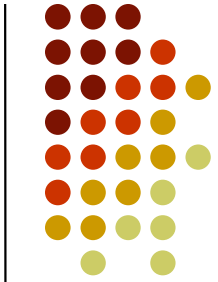
- Uno dei modelli più rilevanti per sistemi che adottano politiche discrezionali
- Uno dei primi che ha ricevuto un'effettiva implementazione nel DBMS IBM System R
- Costituisce la base per i meccanismi di controllo dell'accesso oggi presenti nei DBMS commerciali e per le funzionalità previste dallo standard SQL

System R



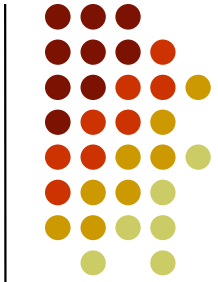
- Gli oggetti del modello sono sia relazioni di base sia viste
  - nel seguito utilizzeremo il termine relazione per far riferimento sia a relazioni di base sia viste
- I privilegi previsti dal modello corrispondono alle operazioni effettuabili tramite comandi SQL (ad esempio, SELECT, INSERT, DELETE, UPDATE)

System R



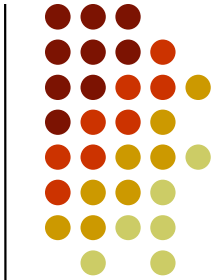
- Il modello realizza una politica di tipo discrezionale adottando il paradigma di sistema chiuso
- L'amministrazione dei privilegi é decentralizzata mediante ownership

System R



- La delega dei privilegi avviene mediante *grant option*
- Se un privilegio è concesso con grant option l'utente che lo riceve può non solo esercitare il privilegio, ma anche concederlo ad altri
- Un utente può quindi concedere un privilegio su una determinata relazione solo se è il proprietario della relazione o ha ricevuto tale privilegio con grant option

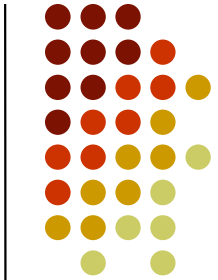
Comando GRANT



- Le autorizzazioni sono concesse tramite il comando GRANT:

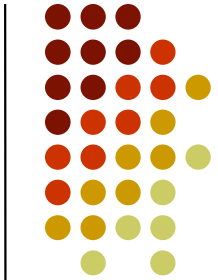
```
GRANT {<lista privilegi> | ALL[PRIVILEGES]}
ON <nome relazione>
TO {<lista utenti> | PUBLIC}
[WITH GRANT OPTION];
```

Comando GRANT



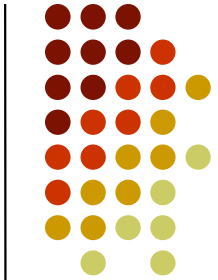
- Dove
  - `<lista privilegi>` indica l'insieme dei privilegi concessi con il comando GRANT. Le parole chiave `ALL` o `ALL PRIVILEGES` consentono di concedere con un solo comando tutti i privilegi
  - `<nome relazione>` indica il nome della relazione su cui sono concessi i privilegi
  - `<lista utenti>` indica l'insieme degli utenti a cui il comando si applica. La parola chiave `PUBLIC` consente di specificare le autorizzazioni implicate dal comando per tutti gli utenti del sistema.
  - La clausola opzionale `WITH GRANT OPTION` consente la delega dell'amministrazione dei privilegi oggetto del comando

Comando GRANT



- I privilegi concessi con un comando `GRANT` si applicano ad intere relazioni, ad eccezione del privilegio `update` per cui è possibile specificare le colonne su cui si applica (racchiuse tra parentesi tonde e separate da virgole)
- È inoltre possibile concedere piu' privilegi su una stessa relazione con un unico comando (i privilegi devono essere separati tramite virgole)
- Analogamente, un unico comando `GRANT` può essere utilizzato per concedere privilegi sulla stessa relazione a più utenti

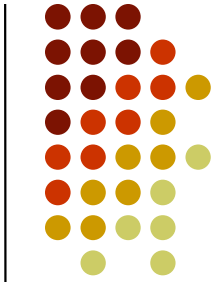
Esempio



- Base di dati relativa alla gestione della Videoteca dove tutte le relazioni sono state create da Luca

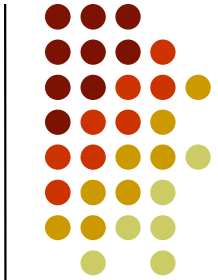
```
luca: GRANT update(telefono) ON Clienti TO marco;

luca: GRANT select ON Film TO barbara, giovanna WITH
  GRANT OPTION;

giovanna: GRANT select ON Film TO matteo;

luca: GRANT ALL PRIVILEGES ON Video, Film TO elena
  WITH GRANT OPTION;

elena: GRANT insert, select ON Film TO barbara;
```

System R



- Un utente potrebbe ricevere lo stesso privilegio sulla stessa relazione da utenti diversi
  - questa possibilità ha ripercussioni sull'operazione di revoca dei privilegi
- Dato che i privilegi possano essere concessi con grant option, i privilegi che ogni utente possiede possano essere classificati in due categorie:
  - *privilegi delegabili*
  - *privilegi non delegabili*

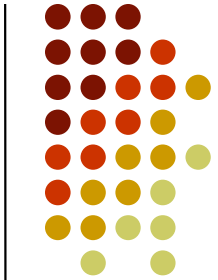
Comando REVOKE



- I privilegi concessi possono essere revocati tramite il comando REVOKE

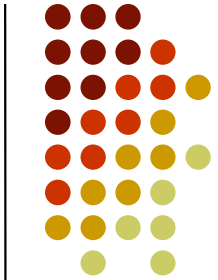
```
REVOKE {<lista privilegi>|ALL[PRIVILEGES]}
ON <nome relazione>
FROM {<lista utenti> | PUBLIC};
```

Comando REVOKE



- Un utente può revocare solo i privilegi da lui stesso concessi
- È possibile revocare più privilegi con un unico comando di REVOKE
- Un unico comando di REVOKE può essere utilizzato per revocare gli stessi privilegi sulla stessa relazione ad utenti diversi

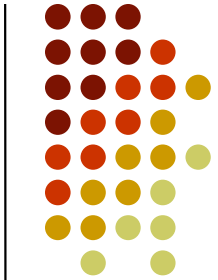
Esempio



- Comandi di REVOKE eseguiti da Luca

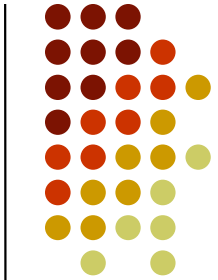
```
REVOKE update, insert ON Video FROM elena;

REVOKE update ON Clienti FROM marco;

REVOKE select ON Film FROM giovanna;
```

System R



- La presenza della grant option fa si che la revoca di un privilegio ad un utente non necessariamente comporti la perdita di quel privilegio da parte dell'utente:
  - L'utente può continuare ad esercitare il privilegio anche dopo l'operazione di revoca nel caso in cui lo abbia ricevuto da altre fonti *indipendenti*

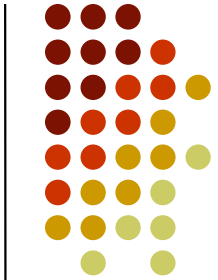
Esempio



```
luca: REVOKE  select ON  Film  FROM  barbara,
giovanna;
```

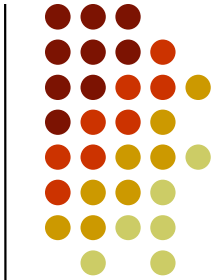
- Giovanna non può più effettuare interrogazioni sulla relazione `Film`, mentre Barbara mantiene ancora tale privilegio, grazie all'autorizzazione concessale da Elena
- Barbara non potrà però più concedere a terzi il privilegio `select` su `Film`

Cataloghi `Sysauth` e `Syscolauth`



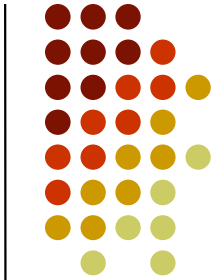
- Le informazioni sull'insieme di autorizzazioni correntemente presenti nel sistema sono memorizzate in due cataloghi di sistema di nome `Sysauth` e `Syscolauth`
- `Sysauth` contiene informazioni sui privilegi che ogni utente ha sulle relazioni della base di dati
- `Syscolauth` serve per la gestione del privilegio `update`

Cataloghi `Sysauth` e `Syscolauth`



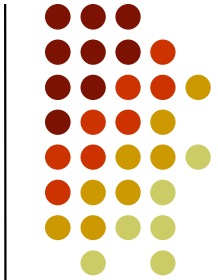
- Lo schema di Sysauth è costituito dai seguenti attributi:
  - `utente`, è l'identificatore dell'utente a cui sono concessi i privilegi.
  - `nome_rel`, indica il nome della relazione su cui sono concessi i privilegi.
  - `tipo` in {R,V}, indica se l'oggetto dell'autorizzazione è una relazione di base (`tipo`='R') o una vista (`tipo`='V')

Cataloghi `Sysauth` e `Syscolauth`



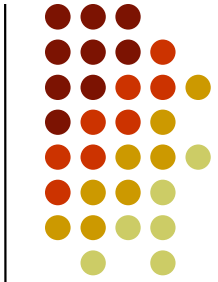
- `select`, indica se l'utente ha il privilegio di effettuare interrogazioni sulla relazione considerata
  - Tale colonna contiene un *timestamp*, che denota il tempo in cui il privilegio è stato concesso; il valore 0 indica che il relativo privilegio non è stato concesso
  - Esistono colonne analoghe per ogni privilegio previsto dal modello
- `grantor`, è l'identificatore dell'utente che ha concesso l'autorizzazione
- `grantopt` in {Y,N}, indica se i privilegi sono delegabili (`grantopt` ='Y') o meno (`grantopt` ='N')

Cataloghi `Sysauth` e `Syscolauth`



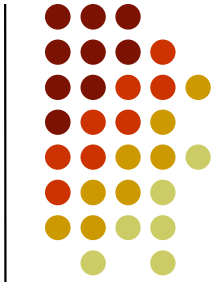
- Le informazioni sul timestamp sono fondamentali per la corretta implementazione dell'operazione di revoca dei privilegi
- Il timestamp può essere sia un semplice contatore sia indicare un tempo reale.
- Deve rispettare le seguenti proprietà:
  - Essere monotonicamente crescente;
  - Non esistano due comandi `GRANT` con lo stesso timestamp

Esempio



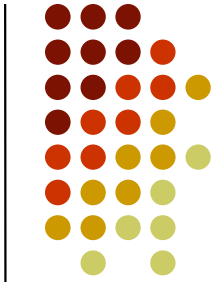
Utente	nomeRel	Tipo	Select	Insert	Update	Grantor	Grantopt
Luca	Clienti	R	20	20	20		Y
Luca	Video	R	22	22	22		Y
Luca	Film	R	25	25	25		Y
Marco	Clienti	R	0	0	30	Luca	N
Barbara	Film	R	32	0	0	Luca	Y
Giovanna	Film	R	32	0	0	Luca	Y
Matteo	Film	R	35	0	0	Giovanna	N
Elena	Video	R	40	40	40	Luca	Y
Elena	Film	R	40	40	40	Luca	Y
Barbara	Film	R	47	47	0	Elena	N

Cataloghi `Sysauth` e `Syscolauth`



- Il catalogo `Sysauth` mantiene solo informazioni di tipo “qualitativo” sui privilegi `update` che ogni utente può esercitare
  - registra solo il fatto che un utente possa esercitare o meno il privilegio `update` su una certa relazione, ma non tiene traccia delle colonne su cui può essere esercitato

Cataloghi `Sysauth` e `Syscolauth`

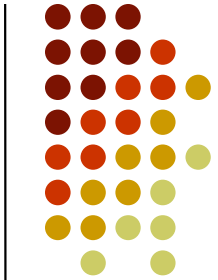


- Tali informazioni sono mantenute nel catalogo `Syscolauth` che contiene una tupla:

(`utente`,`nome_rel`,`colonna`,`grantor`,`grantopt`)

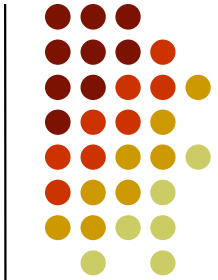
per ogni colonna della relazione `nome_rel` su cui l'utente identificato da `utente` può esercitare il privilegio `update`

Concessione di privilegi



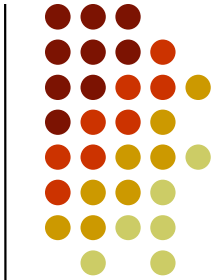
- Quando un utente richiede l'esecuzione di un comando `GRANT`, il meccanismo di controllo dell'accesso legge i cataloghi `Sysauth` e `Syscolauth`, per determinare se il richiedente possiede o meno il diritto di concedere i privilegi specificati nel comando

Concessione di privilegi



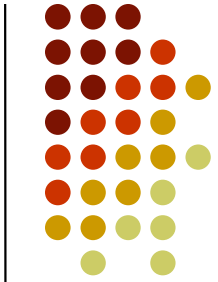
- L'insieme dei privilegi delegabili che l'utente possiede è intersecato con l'insieme dei privilegi specificati nel comando `GRANT`. Sono possibili tre risultati:
  - l'intersezione è vuota: il comando non viene eseguito;
  - l'intersezione coincide con i privilegi presenti nel comando, il comando viene eseguito totalmente e tutti i privilegi specificati vengono quindi concessi;
  - altrimenti, il comando viene eseguito parzialmente, solo per i privilegi contenuti nell'intersezione

Esempio



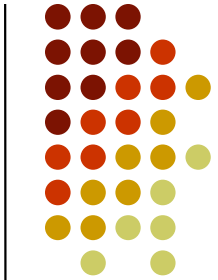
```
marco: GRANT  update(telefono) ON  Clienti TO
   roberto;

luca: GRANT delete  ON  Clienti TO  giovanni, anna;

barbara: GRANT select, insert ON Film TO alessandro;
```

Revoca ricorsiva



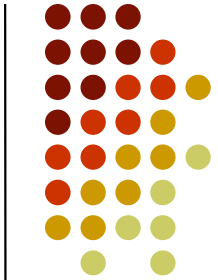
- Un problema di notevole interesse è quello della semantica da attribuire all'operazione di revoca dei diritti delegabili
- Il modello di controllo dell'accesso del System R adotta la cosidetta *revoca ricorsiva* (o a *cascata* o *basata su timestamp*)

Revoca ricorsiva



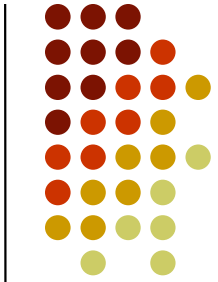
- Un'operazione di revoca del privilegio *p* concesso sulla relazione *rel* all'utente *u1* da parte dell'utente *u2* ha l'effetto non solo di far perdere ad *u1* il privilegio *p* sulla relazione *rel*, se *u1* non ha ottenuto tale privilegio da fonti indipendenti, ma anche di modificare l'insieme di autorizzazioni nel sistema portandolo in uno stato equivalente a quello in cui si sarebbe trovato se *u2* non avesse mai concesso ad u1 il privilegio *p* sulla relazione *rel*

Revoca ricorsiva



- Dopo un'operazione di revoca il sistema deve ricorsivamente revocare tutti i privilegi che non avrebbero potuto essere concessi se l'utente specificato nel comando di revoca non avesse mai ricevuto il privilegio revocato

Revoca ricorsiva



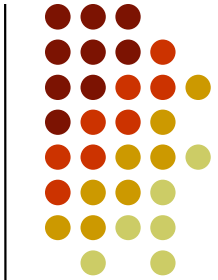
- Siano $G_1,\ldots,G_n$ una sequenza di operazioni di GRANT di un singolo privilegio sulla stessa relazione, tali che se $i<j$, $1\leq i,j, \leq n$, allora il comando G_i è eseguito prima di G_j.
- Sia R_i il comando che revoca il privilegio concesso con G_i
- La semantica della revoca ricorsiva impone che l'insieme di autorizzazioni nel sistema dopo l'esecuzione della sequenza do comandi:

$$G_1,\ldots,G_n,R_i$$

sia identico allo stato raggiunto dopo l'esecuzione della sequenza di comandi:

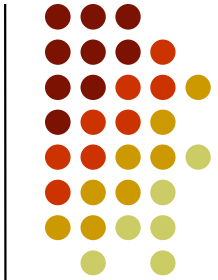
$$G_1,\ldots,G_{i-1},G_{i+1},\ldots,G_n$$

Grafo delle autorizzazioni



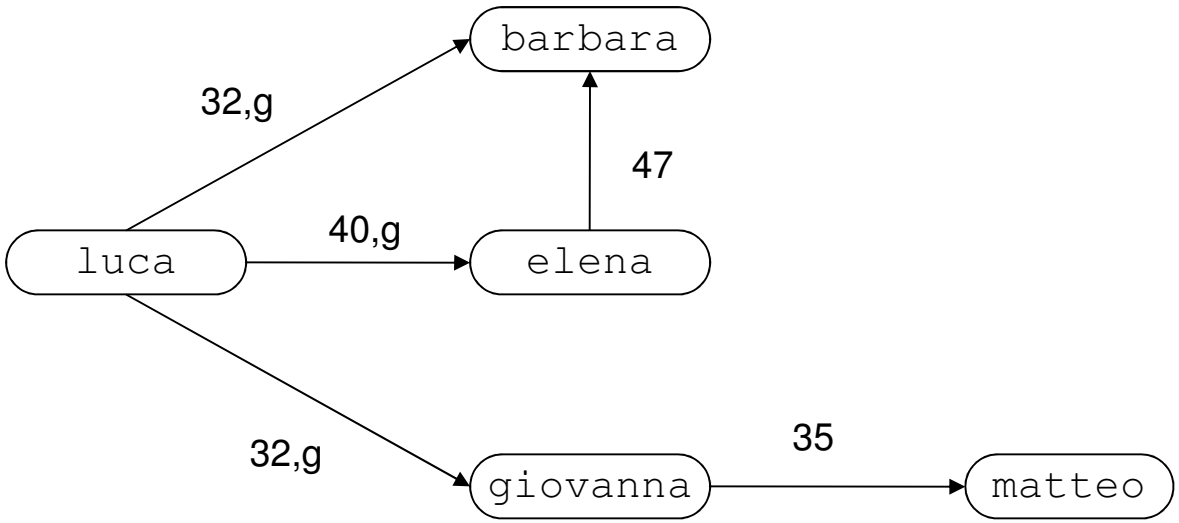
- Implementare l'operazione di revoca ricorsiva non è banale
- Per caratterizzare più precisamente il problema, è opportuno dare una rappresentazione a grafo dello stato delle autorizzazioni rispetto ad un dato privilegio *p* ed una data relazione *rel*

Grafo delle autorizzazioni



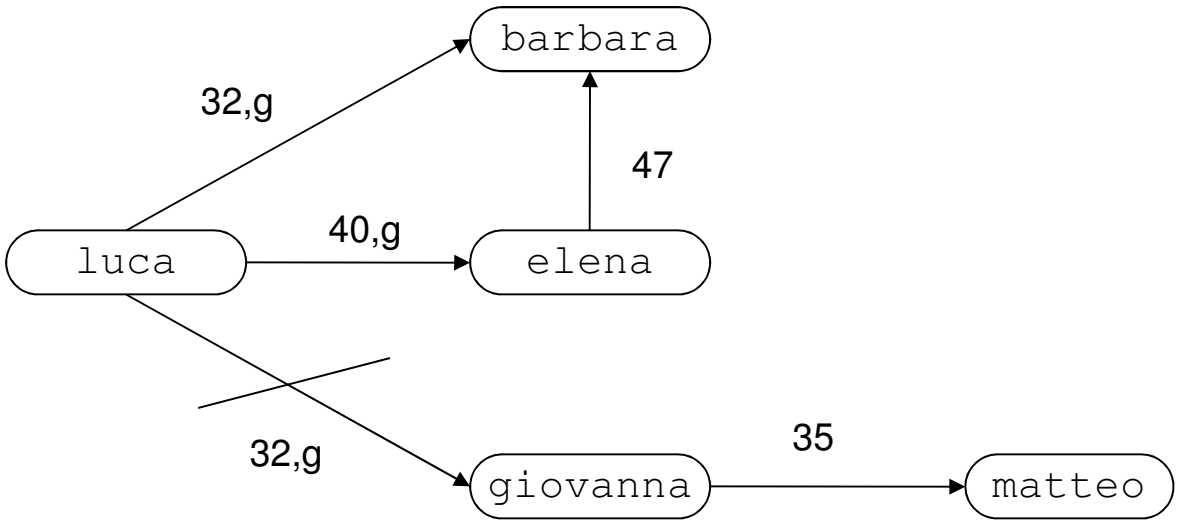
- Il grafo, chiamato *grafo delle autorizzazioni,* è definito in modo tale da:
  - contenere un nodo per ogni utente che possiede il privilegio *p* sulla relazione *rel*
  - contenere un arco dal nodo *u1* al nodo *u2* se l'utente *u1* ha concesso *p* ad *u2* su *rel*
    - l'arco è etichettato con il timestamp del privilegio e con la lettera *g* se il privilegio è delegabile

Esempio



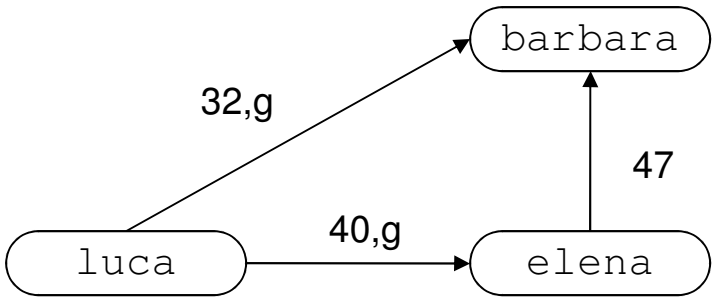
- Grafo delle autorizzazioni relativo al privilegio `select` e alla relazione `Film`

Esempio



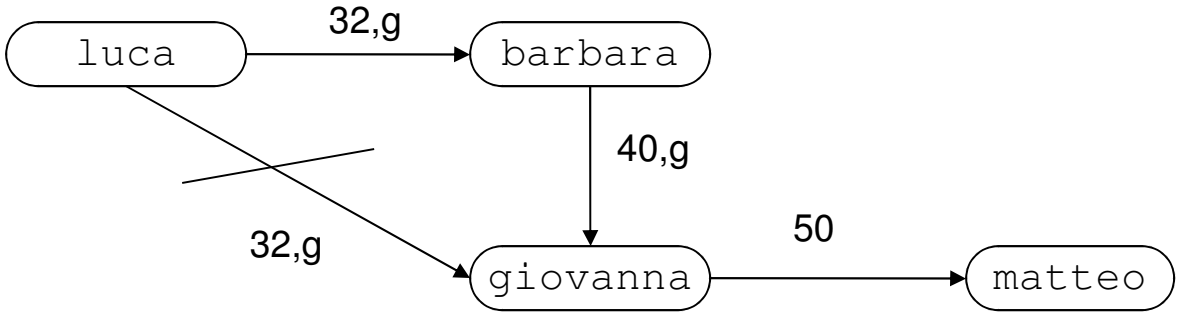
- Se Luca revocasse a Giovanna il privilegio `select` sulla relazione `Film`?

Esempio



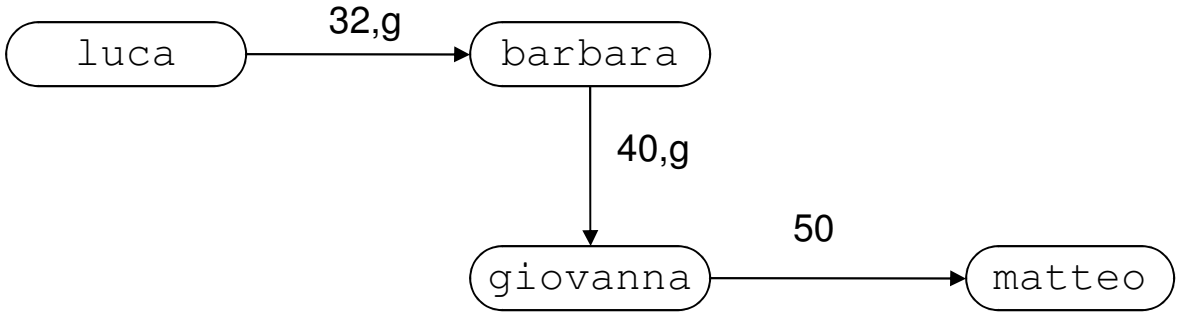
- Anche il privilegio concesso da Giovanna a Matteo sarebbe revocato in quanto non avrebbe mai potuto essere concesso se Luca non avesse concesso a Giovanna il privilegio per cui richiede la revoca.

Esempio



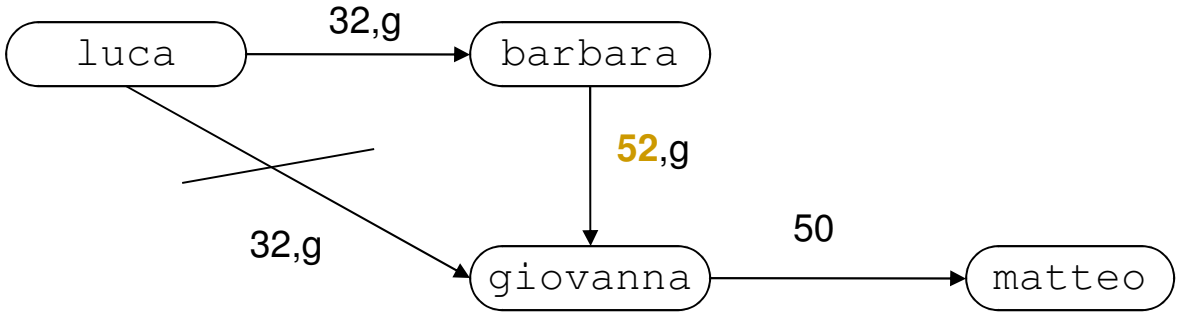
- Se Luca revocasse a Giovanna il privilegio `select` sulla relazione `Film`?

Esempio



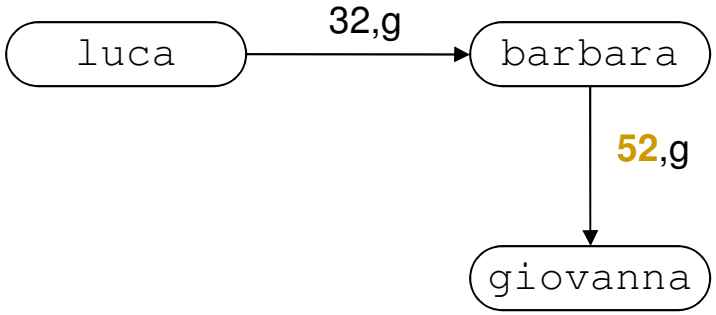
- Questo non comporta la revoca in cascata del privilegio `select` che Giovanna ha concesso a Matteo, in quanto tale privilegio avrebbe potuto essere concesso da Giovanna al tempo 50 anche se Giovanna non lo avesse ricevuto da Luca al tempo 32, in virtù del comando `GRANT` effettuato da Barbara al tempo 40

Esempio



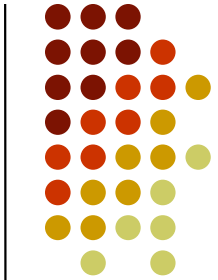
- Se Luca revocasse a Giovanna il privilegio `select` sulla relazione `Film`?

Esempio



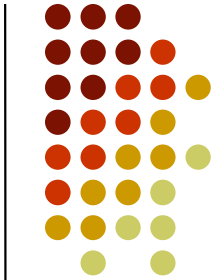
- La revoca effettuata da Luca comporterebbe anche la revoca del privilegio concesso da Giovanna a Matteo in quanto al tempo 50 Giovanna non avrebbe potuto concedere il privilegio `select` a Matteo se non lo avessericevuto da Luca.

Revoca ricorsiva



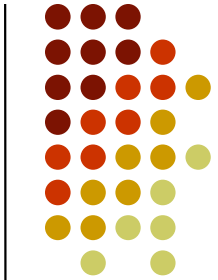
- La revoca ricorsiva è stata anche oggetto di critiche in quanto può portare alla cancellazione di un eccessivo numero di autorizzazioni
- Sono state proposte semantiche alternative a quella della revoca ricorsiva
  - *Revoca senza cascata*: quando un utente revoca un privilegio ad un altro utente, le autorizzazioni concesse da quest'ultimo in virtù del privilegio revocato non sono revocate ricorsivamente ma vengono rispecificate come se fossero state concesse dall'utente che ha richiesto l'operazione di revoca

Revoca ricorsiva



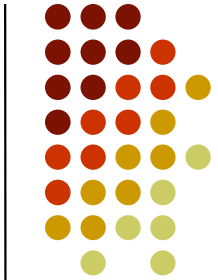
- A partire dallo standard SQL:1999, la revoca può essere richiesta *con o senza cascata*:
  - la revoca senza cascata prevede che un'operazione di revoca non venga eseguita se comporta la cancellazione di altre autorizzazioni, oltre a quella oggetto del comando
  - la revoca con cascata, invece, è una revoca ricorsiva, in cui però non sono considerati i timestamp delle autorizzazioni

Autorizzazioni su viste



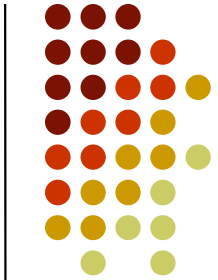
- Le viste sono un importante meccanismo attraverso cui è possibile fornire forme più sofisticate di controllo dell'accesso

Autorizzazioni su viste



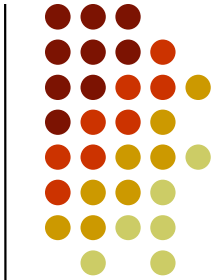
- La sintassi del comando GRANT non consente di concedere privilegi solo su alcuni attributi di una relazione, ad eccezione del privilegio `update`
  - ad esempio, non è possibile autorizzare un utente *u* a vedere solo le colonne relative al nome e cognome di clienti
- È però sufficiente definire una vista come proiezione sugli attributi su cui vogliamo concedere i privilegi ed autorizzare l'accesso alla vista invece che alla relazione di base
  - si crea una vista come proiezioni di nome e cognome dalla relazione clienti e si concede ad *u* il privilegio di select sulla vista

Autorizzazioni su viste



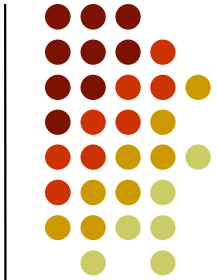
- Le viste permettono di concedere privilegi *statistici*
  - ad esempio, un utente potrebbe non essere autorizzato a vedere i titoli dei film noleggiati da un cliente, ma solo il numero di noleggi da lui effettuati
- Tramite le viste, è possibile soddisfare questo requisito di sicurezza:
  - basta definire una vista che computa il numero di noleggi effettuati da ogni cliente e concedere all'utente l'accesso alla vista invece che alle relazioni di base

Autorizzazioni su viste



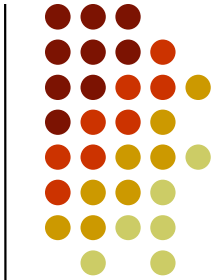
- Le viste consentono di realizzare il *controllo dell'accesso in base al contenuto*
- Ovvero autorizzare l'accesso solo a specifiche tuple di una relazione, sulla base dei valori dei loro attributi
  - ad esempio, se vogliamo autorizzare un utente a vedere solo le tuple della relazione `Film` relative a commedie, è sufficiente definire una vista che seleziona dalla relazione `Film` le tuple che soddisfano tale condizione e concedere all'utente il privilegio `select` sulla vista, invece che sulla relazione di base.

Autorizzazioni su viste



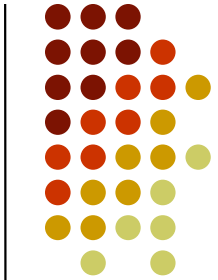
- Un utente può creare una vista solo se ha il privilegio `select` sulle relazioni/viste su cui è definita
- I privilegi che l'utente che crea una vista può esercitare sulla vista stessa dipendono da due fattori:
  - (i) le autorizzazioni che l'utente possiede sulle relazioni/viste su cui la vista è definita
  - (ii) la semantica della vista, ovvero la sua definizione in termini delle relazioni o viste componenti

Autorizzazioni su viste



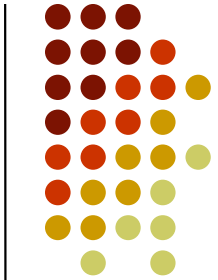
- Rispetto al punto (i): se la vista è definita su una singola relazione (o vista)
  - i privilegi che l'utente che crea la vista ha su di essa sono gli stessi che ha sulla relazione o vista componente
  - i privilegi sulla vista saranno delegabili o meno, a seconda di come sono definiti per la relazione o vista componente

Autorizzazioni su viste



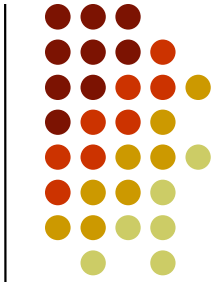
- Rispetto al punto (i): se la vista è definita su più relazioni (o viste)
  - i privilegi che l'utente che crea la vista ha su di essa sono ottenuti intersecando i privilegi che l'utente ha su tutte le relazioni (o viste) componenti
  - un privilegio sulla vista è delegabile solo se il creatore della vista ha il diritto di delegare tale privilegio su tutte le relazioni o viste componenti

Autorizzazioni su viste



- Rispetto al punto (ii):
  - SQL pone alcune restrizioni sulle operazioni che possono essere effettuate su una vista
    - ad esempio, una vista che computa delle statistiche non può essere aggiornata
  - queste restrizioni si riflettono anche sulle autorizzazioni che un utente che crea una vista ha sulla vista stessa

Esempio



Utente	nomeRel	Tipo	Select	Insert	Update	Grantor	Grantopt
Luca	Clienti	R	20	20	20		Y
Luca	Video	R	22	22	22		Y
Luca	Film	R	25	25	25		Y
Marco	Clienti	R	0	0	30	Luca	N
Barbara	Film	R	32	0	0	Luca	Y
Giovanna	Film	R	32	0	0	Luca	Y
Matteo	Film	R	35	0	0	Giovanna	N
Elena	Video	R	40	40	40	Luca	Y
Elena	Film	R	40	40	40	Luca	Y
Barbara	Film	R	47	47	0	Elena	N

Esempio

- supponiamo che Barbara esegua il comando:

```
CREATE VIEW Commedie  AS
SELECT *  FROM  Film
WHERE  genere = 'commedia';
```

- L'esecuzione di tale comando viene autorizzata in quanto Barbara ha il privilegio `select` su `Film`
- Barbara può esercitare sulla vista appena creata, privilegi di `select` e `insert` in quanto li possiede sulla relazione `Film`
- Barbara può concedere a terzi solo il privilegio di `select`, avendolo ricevuto da Luca con grant option

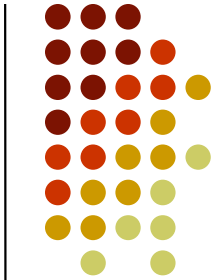
esempio

- supponiamo che Elena esegua il comando:

```
CREATE VIEW  NumFilm AS
SELECT   COUNT(*)
FROM  Film;
```

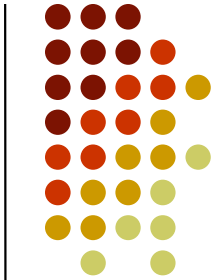
- Sulla relazione `Film` Elena ha i privilegi `select`, `insert` ed `update`
- Tutti questi privilegi dovrebbero essere esercitabili da Elena anche sulla vista, ma per il punto (ii), i privilegi di `update` ed `insert` non sono esercitabili:
  - Elena può quindi esercitare sulla vista solo il privilegio `select`
- Dato che Elena possiede tale privilegio con grant option sulla relazione `Film` potrà concedere a terzi l'autorizzazione di selezionare tuple dalla vista `NumFilm`

Autorizzazioni su viste



- La concessione di privilegi su una vista è molto simile a quella su relazioni di base:
  - i privilegi che un utente può concedere ad altri su una vista sono quelli che possiede con grant option
- Le operazioni di revoca sono, invece, più complicate, in quanto è necessario stabilire cosa succede ad una vista se un privilegio `select` su una delle relazioni o viste componenti è revocato
  - in accordo alla semantica della revoca ricorsiva, si cancella la vista se il privilegio revocato era l'unico utile per la sua definizione

Esempio



- Supponiamo che Barbara abbia eseguito il comando:

```
CREATE VIEW Commedie  AS
SELECT *  FROM  Film
WHERE  genere = 'commedia';
```

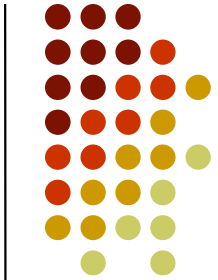
- mentre Elena:

```
CREATE VIEW  NumFilm AS
SELECT   COUNT(*)
FROM  Film;
```

- Se Luca revoca ad Elena il privilegio `select` sulla relazione `Film`
  - la vista NumFilm viene a sua volta cancellata
- Se Luca revoca a Barbara il privilegio `select` sulla relazione `Film`, la decisione se cancellare o meno la vista `Commedie` dipenderebbe dal timestamp del comando `CREATE VIEW`
  - se il timestamp è maggiore di 47 la vista non viene cancellata in quanto Barbara ha ricevuto da Elena il privilegio `select` su `Film` al tempo 47
  - altrimenti la vista viene ricorsivamente revocata.

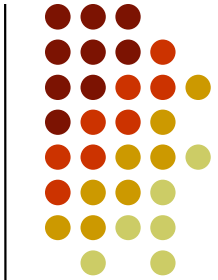
Controllo dell'accesso basato sui ruoli

Modelli basati suoi i ruoli



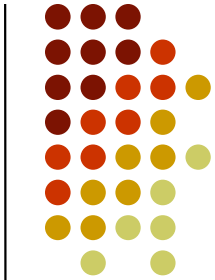
- Nel *controllo dell'accesso basato su ruoli* (RBAC - Role-based Access Control) i ruoli rappresentano delle funzioni che gli utenti ricoprono all'interno dell'organizzazione o azienda in cui operano
  - esempi di ruolo per il dominio della videoteca: commesso, cliente e direttoreVideoteca

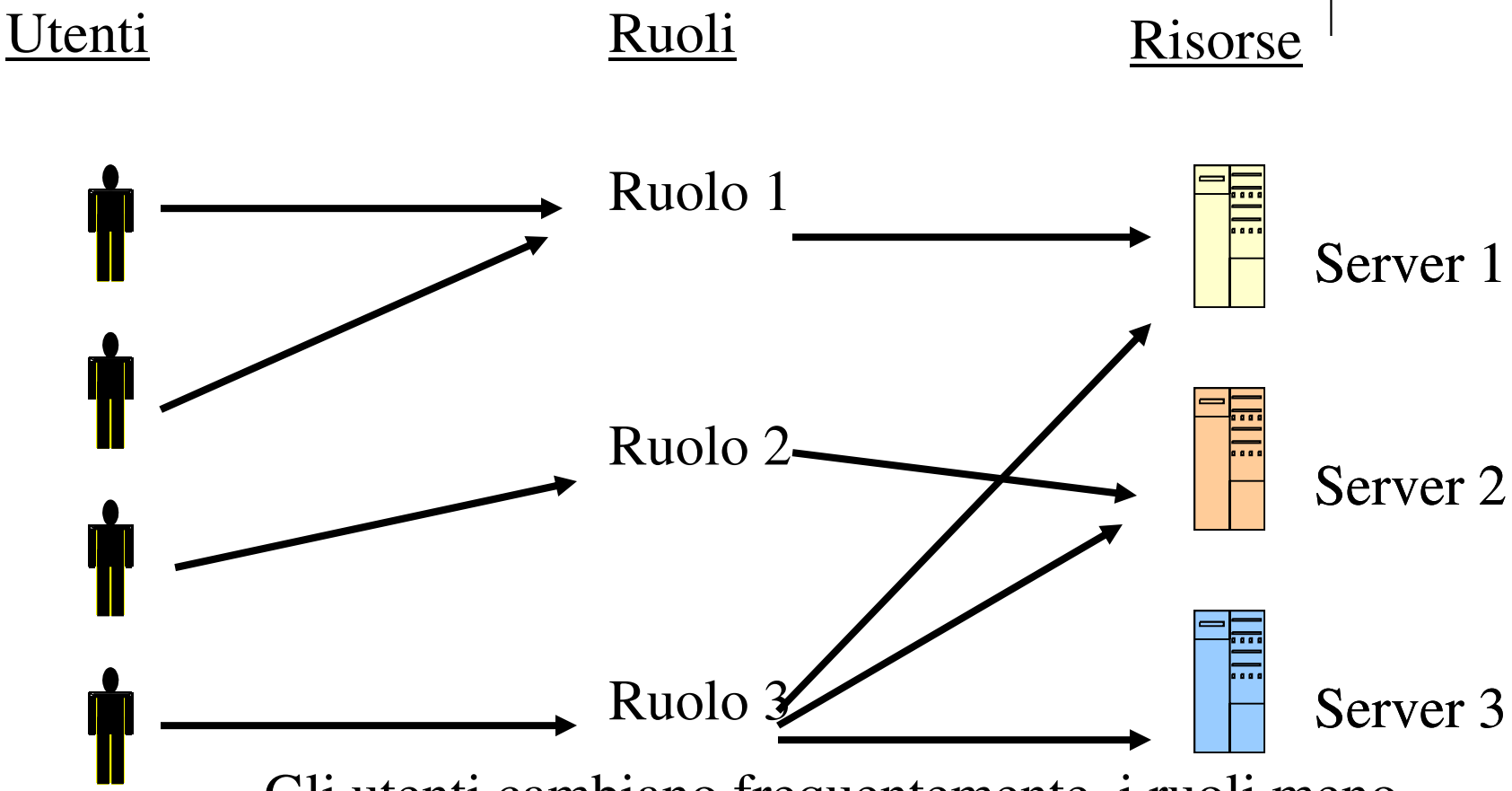
Modelli basati suoi i ruoli



- I privilegi sono concessi ai ruoli invece che ai singoli utenti
  - le autorizzazioni specificate per un ruolo sono quelle necessarie per esercitare le funzioni connesse al ruolo stesso
- Gli utenti sono abilitati a ricoprire uno o più ruoli, in base alle mansioni che devono svolgere
  - l'abilitazione a ricoprire un ruolo implica l'acquisizione di tutte le autorizzazioni ad esso connesse

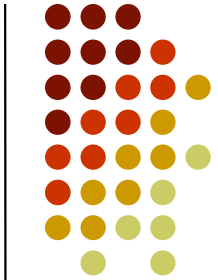
RBAC





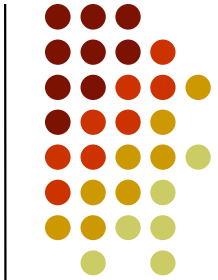
Gli utenti cambiano frequentemente, i ruoli meno

Modelli basati suoi i ruoli



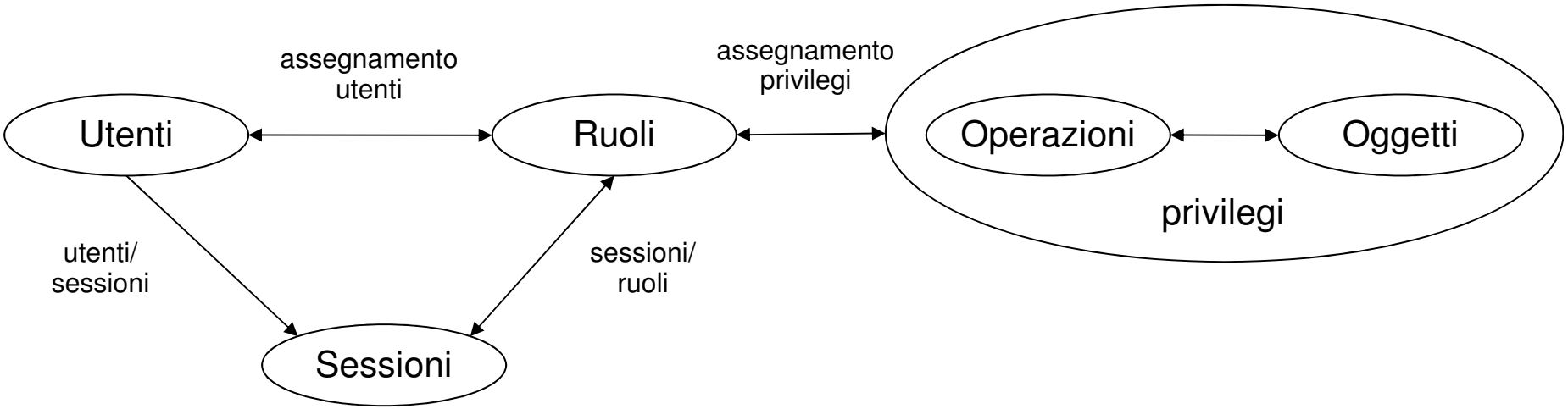
- Per rendere più agevole lo sviluppo di meccanismi per il controllo dell'accesso basati su ruoli, è stato definito uno standard NIST che introduce un modello di riferimento
- È costituito da tre componenti:
  - *modello base* - Core RBAC: definisce i requisiti minimi per realizzare il controllo dell'accesso basato su ruoli
  - *modello gerarchico* - Hierarchical RBAC: aggiunge al modello base la strutturazione gerarchica dei ruoli
  - *modello con vincoli* - Constrained RBAC: permette di specificare vincoli sull'attivazione e sull'assegnazione dei ruoli

Modello base

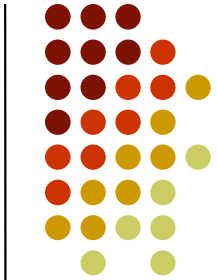


- Il modello base si compone di quattro componenti principali:
  - *utente*: un essere umano, una macchina, un processo, un agente attivo nel sistema
  - *ruolo*: una funzione all'interno di un contesto organizzativo, con associati un insieme di privilegi
  - privilegi: sono i diritti d'accesso esercitabili sugli oggetti del sistema. Sono definiti come coppie (*obj,o*), dove *obj* é un oggetto ed *o* è un'operazione
  - sessione: stabilisce la corrispondenza tra un utente ed i ruoli attivi durante la sua connessione al sistema

Modello base

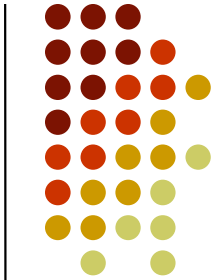


Modello gerarchico



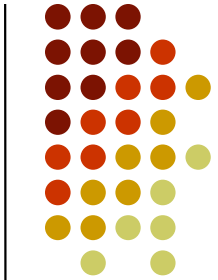
- Il modello gerarchico aggiunge al modello base la possibilità di strutturare i ruoli in gerarchie
- Una gerarchia sui ruoli definisce un ordinamento parziale tra di essi:
  - induce un'ereditarietà dei privilegi tra ruoli nella gerarchia
  - stabilisce una relazione tra gli utenti abilitati a ricoprire i vari ruoli

Modello gerarchico



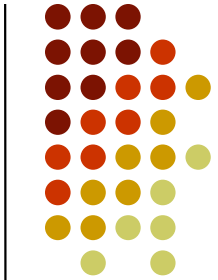
- E’ definita una relazione d'ordine parziale sull'insieme dei ruoli, chiamata relazione di ereditarietà e denotata con ≥, tale per cui, dati due ruoli *r1* ed *r2*, *r1*≥*r2* implica che:
  - *r1* eredita tutti i privilegi assegnati ad *r2*
  - tutti gli utenti associati ad *r1* sono anche utenti associati ad *r2*

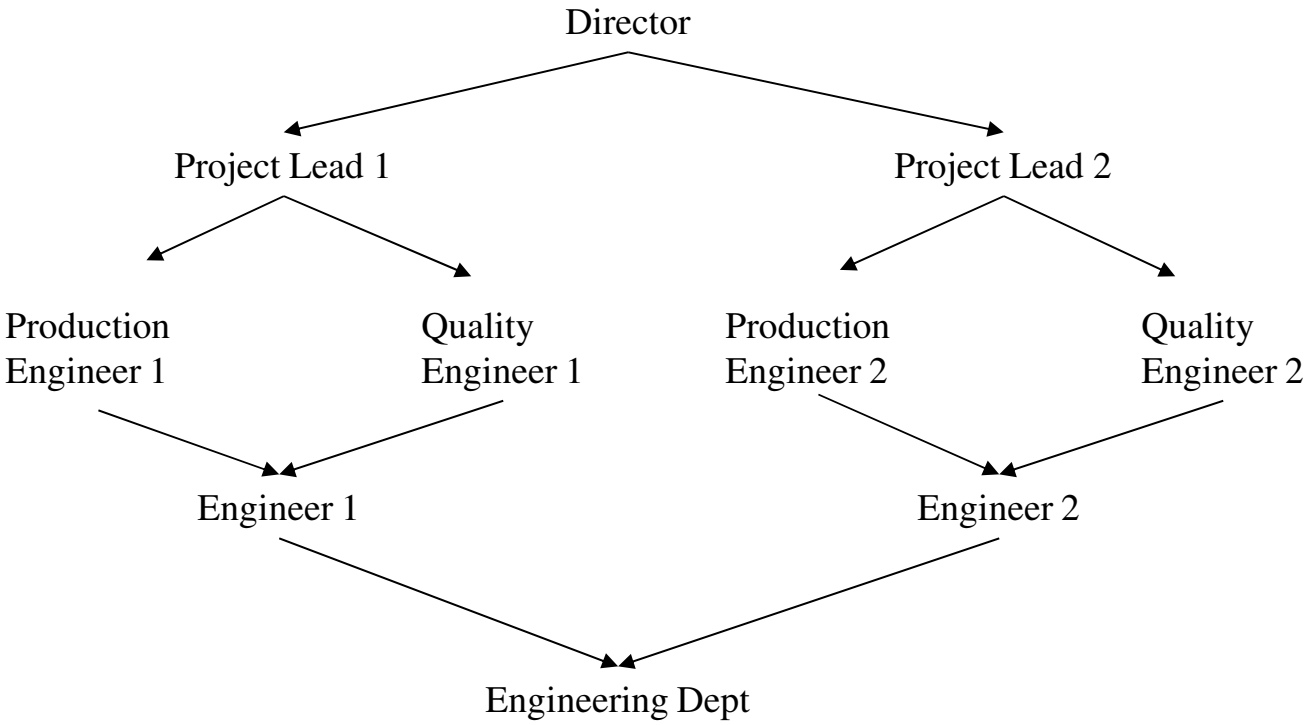
Modello gerarchico



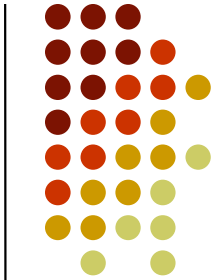
- Il principio di ereditarietà è motivato dal fatto che un ruolo dovrebbe poter effettuare sugli oggetti del sistema tutte le operazioni che possono essere effettuate da ruoli corrispondenti a funzioni più in basso nella gerarchia

Modello gerarchico



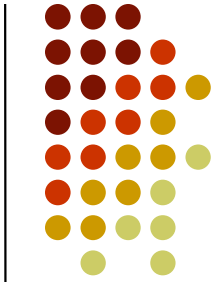


Modello con vincoli



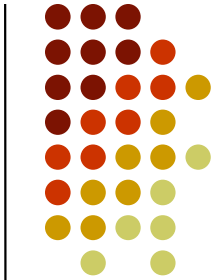
- Il modello con vincoli aggiunge al modello base la possibilità di specificare vincoli
- Lo standard attuale considera solo una tipologia di vincolo, nota come separazione delle mansioni (SoD -Separation of Duties)
- I vincoli per la separazione delle mansioni possono essere classificati in due categorie:
  - *vincoli statici*
  - *vincoli dinamici*

Modello con vincoli



- I vincoli statici stabiliscono delle relazioni di mutua esclusione tra ruoli assegnabili ad un certo utente
  - ad esempio, la politica aziendale potrebbe impedire che un utente sia assegnato contemporaneamente al ruolo *segretarioAmministrativo* e *revisoreConti*, in quanto il secondo esercita una funzione di controllo sul primo
- Un vincolo di tipo statico è formalizzato come una coppia (*RS,n*), dove *RS* è un sottoinsieme dei ruoli previsti dal modello ed *n* è un numero naturale
- Se i ruoli sono strutturati in gerarchia, i vincoli di separazione delle mansioni vengono propagati lungo la gerarchia

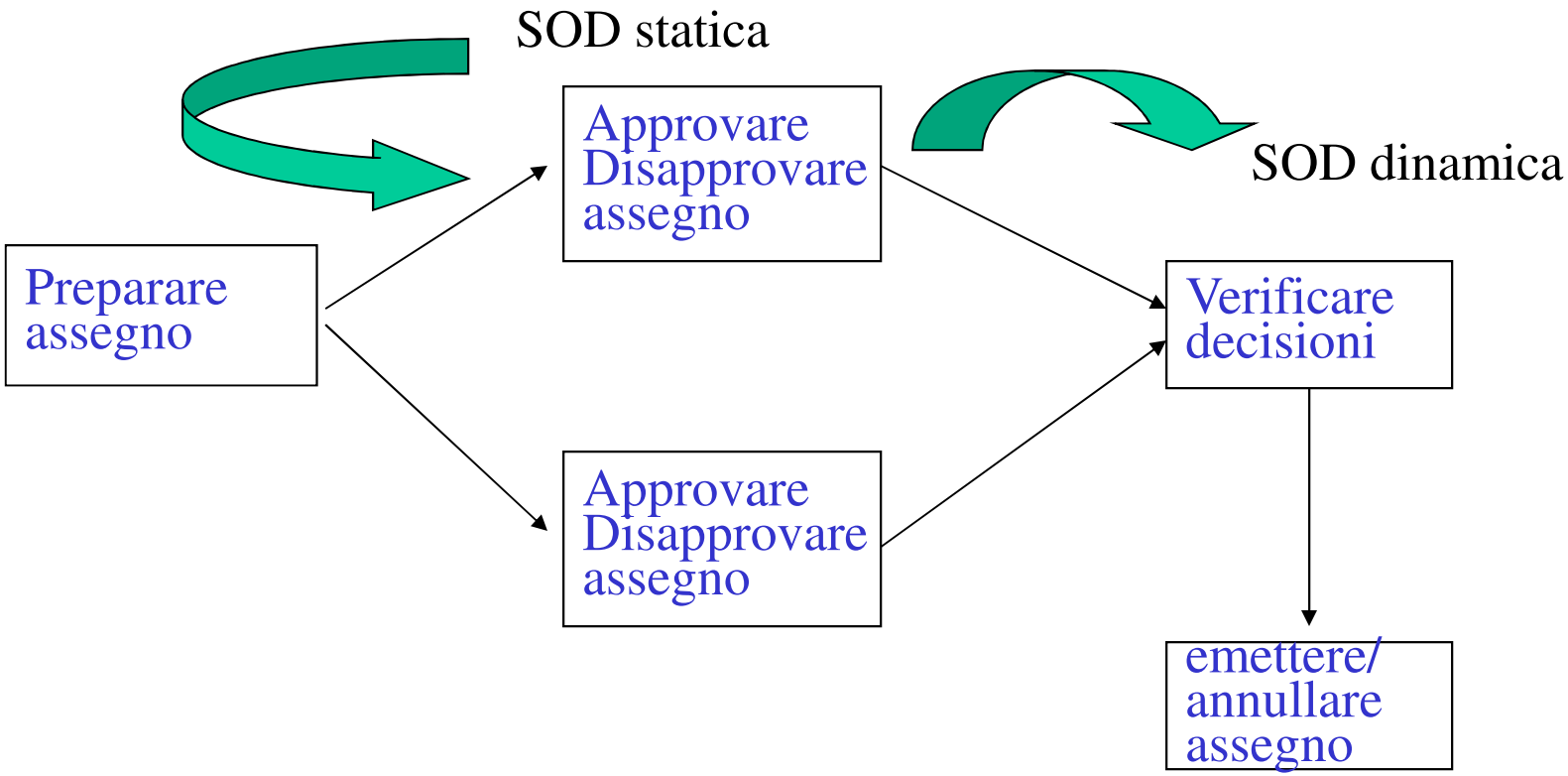
Modello con vincoli



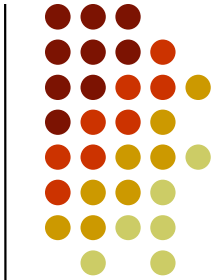
- I vincoli dinamici di separazione delle mansioni consentono di imporre delle restrizioni sull'insieme dei ruoli che un utente può attivare in una sessione
  - ad esempio, un utente potrebbe essere autorizzato ad attivare il ruolo *cassiere* e *supervisoreCassiere* in due momenti diversi, ma non contemporaneamente nella stessa sessione
- Un vincolo di tipo dinamico è formalizzato come i vincoli statici -- una coppia (*RS,n*), ma con semantica diversa. Tale vincolo dinamico impone che nessun utente possa attivare contemporaneamente un numero di ruoli maggiore o uguale ad *n* tra quelli in *RS*

Modello con vincoli

Rimborso tasse

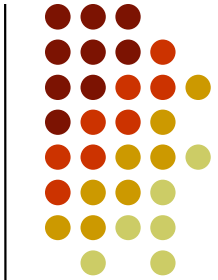


RBAC: alcune questioni aperte



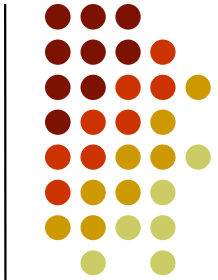
- Amministrazione dei ruoli
- Role engineering
- RBAC per nuovi domini applicativi
- Standard per la specifica di vincoli

Amministrazione dei ruoli



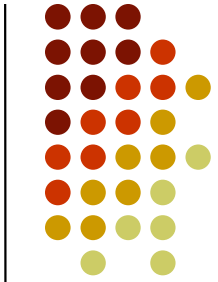
- Sistemi RBAC reali possono avere centinaia di ruoli e migliaia di utenti
  - Un case study effettuato alla Dresdner Bank ha prodotto un sistema con circa 40.000 utenti e 1.300 ruoli
- Necessità di tecniche per l'amministrazione decentralizzata dei ruoli
- Al momento esistono molte proposte di ricerca (ARBAC97, ARBAC00, SARBAC, UARBAC, …) ma non ancora uno standard

Amministrazione dei ruoli



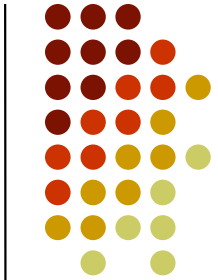
- Idea generale
  - Usare i ruoli anche per la loro amministrazione
  - Ruoli amministrativi da utilizzare per l'amministrazione dei ruoli standard
  - Ad ogni ruolo amministrativo viene assegnato uno scope
  - Problema: molti side-effect indesiderati (ad es. cicli nella gerarchia dei ruoli) richiedono una gestione centralizzata di molte operazioni amministrative

Role engineering e role mining



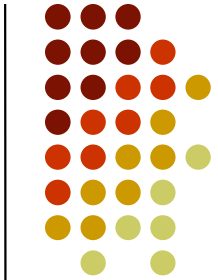
- Role engineering: il problema di indentificare un insieme di ruoli completo, corretto ed efficiente
- Linea di tendenza --- role mining:
  - Tecniche di data mining sono usate per derivare automaticamente i ruoli dall'analisi dei dati legati al controllo dell'accesso, in particolare dall'analisi dei permessi assegnati agli utenti

Role mining



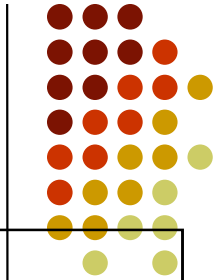
- Problema principale: come misurare la “bontà” dell’insieme dei ruoli identificati
  - E’ stato provato che tale problema è NP-completo
  - Definizione di euristiche per ottenere implementazioni con una complessità sostenibile e buoni risultati
  - Definizione di criteri standard per valutare le varie tecniche fino ad ora proposte

Estensioni di RBAC



- TRBAC e sue estensioni: RBAC + vincoli di cardinalità e temporali sulla attivazione/disattivazione di ruoli
- Geo-RBAC: RBAC + informazioni di localizzazione, tramite cui i ruoli sono attivati/disattivati in base alla posizione dell'utente
- PRBAC: RBAC + politiche e preferenze di privacy

TRBAC



([1/1/10,12/31/11], night-time, VH:activate doctor-on-night-duty)

([1/1/10,12/31/11], day-time, VH:deactivate doctor-on-night-duty)

([1/1/10,12/31/11], day-time, VH:activate doctor-on-day-duty)

([1/1/10,12/31/11], night-time, VH:deactivate doctor-on-day-duty)

activate doctor-on-night-duty ⟶ H: activate nurse-on-nigth-duty

deactivate doctor-on-night-duty ⟶ H: deactivate nurse-on-nigth-duty

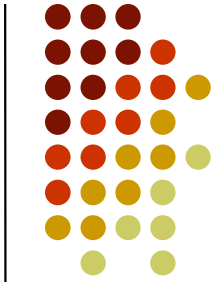
activate doctor-on-day-duty ⟶ H: activate nurse-on-day-duty

deactivate doctor-on-day-duty ⟶ H: deactivate nurse-on-day-duty

activate nurse-on-day-duty ⟶ H: activate nurse-on-training after 2 Hours

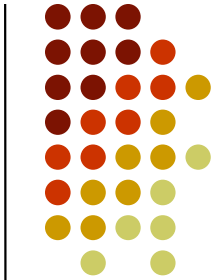
deactivate nurse-on-day-duty ⟶ VH: deactivate nurse-on-training

Geo-RBAC



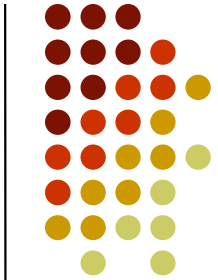
- Basato sulla nozione si ruolo spaziale (spatial role):
  - Ruolo con aggiunta una regione (ospedale, strada) che indica i confini geografici entro cui può essere attivato

Case Study: Dresdner Bank



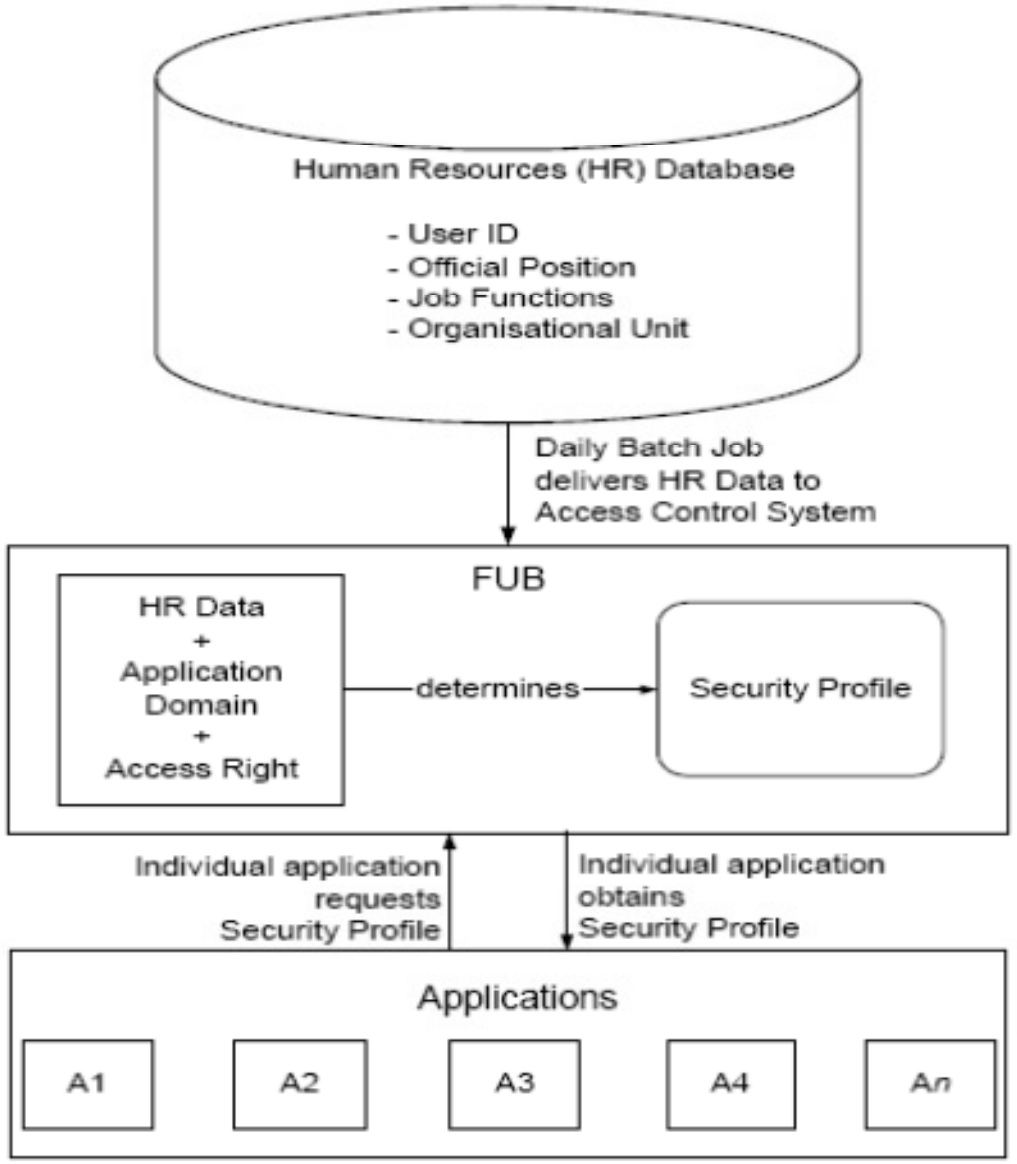
- 50.659 impiegati, 1459 sedi sparse in tutto il mondo (la maggioranza in Germania e Europa), 6.5 milioni di clienti privati
- Pre-esistente:
  - applicazioni eterogenee migrate lentamente da mainframe a client-server
  - Accessi: gestiti per ogni singolo utente a livello di applicazione

Case Study: Dresdner Bank

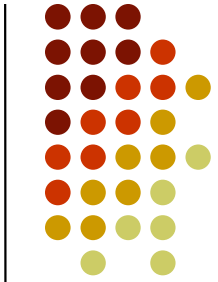


- Sviluppo di FUB (Funktionale Berechtigung), sistema che implementa RBAC e sostituisce la gestione della sicurezza a livello applicazione
- Le applicazioni non possono più gestire in proprio i diritti di accesso ma devono farlo in base ad un profilo di sicurezza generato da FUB
- FUB è usato da 60 applicazioni all'interno della banca e distribuisce una media di 42.000 profili al giorno, con un tempo medio di risposta di 85ms

FUB

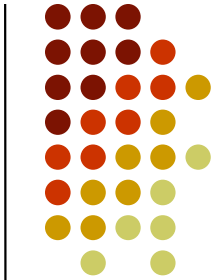

Human Resources (HR) Database
- User ID
- Official Position
- Job Functions
- Organisational Unit
Daily Batch Job
delivers HR Data to
Access Control System
FUB
HR Data
+
Application
Domain
+
Access Right
determines
Security Profile
Individual application
requests
Security Profile
Individual application
obtains
Security Profile
Applications
A1
A2
A3
A4
An

FUB: un esempio di utilizzo



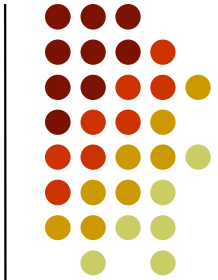
- Un cliente vuole discutere la situazione dei suoi depositi con il suo consulente
- Il consulente si indentifica tramite una smartcard e una password e lancia l'applicazione che gli consente la visualizzazione dei conti dei suoi clienti
- L'applicazione interroga FUB per determinare i diritti di accesso del consulente all'interno dell'applicazione, mandando l'id del consulente ottenuto durante l'autenticazione e l'id dell'applicazione
- FUB restituisce un profilo di sicurezza che determina le operazioni consentite al consulente

FUB: definizione dei ruoli



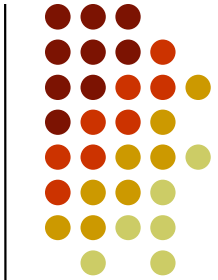
- I ruoli sono definiti considerando:
  - La gerarchia organizzativa
  - Le funzioni specifiche
- Questi dati sono resi disponibili da HR
- Nella banca ci sono 65 qualifiche ufficiali (segretaria, componente del consiglio di amministrazione, ecc.) e 368 funzioni (analista, finanziario, supporto all'e-commerce, ecc.)
  - Teoricamente 23.920 ruoli
  - Nella pratica: 1300 in quanto alcune combinazioni non hanno senso

FUB: ruoli



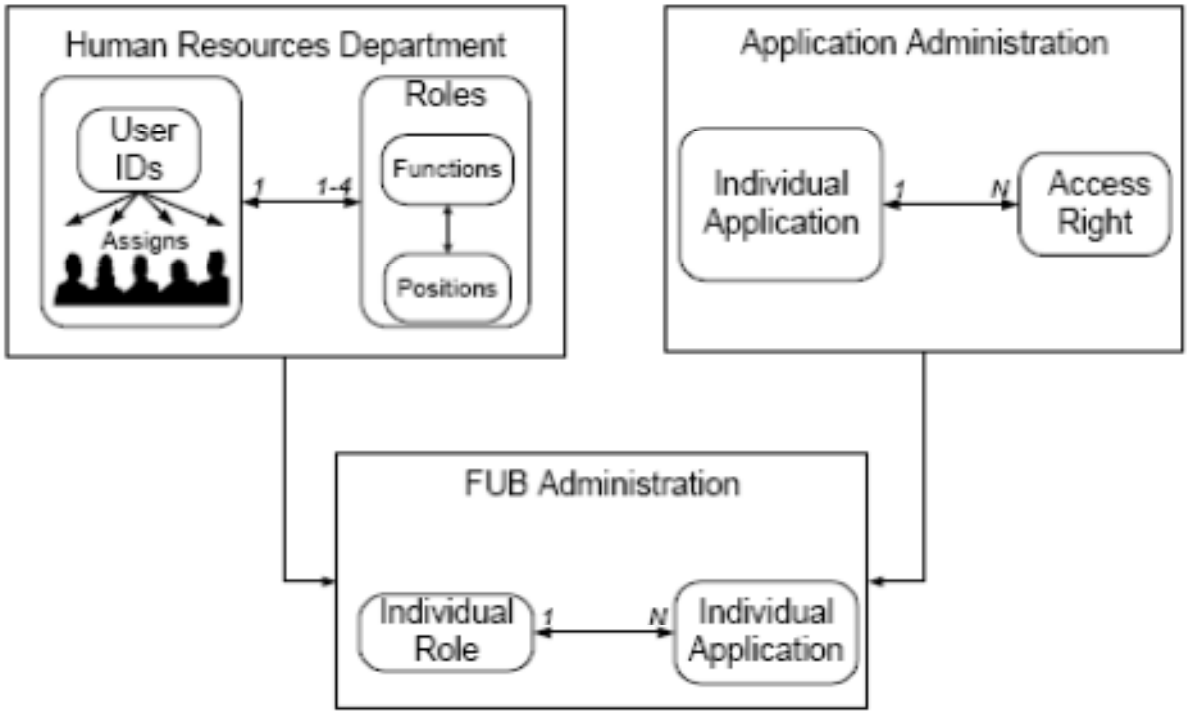
- Ogni notte FUB viene aggiornato per riflettere i cambiamenti nel db di HR
  - 40.000 utenti usano FUB
  - I ruoli sono il 3.2% degli utenti
  - In linea con il trend generale che stima i ruoli essere il 3-4% del numero totale di utenti

FUB: amministrazione



- Effettuata a vari livelli per realizzare SoD:
  - HR: definizione dei ruoli e assegnamento agli utenti
  - Application Administrator: definizione dei diritti di accesso e assegnamento diritti alle applicazioni
  - FUB Administration: Assegnamento Ruoli/Applicazioni

FUB: amministrazione



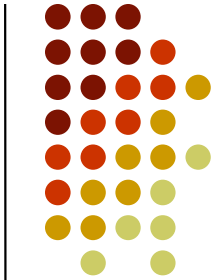
FUB: amministrazione

Role	Function	Official Position
A	financial analyst	Clerk
B	financial analyst	Group Manager
C	financial analyst	Head of Division
D	financial analyst	Junior
E	financial analyst	Senior
F	financial analyst	Specialist
G	financial analyst	Assistant
…	…	…
X	share technician	Clerk
Y	support e-commerce	Junior
Z	office banking	Head of Division

Role	Application	Access Right
A	Money Market Instruments	1,2,3,4
	Derivatives Trading	1,2,3,7,10,12
	Interest Instruments	1,4,8,12,14,16
B	Money Market Instruments	1,2,3,4,7
	Derivatives Trading	1,2,3,7,10,12,14
	Interest Instruments	1,4,8,12,14,16
	Private Customer Instruments	1,2,4,7
…	…	…

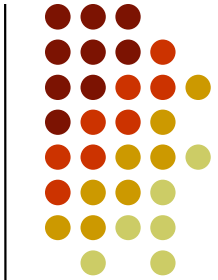
Controllo dell'accesso in SQL

Controllo dell'accesso in SQL



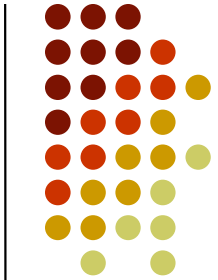
- Per quanto riguarda il controllo dell'accesso, SQL è basato sul modello del System R, con le seguenti differenze ed estensioni:
  - supporto per i ruoli
  - una diversa semantica per l'operazione di revoca
  - l'aggiunta di nuovi privilegi ed oggetti di autorizzazione
- Nel seguito ci focalizzeremo sulle differenze rispetto al modello del System R

Controllo dell'accesso in SQL



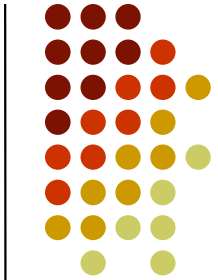
- Creazione di un ruolo:
  - `CREATE ROLE <nome ruolo>;`
- Eliminazione di un ruolo:
  - `DROP ROLE <nome ruolo>;`

Comando GRANT



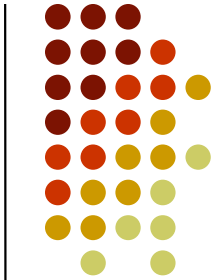
- Il comando GRANT del System R è stato esteso con la possibilità:
  - di concedere privilegi non solo ad utenti ma anche a ruoli
  - di autorizzare un utente non solo all'esercizio di privilegi ma anche a ricoprire uno o più ruoli

Comando GRANT



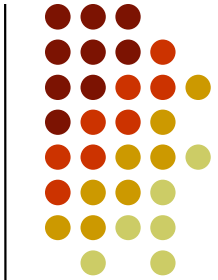
```
GRANT {<lista privilegi>| ALL PRIVILEGES}
ON [<qualificatore oggetto>] <nome oggetto>
TO {<lista utenti>|<lista  ruoli> |PUBLIC}
[WITH GRANT OPTION]| [WITH HIERARCHY OPTION];
```

Comando GRANT



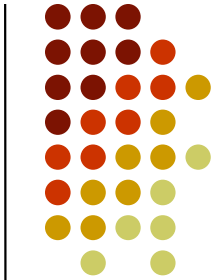
- `<lista privilegi>` indica l'insieme dei privilegi concessi con il comando GRANT. La parola chiave ALL PRIVILEGES indica tutti i privilegi previsti dal modello
- `<nome oggetto>` indica il nome dell'oggetto della base di dati su cui sono concessi i privilegi
  - Se necessario è possibile specificare un qualificatore dell'oggetto
- `<lista ruoli>` indica l'insieme dei ruoli a cui vengono concessi i privilegi. La parola chiave `PUBLIC` consente di specificare tutti gli utenti/ruoli del sistema
- La clausola opzionale `WITH GRANT OPTION` consente la delega dell'amministrazione dei privilegi
- La clausola opzionale `WITH HIERARCHY OPTION` può essere specificata solo per il privilegio `select` e consente, nel caso di relazioni legate da una gerarchia di ereditarietà di propagare il privilegio a tutte le sotto-tabelle della tabella riferita nel comando `GRANT`

Comando GRANT



- Estensioni all'insieme dei privilegi ed oggetti di autorizzazione rispetto a quelli previsti dal modello del System R:
  - i privilegi `select` e `insert` possono essere concessi selettivamente solo su alcune colonne di una relazione
  - sono supportati altri privilegi:
    - `references`, che permette di utilizzare una colonna in un vincolo o asserzione
    - `trigger`, che permette di specificare trigger che operano su una certa relazione
    - `under`, che permette la creazione di sotto-tipi o sotto-tabelle
    - `usage`, che permette di utilizzare un oggetto dello schema nella definizione di un altro oggetto
    - `execute`, che permette l'esecuzione di una procedura o funzione.

Comando GRANT

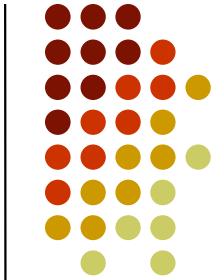


- Il comando Grant consente anche di abilitare utenti/ruoli a ricoprire ruoli

```
GRANT <lista ruoli concessi>
TO {<lista utenti>|<lista  ruoli> |PUBLIC}
[WITH ADMIN OPTION];
```

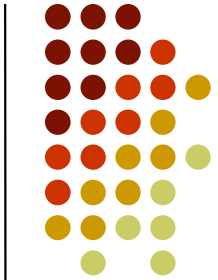
- `<lista ruoli concessi>` indica l'insieme dei ruoli concessi con il comando GRANT
- `<lista utenti>` indica l'insieme degli utenti per cui vengono abilitati i ruoli concessi con il comando
- `<lista ruoli>` indica l'insieme dei ruoli per cui vengono abilitati i ruoli concessi con il comando. La parola chiave PUBLIC consente di abilitare i ruoli specificati nel comando per tutti gli utenti/ruoli del sistema

Comando GRANT



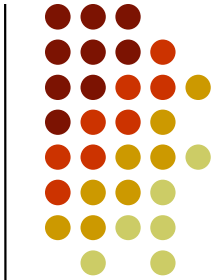
- La clausola opzionale WITH ADMIN OPTION é l'analogo per i ruoli della grant option.Quando si è abilitati a ricoprire un ruolo con admin option non solo vengono ereditate tutte le autorizzazioni specificate per quel ruolo, ma è anche possibile concedere a terzi la possibilità di ricoprire il ruolo
- La sintassi del comando GRANT permette di abilitare un ruolo a ricoprirne un altro (ereditando quindi le sue autorizzazioni). Questo é il modo con cui SQL fornisce implicitamente la possibilità di strutturare i ruoli in gerarchia

Esempio



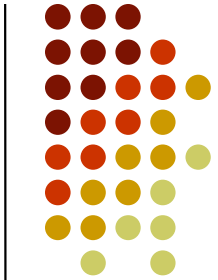
```
GRANT usage  ON TYPE indirizzo TO giovanna WITH
GRANT OPTION;

GRANT execute ON aggiornaClienti TO elena;

GRANT select(nome, cognome), REFERENCES(codCli) ON
Clienti TO marco;

GRANT direttoreVideoteca  TO roberto WITH ADMIN
OPTION;

GRANT delete, update ON Clienti TO
direttoreVideoteca;
```

Comando REVOKE



- Anche per il comando REVOKE vi è una duplice sintassi:
  - per la revoca di privilegi
  - per la revoca di ruoli

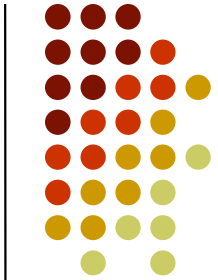
Comando REVOKE



Il comando per la revoca dei privilegi ha la seguente sintassi

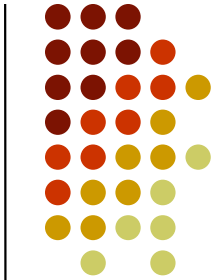
```
REVOKE [{GRANT OPTION FOR| HIERARCHY OPTION FOR} <lista
privilegi>]
ON [<qualificatore oggetto>] <nome oggetto>
FROM {<lista utenti>|<lista  ruoli>}
{RESTRICT | CASCADE};
```

Comando REVOKE



- Le clausole opzionali `GRANT OPTION FOR` e `HIERARCHY OPTION FOR` servono per revocare la sola grant o hierarchy option, mantenendo il diritto ad esercitare i privilegi oggetto del comando di revoca
- `<lista privilegi>` indica l'insieme di privilegi oggetto del comando di revoca
- `<nome oggetto>` indica il nome dell'oggetto della base di dati su cui sono revocati i privilegi
- `<lista utenti>` indica l'insieme degli utenti a cui sono revocati i privilegi
- `<lista ruoli>` indica l'insieme dei ruoli a cui sono revocati i privilegi
- Le clausole `RESTRICT` e `CASCADE` servono per gestire l'operazione di revoca

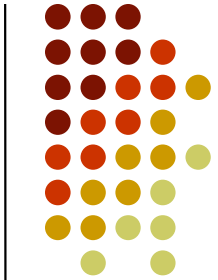
Comando REVOKE



Il comando per la revoca dei ruoli ha la seguente sintassi

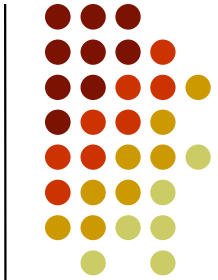
```
REVOKE [ADMIN OPTION FOR] <lista ruoli revocati>
FROM {<lista utenti>|<lista  ruoli>}
{RESTRICT | CASCADE};
```

Comando REVOKE



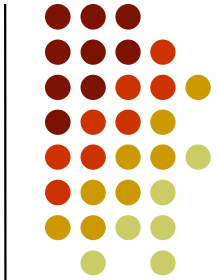
- La clausola opzionale `ADMIN OPTION FOR` serve per revocare la sola admin option, mantenendo il diritto a ricoprire i ruoli oggetto del comando di revoca
- `<lista ruoli revocati>` indica l'insieme di ruoli revocati con il comando
- `<lista utenti>` indica l'insieme degli utenti a cui viene revocata l'abilitazione a ricoprire i ruoli oggetto della revoca
- `<lista ruoli>` indica l'insieme dei ruoli a cui viene revocata l'abilitazione a ricoprire i ruoli oggetto della revoca.
- Le clausole `RESTRICT` e `CASCADE` servono per gestire l'operazione di revoca

Esempio



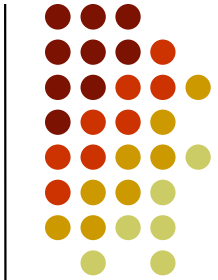
```
REVOKE GRANT OPTION FOR usage ON TYPE
indirizzo FROM giovanna;

REVOKE delete ON Clienti FROM
direttoreVideoteca;

REVOKE direttoreVideoteca FROM roberto;
```

Comando REVOKE



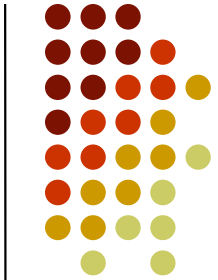
- Una delle differenze più importanti dello standard SQL rispetto al modello di controllo dell'accesso del System R riguarda la semantica da attribuire all'operazione di revoca
- Lo standard prevede due possibilità: `RESTRICT` e `CASCADE`

Comando REVOKE



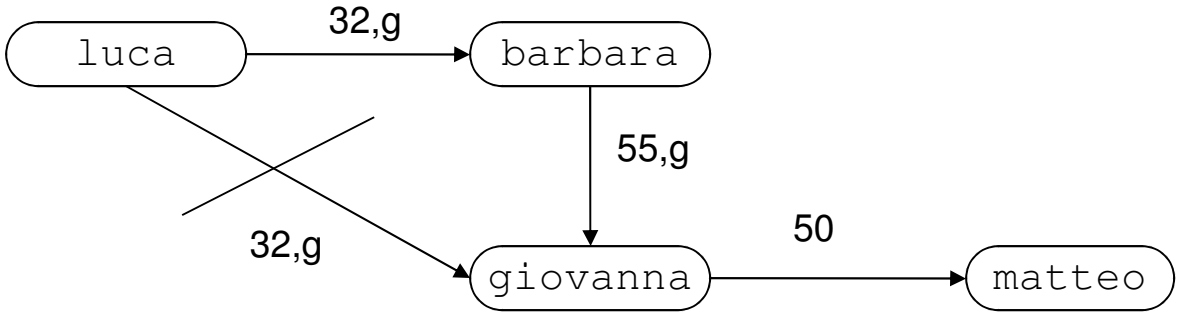
- Se la revoca di un privilegio/ruolo è richiesta con l'opzione `RESTRICT`, allora l'esecuzione del comando non viene concessa se questo comporta la revoca di altri privilegi, oppure la cancellazione di oggetti dello schema (ad esempio nel caso di viste create grazie al privilegio revocato).

Comando REVOKE



- Se la revoca di un privilegio è richiesta con l'opzione `CASCADE`, allora viene implementata una revoca simile alla revoca ricorsiva del System R ma senza considerare i timestamp:
  - le autorizzazioni specificate da un utente, a cui è stato revocato un privilegio, non sono ricorsivamente revocate se l'utente ha ricevuto da altre fonti indipendenti il privilegio specificato nel comando `REVOKE` con grant option, indipendentemente dal tempo in cui ha ricevuto il privilegio; sono ricorsivamente revocate, in caso contrario.

esempio



- Supponiamo che Luca revochi a Giovanna il privilegio concessole al tempo 32
- Questo non comporterebbe la revoca del privilegio concesso da Giovanna a Matteo, in quanto Giovanna continua ad avere il privilegio `select` con grant option sulla relazione `Film` concessole da Barbara