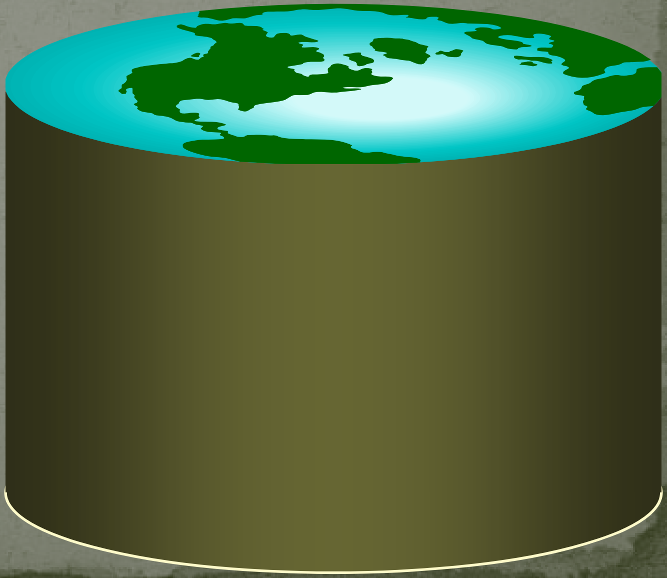
CS 505: Intermediate Topics to Database Systems

Instructor: Jinze Liu

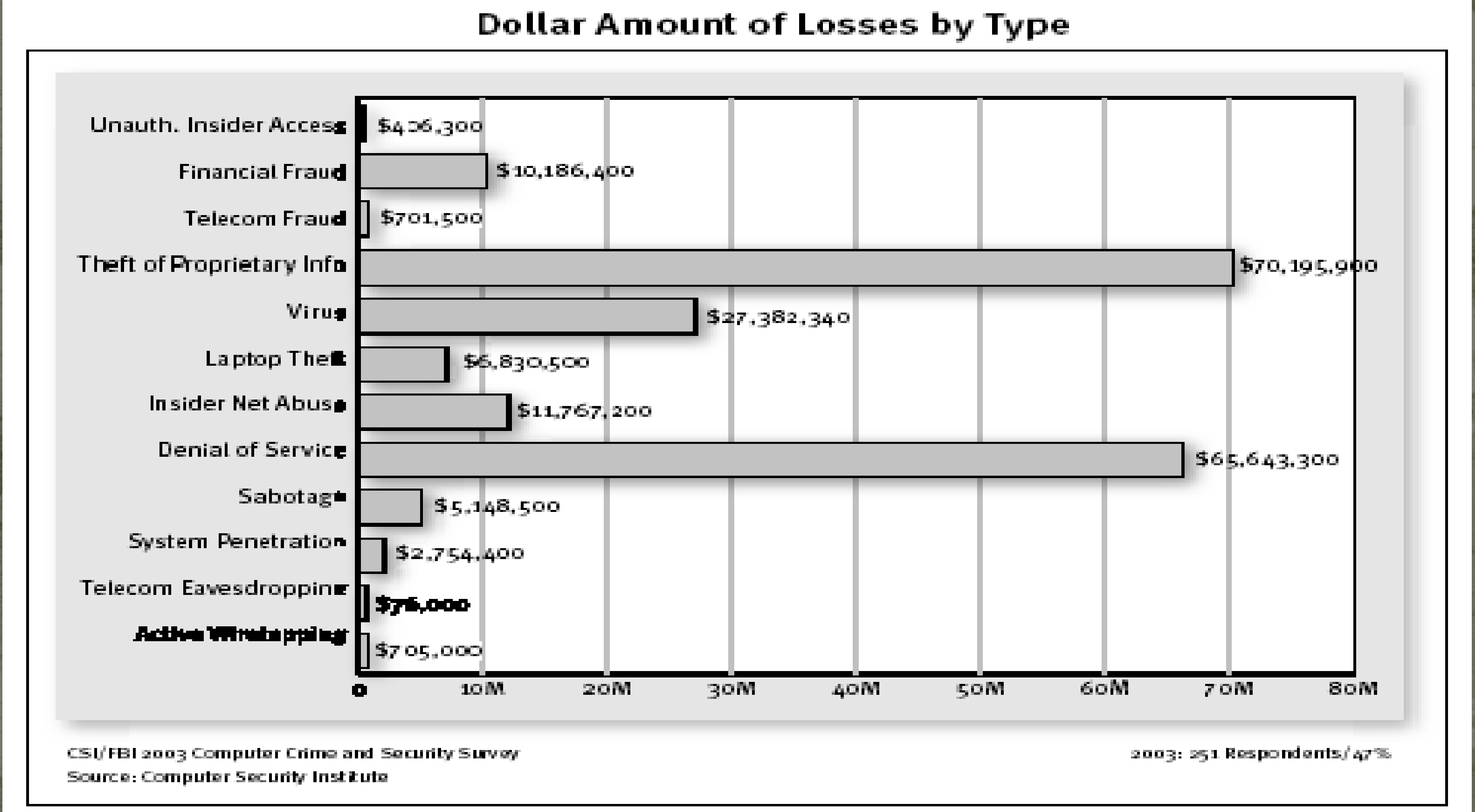
Fall 2008



TJX credit card theft

http://www.securityfocus.com/news/11455

2003 CSI/FBI Computer Security Survey



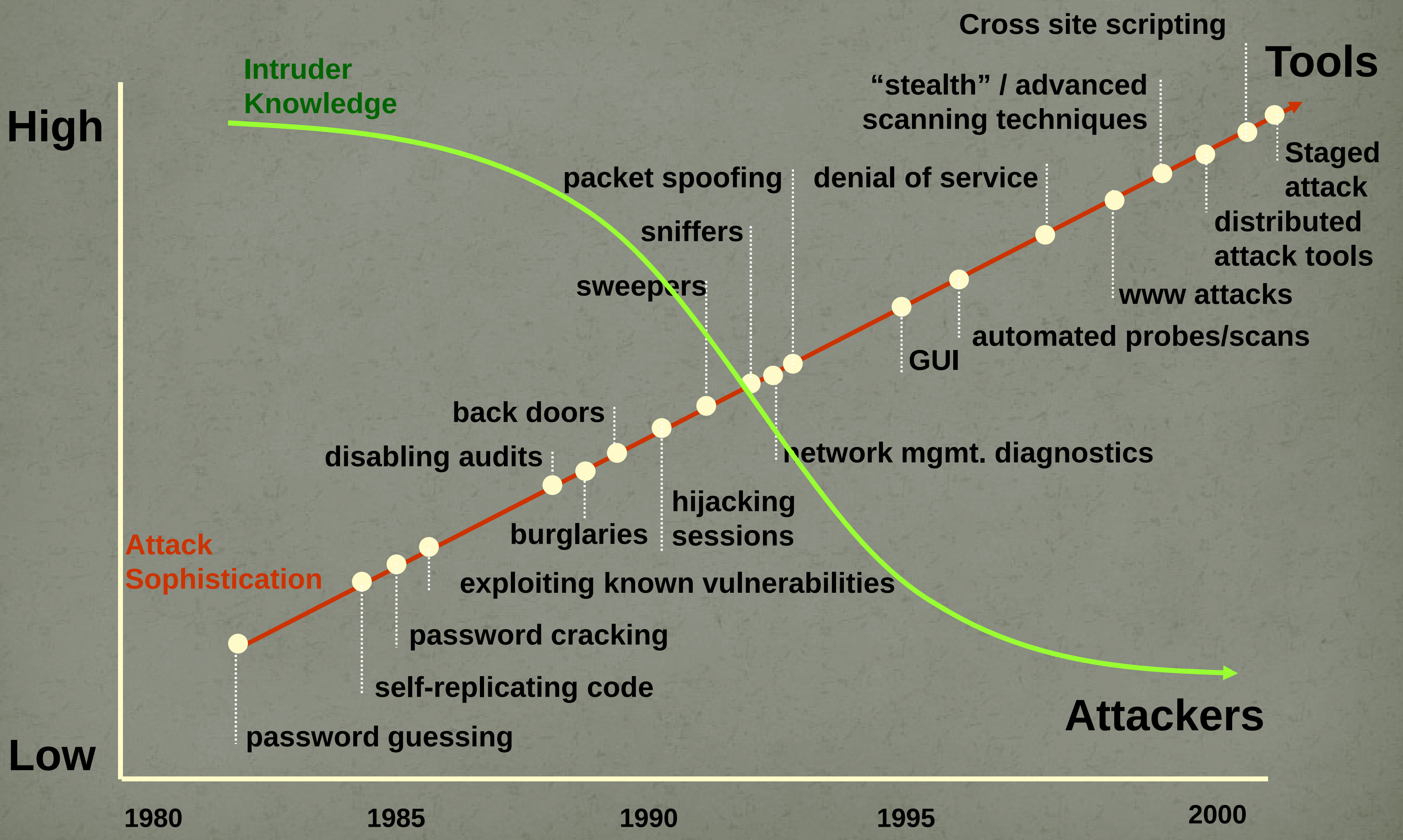
Source: http://www.gocsi.com/

Why Database Security is Important?

- Almost all corporate/organizational
  - information is stored in databases
- Use databases in daily life:
  - Web sites (Weather, news, other…)
  - Registration, Grades
  - DMV, gov't…
  - Travel…
- for many reasons, including concurrency control, centralization, management…
- Databases raise unique security challenges.

- http://www.scmagazineuk.com/Database-security-ITs-biggest-problem/article/33770/

Attack Sophistication vs. Intruder Technical Knowledge

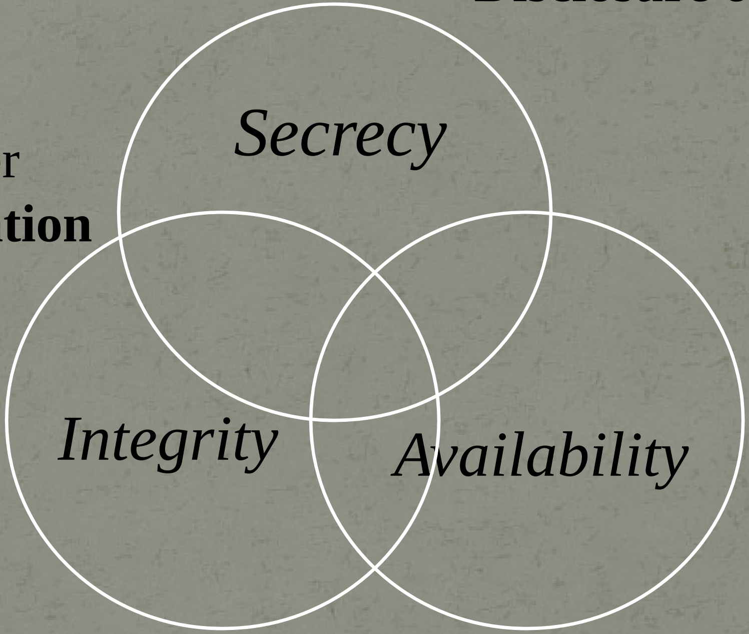


Copyright: CERT, 2000

Security Objectives

Prevent/detect/deter improper **Disclosure** of information

Prevent/detect/deter Improper **modification** of information



Prevent/detect/deter improper **Denial of access** to services

Security System Assurance

"*How well*" the security system meets the protection requirements and executes

the expected functions.

Security Controls

- *Access control* – which data users can access
- *Information flow control* – what users can do with the accessed data
- *Inference control* – how the accessed data can be used to infer additional information

Access Control

- Ensures that all *direct accesses* to object are authorized
- Protects against accidental and malicious threats by regulating the *read, write and execution* of data and programs

Access Control

Need:

- Proper *user identification*
- Information specifying the *access rights is protected* form modification

Access Control

Access control components:

- *Access control policy*: specifies the authorized accesses of a system
- *Access control mechanism*: implements and enforces the policy

Authorization

- A file system identifies certain privileges on the objects (files) it manages.
    - Typically read, write, execute.
- A file system identifies certain participants to whom privileges may be granted.
    - Typically the owner, a group, all users.

Privileges – (1)

- SQL identifies a more detailed set of privileges on objects (relations) than the typical file system.
- Nine privileges in all, some of which can be restricted to one column of one relation.

Privileges – (2)

- Some important privileges on a relation:
  1. SELECT = right to query the relation.
  2. INSERT = right to insert tuples.
     - May apply to only one attribute.
  3. DELETE = right to delete tuples.
  4. UPDATE = right to update tuples.
     - May apply to only one attribute.

Example: Privileges

- For the statement below:

```
INSERT INTO Beers(name)
    SELECT beer FROM Sells
    WHERE NOT EXISTS
            (SELECT * FROM Beers
             WHERE name = beer);
```

beers that do not appear in Beers. We add them to Beers with a NULL manufacturer.

- We require privileges SELECT on Sells and Beers, and INSERT on Beers or Beers.name.

Database Objects

- The objects on which privileges exist include stored tables and views.
- Other privileges are the right to create objects of a type, e.g., triggers.
- Views form an important tool for access control.

Example: Views as Access Control

- We might not want to give the SELECT privilege on Emps(name, addr, salary).
- But it is safer to give SELECT on:

```
CREATE VIEW SafeEmps AS
  SELECT name, addr FROM Emps;
```

- Queries on SafeEmps do not require SELECT on Emps, just on SafeEmps.

Authorization ID's

- A user is referred to by *authorization ID*, typically their login name.
- There is an authorization ID PUBLIC.
  - Granting a privilege to PUBLIC makes it available to any authorization ID.

Granting Privileges

- You have all possible privileges on the objects, such as relations, that you create.
- You may grant privileges to other users (authorization ID's), including PUBLIC.
- You may also grant privileges WITH GRANT OPTION, which lets the grantee also grant this privilege.

The GRANT Statement

- To grant privileges, say:

```
GRANT <list of privileges>
ON <relation or other object>
TO <list of authorization ID’s>;
```

- If you want the recipient(s) to be able to pass the privilege(s) to others add:

```
WITH GRANT OPTION
```

Example: GRANT

- Suppose you are the owner of Sells. You may say:

```
GRANT SELECT, UPDATE(price)
ON Sells
TO sally;
```

- Now Sally has the right to issue any query on Sells and can update the price component only.

Example: Grant Option

- Suppose we also grant:

```
GRANT UPDATE ON Sells TO sally
WITH GRANT OPTION;
```

- Now, Sally not only can update any attribute of Sells, but can grant to others the privilege UPDATE ON Sells.
  - Also, she can grant more specific privileges like `UPDATE(price)ON Sells`.

Revoking Privileges

```
REVOKE <list of privileges>
ON <relation or other object>
FROM <list of authorization ID’s>;
```

- Your grant of these privileges can no longer be used by these users to justify their use of the privilege.
  - But they may still have the privilege because they obtained it independently from elsewhere.

REVOKE Options

- We must append to the REVOKE statement either:
  1. CASCADE. Now, any grants made by a revokee are also not in force, no matter how far the privilege was passed.
  2. RESTRICT. If the privilege has been passed to others, the REVOKE fails as a warning that something else must be done to “chase the privilege down.”

Grant Diagrams

- Nodes = user/privilege/grant option?/is owner?
  - UPDATE ON R, UPDATE(a) on R, and UPDATE(b) ON R live in different nodes.
  - SELECT ON R and SELECT ON R WITH GRANT OPTION live in different nodes.
- Edge *X* ->*Y* means that node *X* was used to grant *Y*.

Notation for Nodes

- Use *AP* for the node representing authorization ID *A* having privilege *P*.
  - *P* * = privilege *P* with grant option.
  - *P* ** = the source of the privilege *P*.
    - I.e., *A* is the owner of the object on which *P* is a privilege.
    - Note ** implies grant option.

Manipulating Edges – (1)

- When *A* grants *P* to *B*, We draw an edge from *AP* * or *AP* ** to *BP*.
  - Or to *BP* * if the grant is with grant option.
- If *A* grants a subprivilege *Q* of *P* [say UPDATE(a) on R when *P* is UPDATE ON R] then the edge goes to *BQ* or *BQ* *, instead.

Manipulating Edges – (2)

- Fundamental rule: User C has privilege Q as long as there is a path from XP ** to CQ, CQ *, or CQ **, and P is a superprivilege of Q.
  - Remember that P could be Q, and X could be C.

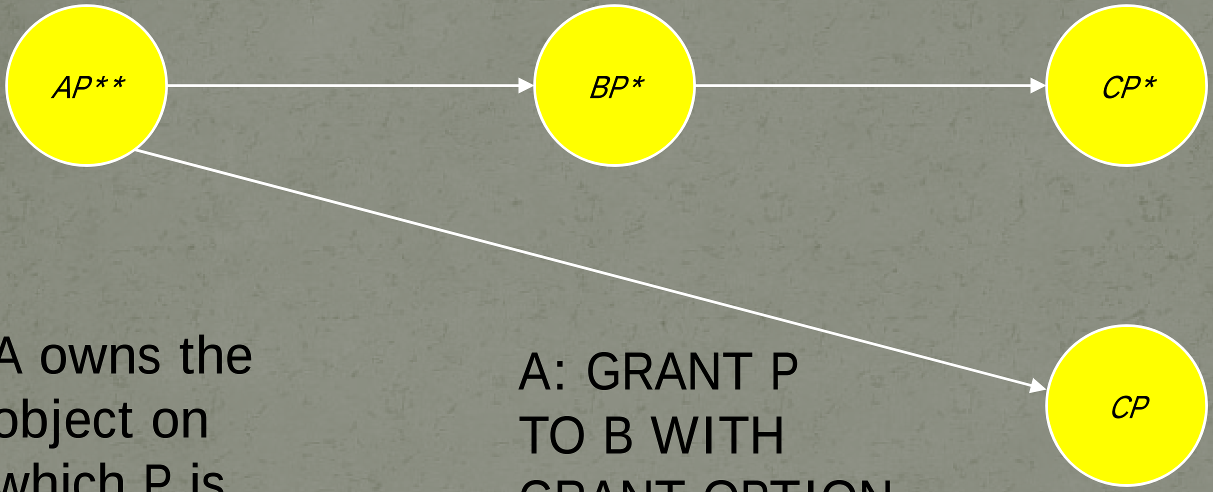
Manipulating Edges – (3)

- If *A* revokes *P* from *B* with the CASCADE option, delete the edge from *AP* to *BP*.
- But if *A* uses RESTRICT instead, and there is an edge from *BP* to anywhere, then reject the revocation and make no change to the graph.

Manipulating Edges – (4)

- Having revised the edges, we must check that each node has a path from some ** node, representing ownership.
- Any node with no such path represents a revoked privilege and is deleted from the diagram.

Example: Grant Diagram



B: GRANT P
TO C WITH
GRANT OPTION

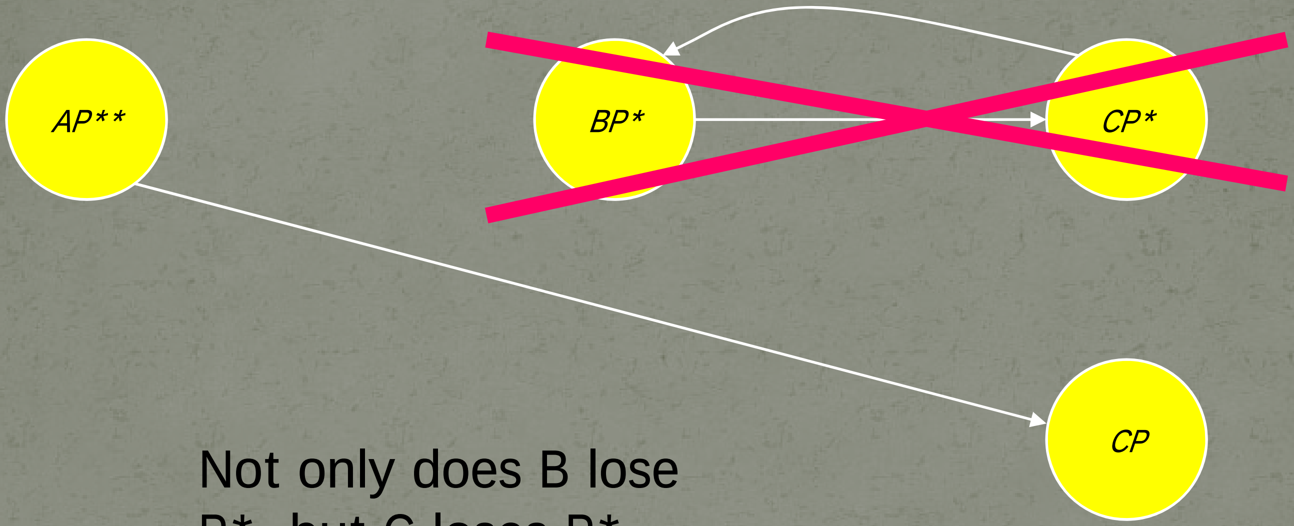
A owns the
object on
which P is
a privilege

A: GRANT P
TO B WITH
GRANT OPTION

A: GRANT P
TO C

Example: Grant Diagram

A executes
REVOKE P FROM B CASCADE;



Even had C passed P to B, both nodes are still cut off.

Not only does B lose P*, but C loses P*. Delete BP* and CP*.

However, C still has P without grant option because of the direct grant.

Access Control

- *Subject*: active entity that requests access to an object

  - e.g., user or program
- *Object*: passive entity accessed by a subject
  - e.g., record, relation, file
- *Access right* (privileges): how a subject is allowed to access an object
  - e.g., subject *s* can read object *o*

Access Control Policies

- Discretionary Access Control (DAC)
- Mandatory Access Control (MAC)
- Role-Based Access Control (RBAC)

DAC by Views

Employee relation

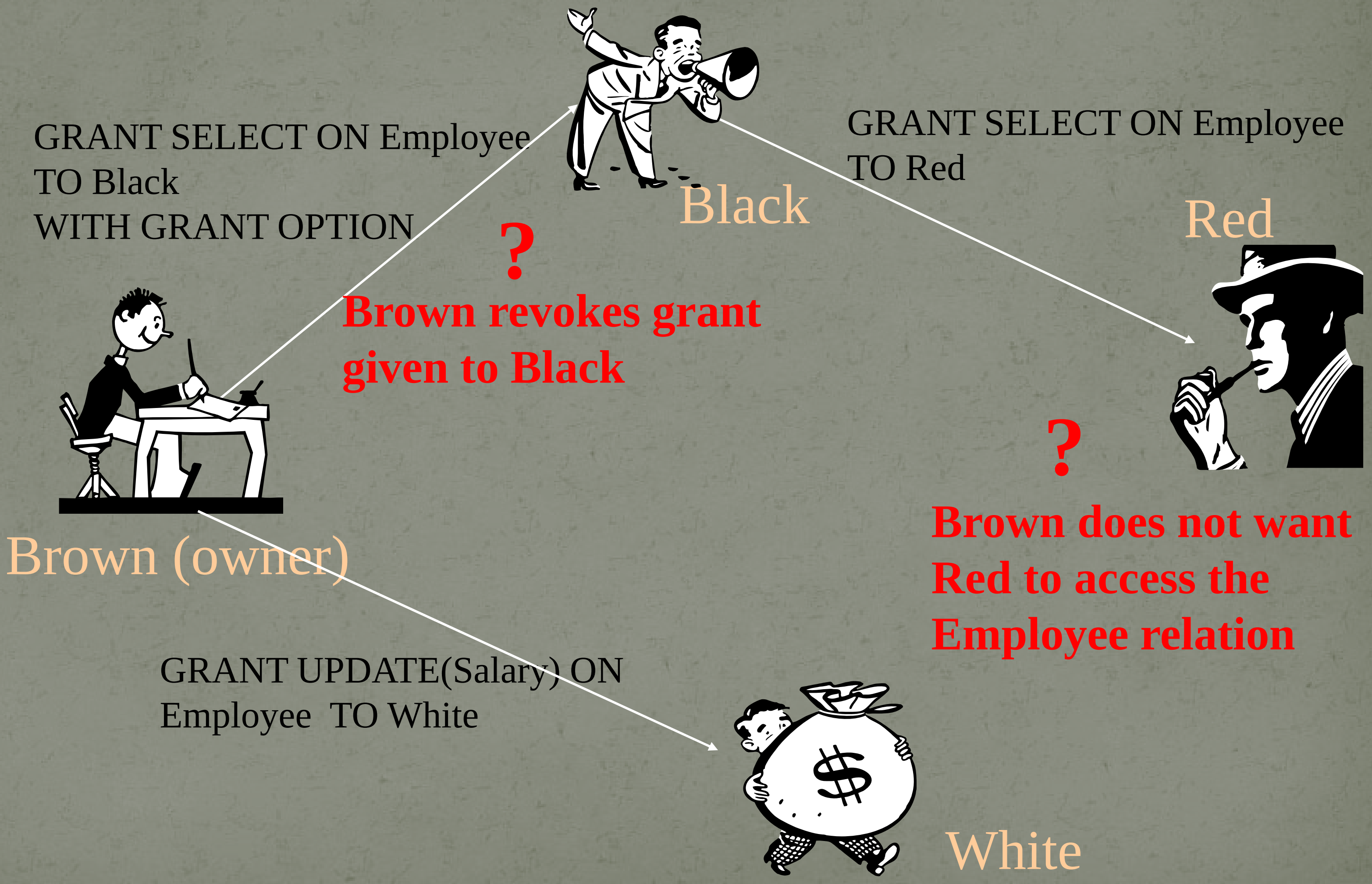
Name	*Dept.*	*Salary*	*Manager*
J. Black	Toy	25,000	S. Red
S. Red	Toy	43,000	K. Brown
L. White	Candy	28,000	G.R. Green

```
CREATE VIEW toy_dept AS
        SELECT Name,Salary, Manager
        FROM Employee
        WHERE Dept.="Toy"
```

toy_dept view

Name	*Salary*	*Manger*
J. Black	25,000	S. Red
S. Red	43,000	K. Brown

DAC by Grant and Revoke



```
GRANT SELECT ON Employee
TO Black
WITH GRANT OPTION
```

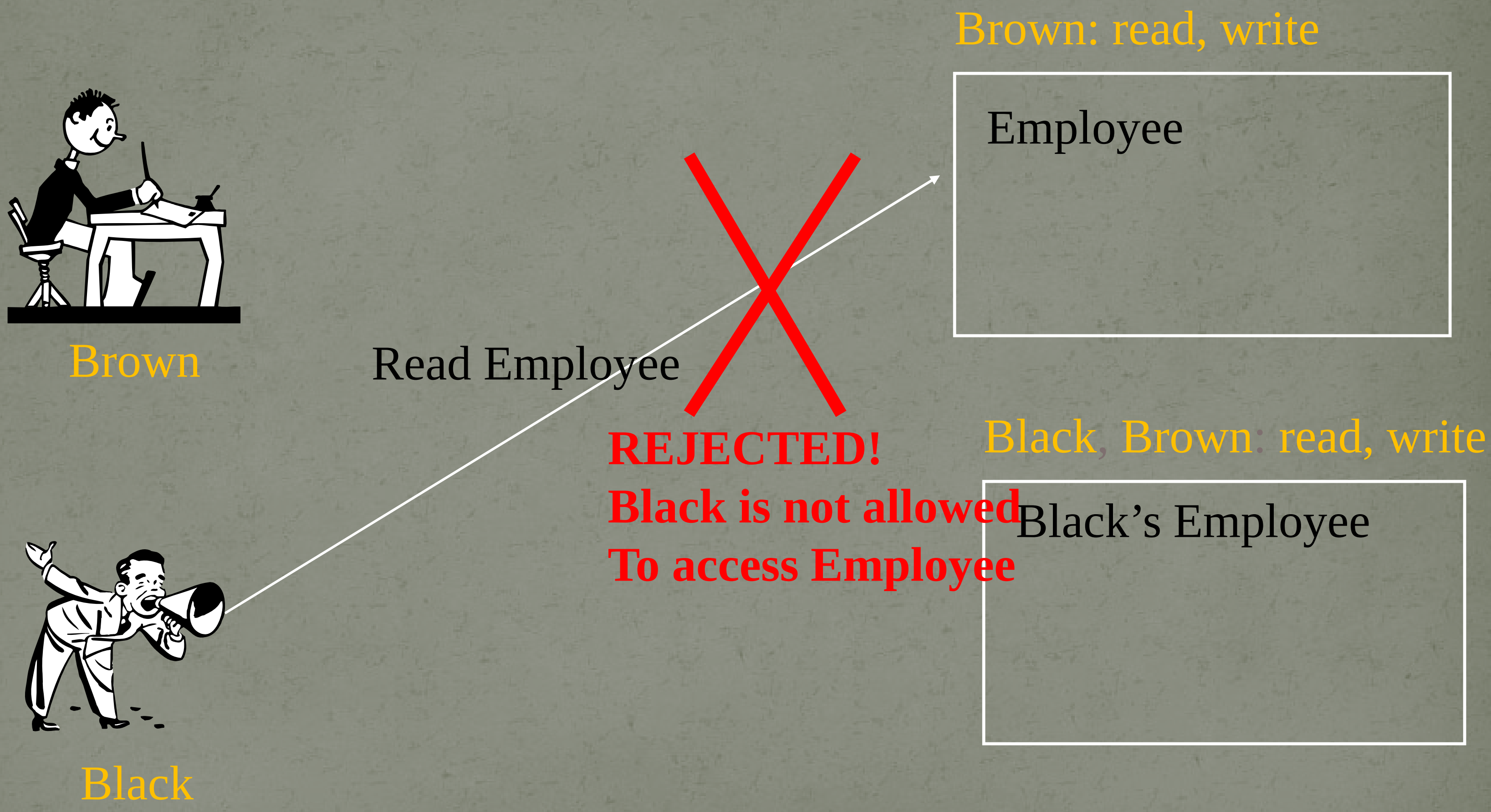
Brown revokes grant given to Black

```
GRANT SELECT ON Employee
TO Red
```

Brown does not want Red to access the Employee relation

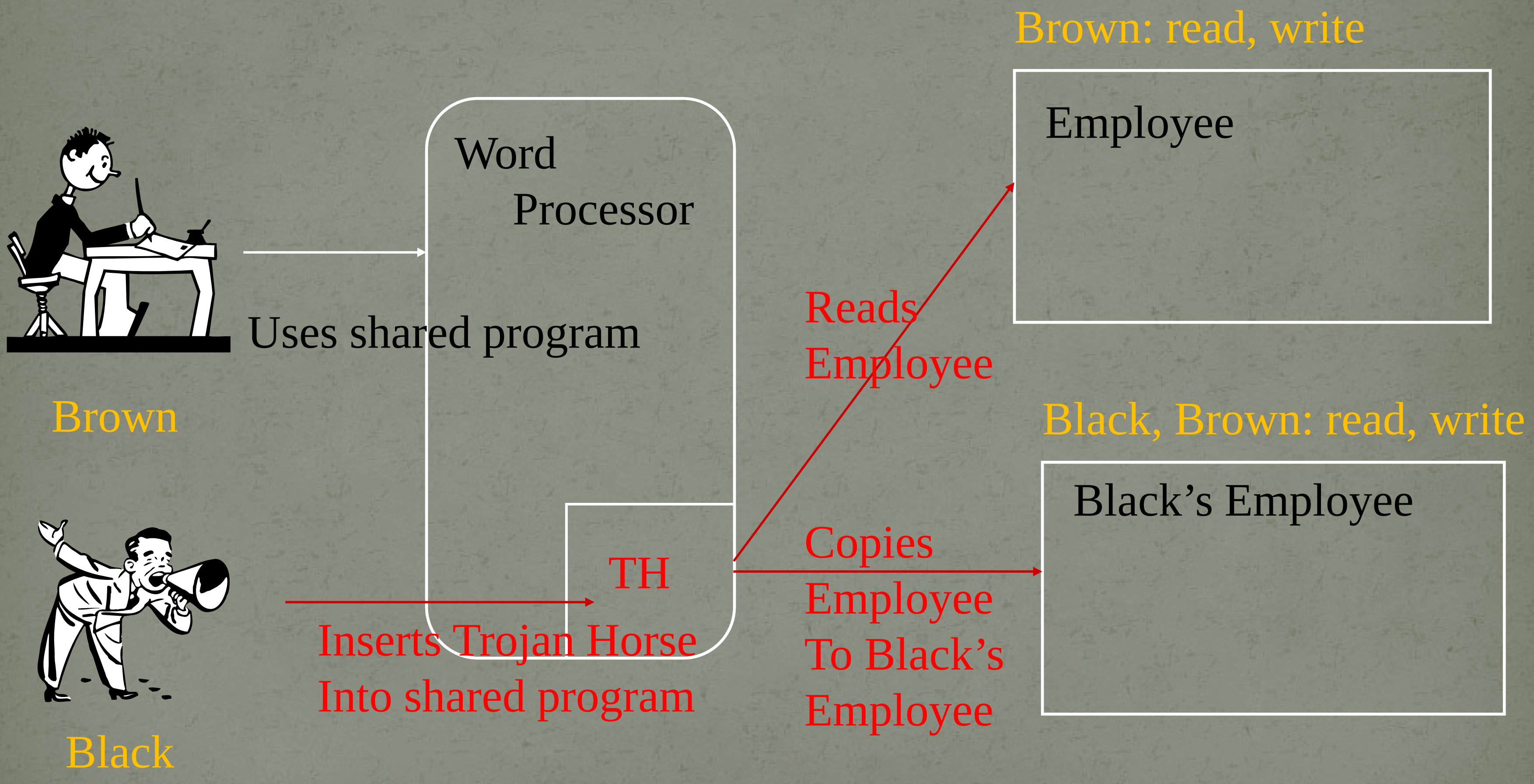
```
GRANT UPDATE(Salary) ON
Employee  TO White
```

DAC and Trojan Horse (1)



REJECTED!
Black is not allowed
To access Employee

DAC and Trojan Horse



Mandatory Access Control (MAC)

- **Security label**
  - Top-Secret, Secret, Public
- **Objects:** security classification
  - File 1 is Secret, File 2 is Public
- **Subjects:** security clearances
  - Brown is cleared to Secret, Black is cleared to Public
- **Dominance (≥)**
  - Top-Secret ≥ Secret ≥ Public

MAC

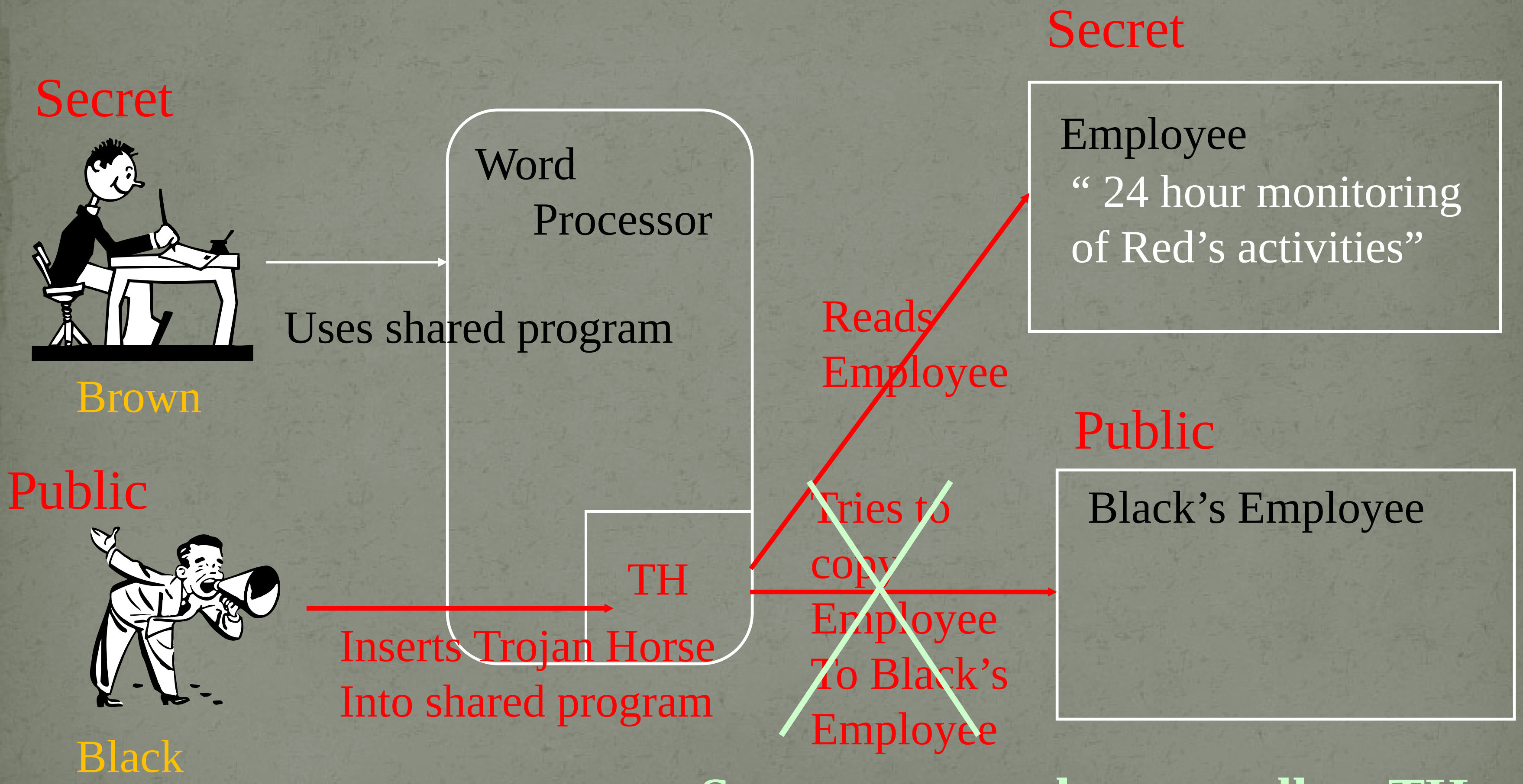
- **Access rights**: defined by comparing the security classification of the requested objects with the security clearance of the subject
- If *access control rules* are satisfied, access is permitted
- Otherwise access is rejected
- *Granularity* of access rights!

MAC – Bell-LaPadula (BLP) Model

- **Single security property**: a subject S is allowed a read access to an object O only if label(S) dominates label(O)
- ***Star-property***: a subject S is allowed a write access to an object O only if label(O) dominates label(S)

No direct flow of information from high security objects to low security objects!

BLP and Trojan Horse


Secret
Brown
Public
Black
Word Processor
Uses shared program
TH
Inserts Trojan Horse Into shared program
Reads Employee
~~Tries to copy Employee To Black's Employee~~
Secret
Employee
" 24 hour monitoring of Red's activities"
Public
Black's Employee

Star-property does not allow TH to Copy Employee to Black's Employee

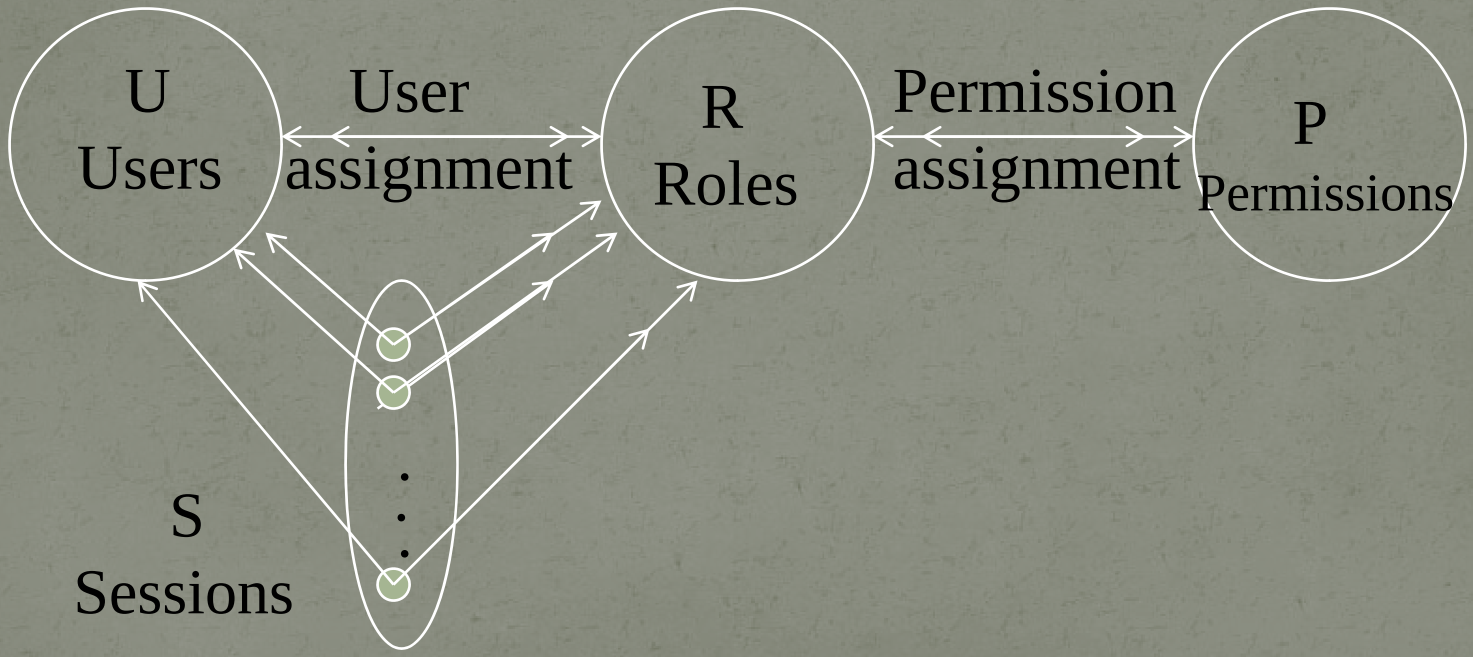
Role-Based Access Control

- Mandatory access control is rigid because the security class should be assigned to each subject and data object.
- In the real world, access privileges are associated with the role of the person in the organization. (example: bank teller)
- Each role is created and is granted/revoked privileges.
- Each user is granted/revoked roles.

Role-based Access Control

- Express organizational policies
  - Separation of duties
  - Delegation of authority
  - Role Hierchary
- Flexible: easy to modify to meet new security requirements
- Supports
  - Least-privilege
  - Separation of duties
  - Data abstraction

RBAC$_o$


U
Users
User
assignment
R
Roles
Permission
assignment
P
Permissions
S
Sessions

Suggested reading

http://csrc.nist.gov/groups/SNS/rbac/documents/design_implementation/Intro_role_based_access.htm

Flow Control

- Flow control checks that information contained in some data objects does not flow (explicitly or implicitly) into less protected objects.
- A clearance for a security class can be assigned to each application program.
- Like a DB user, each application program is subjected to the same read/write restrictions.

Covert Channels

A covert channel allows information to pass from a higher classification level to a lower classification level through improper means.

Example:

A distributed DB system has 2 sites, one with S (secret) level and the other with U (unclassified) level. During the repeated execution of a transaction, the U site agrees to commit all the time while the S site agrees to commit if the bit value is ‘1’ and does not agree to commit if the bit value is ‘0’.

Inference Control

Must prohibit the retrieval of individual data through statistical (aggregate) operations on the database.

Example:

```
SELECT  MAX(Salary)
FROM    EMPLOYEE
WHERE   Dept = 'CSE' AND
           Address LIKE '%Lexington%';
```

Note: What if only few employees live in Cincinnati?

Solutions for Inference Control

- No statistical queries are permitted whenever the number of tuples in the selected population is smaller than a certain number.
- Prohibit a sequence of queries that refer to the same population of tuples repeatedly.
- Partition the database into groups larger than certain size, and queries can refer to any complete group or set of groups, but never to a subset of a group.

Another example of inference control

Flight ID	Cargo Hold	Contents	Classification
1254	A	Boots	Unclassified
1254	B	Guns	Unclassified
1254	C	Atomic Bomb	Top Secret
1254	D	Butter	Unclassified

Flight ID	Cargo Hold	Contents	Classification
1254	A	Boots	Unclassified
1254	B	Guns	Unclassified
1254	D	Butter	Unclassified

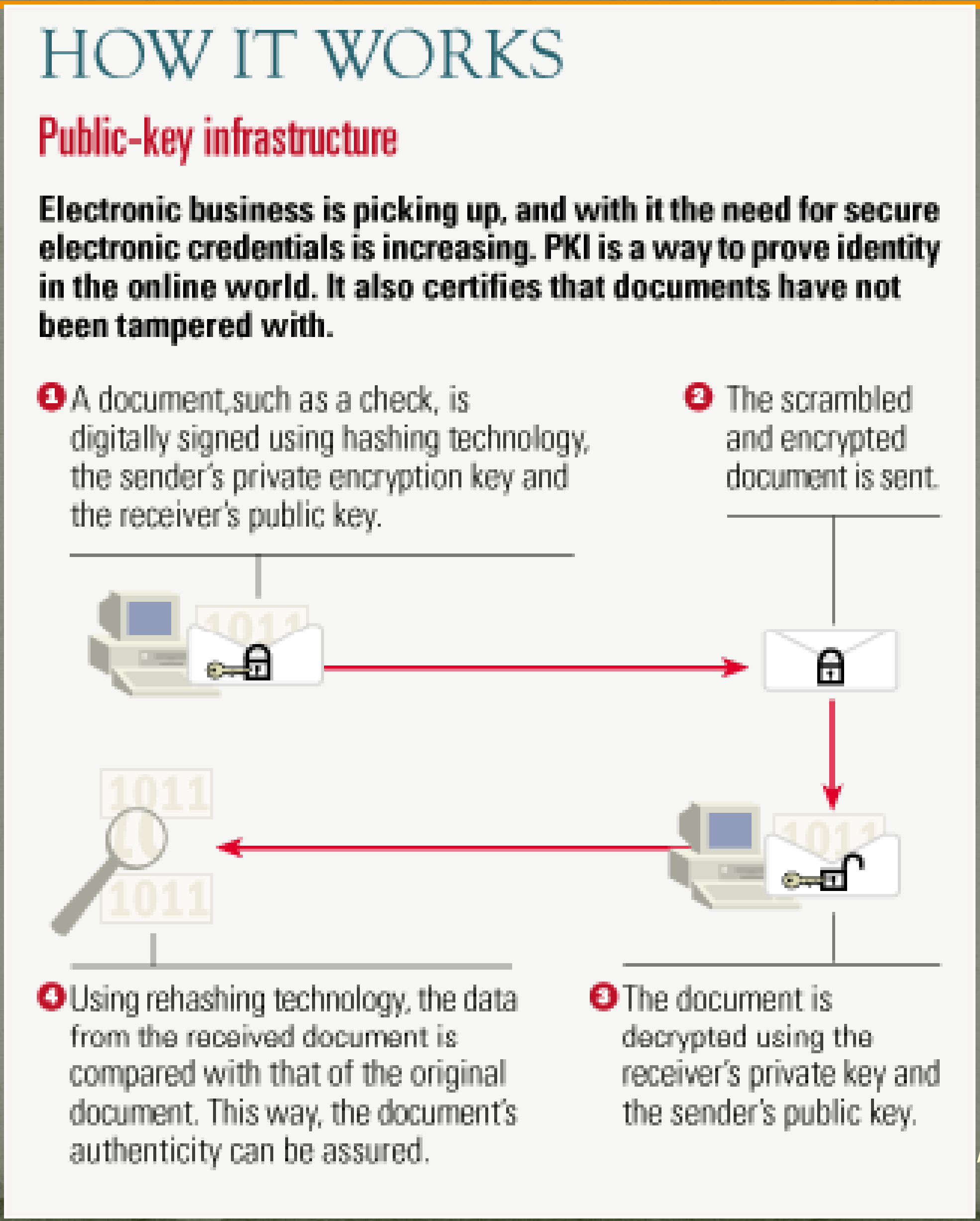
Public Key Infrastructure

- If a sender wishes to send a private message to a receiver, the sender encrypts the message using the receiver’s public key.
- When the receiver receives the message, he or she decrypts it using the receiver’s private key. No other recipient can decrypt the message because only the receiver knows his or her private key.

- What’s the problem?

Digital Signatures

- Like a handwritten signature, a digital signature is a means of associating a mark unique to a person with a body of text.
- The message sender generates the digital signature by hashing the message.
- The sender encrypts the digital signature using his/her private key first, then encrypts it using the public key of the receiver.
- The receiver decrypts the digital signature using his/her private key first, then decrypts it using the public key of the sender.
- To validate the message itself, the receiver hashes the message and compare the hash value with the decrypted digital signature.



HOW IT WORKS

Public-key infrastructure

Electronic business is picking up, and with it the need for secure electronic credentials is increasing. PKI is a way to prove identity in the online world. It also certifies that documents have not been tampered with.

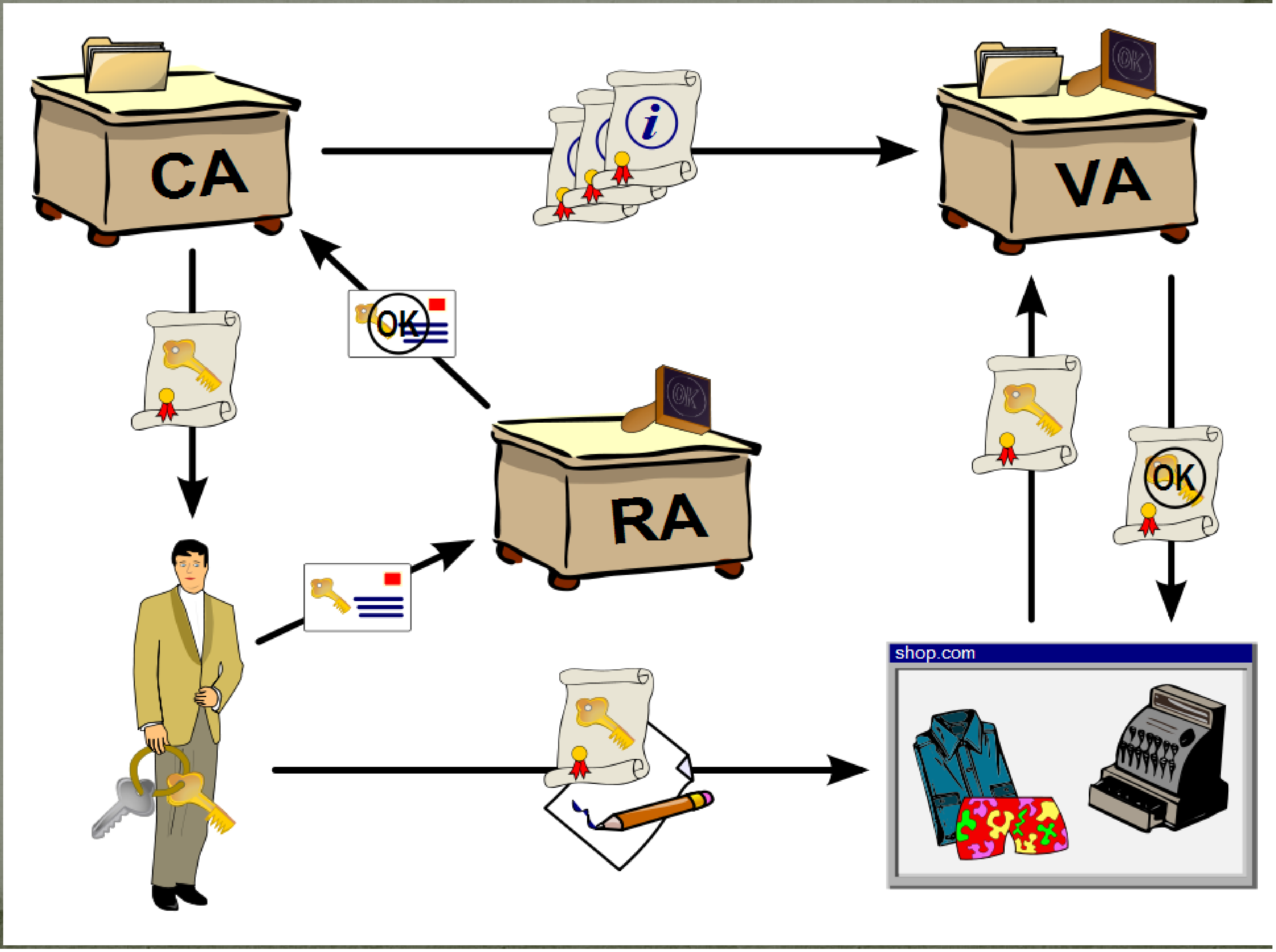
1 A document,such as a check, is digitally signed using hashing technology, the sender's private encryption key and the receiver's public key.

2 The scrambled and encrypted document is sent.

3 The document is decrypted using the receiver's private key and the sender's public key.

4 Using rehashing technology, the data from the received document is compared with that of the original document. This way, the document's authenticity can be assured.

Public Key Infrastructure



Encryption and PKI (Public Key Infrastructure)

- Each user generates a pair of keys: a *public key* and a *private key* for encryption and decryption of messages.
- Public key and private key are interchangeable: a message encrypted using one key can be decrypted by the other key.
- The public key of the pair is made public for others to use, whereas the private key is kept by the owner.
- Since the keys are generated by using exponentiation and modulo functions, it is hard to crack them.

Secure Electronic Transaction

- A buyer encrypts the non-credit card information using the public key of the seller, and encrypts the credit card information using the public key of the credit card company. Then, both are sent to the seller.
- The seller decrypts the non-credit card information using his/her private key, and forwards the credit card information (which he/she cannot decrypt) to the credit card company.
- The credit card company decrypts the card information using its private key. If the credit card company approves the card information, the transaction goes through.