Faculty of Computer Science and Engineering
Ho Chi Minh City University of Technology

Chapter 7: Database Security

Database Systems (CO2013)

Computer Science Program

Assoc. Prof. Dr. Võ Thị Ngọc Châu

(chauvtn@hcmut.edu.vn)

Semester 1 – 2022-2023

Content

- Chapter 1: An Overview of Database Systems
- Chapter 2: The Entity-Relationship Model
- Chapter 3: The Relational Data Model
- Chapter 4: The SQL Language
- Chapter 5: Relational Database Design
- Chapter 6: Physical Storage and Data Management
- **Chapter 7: Database Security**

Chapter 7: Database Security

- 7.1. An overview of database security
- 7.2. Discretionary access control based on granting and revoking privileges
- 7.3. Mandatory access control and role-based access control for multilevel security
- 7.4. Inference control and flow control
- 7.5. Encryption and public key infrastructure
- 7.6. Security in new DBMSs

Main References

Text:

- [1] R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 6th Edition, Pearson- Addison Wesley, 2011.
  - ***R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.***

References:

[1] S. Chittayasothorn, *Relational Database Systems: Language, Conceptual Modeling and Design for Engineers*, Nutcha Printing Co. Ltd, 2017.

[3] A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts – 7th Edition, McGraw-Hill, 2020.

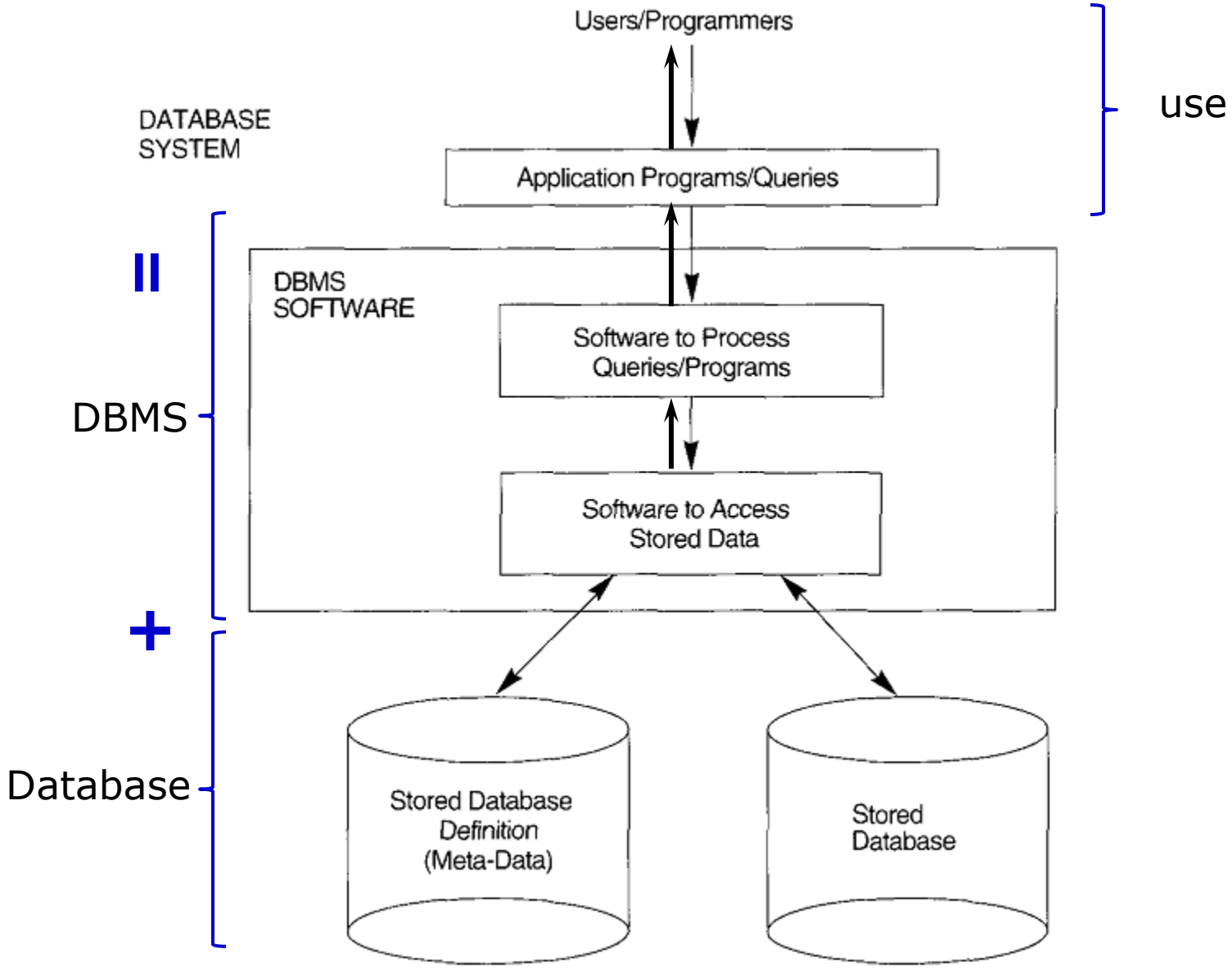
[4] H. G. Molina, J. D. Ullman, J. Widom, *Database Systems: The Complete Book - 2nd Edition,* Prentice-Hall, 2009.

[5] R. Ramakrishnan, J. Gehrke, *Database Management Systems – 4th Edition,* McGraw-Hill, 2018.

[6] M. P. Papazoglou, S. Spaccapietra, Z. Tari, *Advances in Object-Oriented Data Modeling*, MIT Press, 2000.

[7]. G. Simsion, *Data Modeling: Theory and Practice*, Technics Publications, LLC, 2007.

A simplified database system environment



An overview of database security

- Security
  - Protection of a *person, building, organization or country* against threats such as crime or attacks by *foreign countries* [Dictionary]
    - Protection
    - Person, building, organization, country
    - Threats (crime/attacks)/risks
    - Foreign countries

An overview of database security

- Computer security

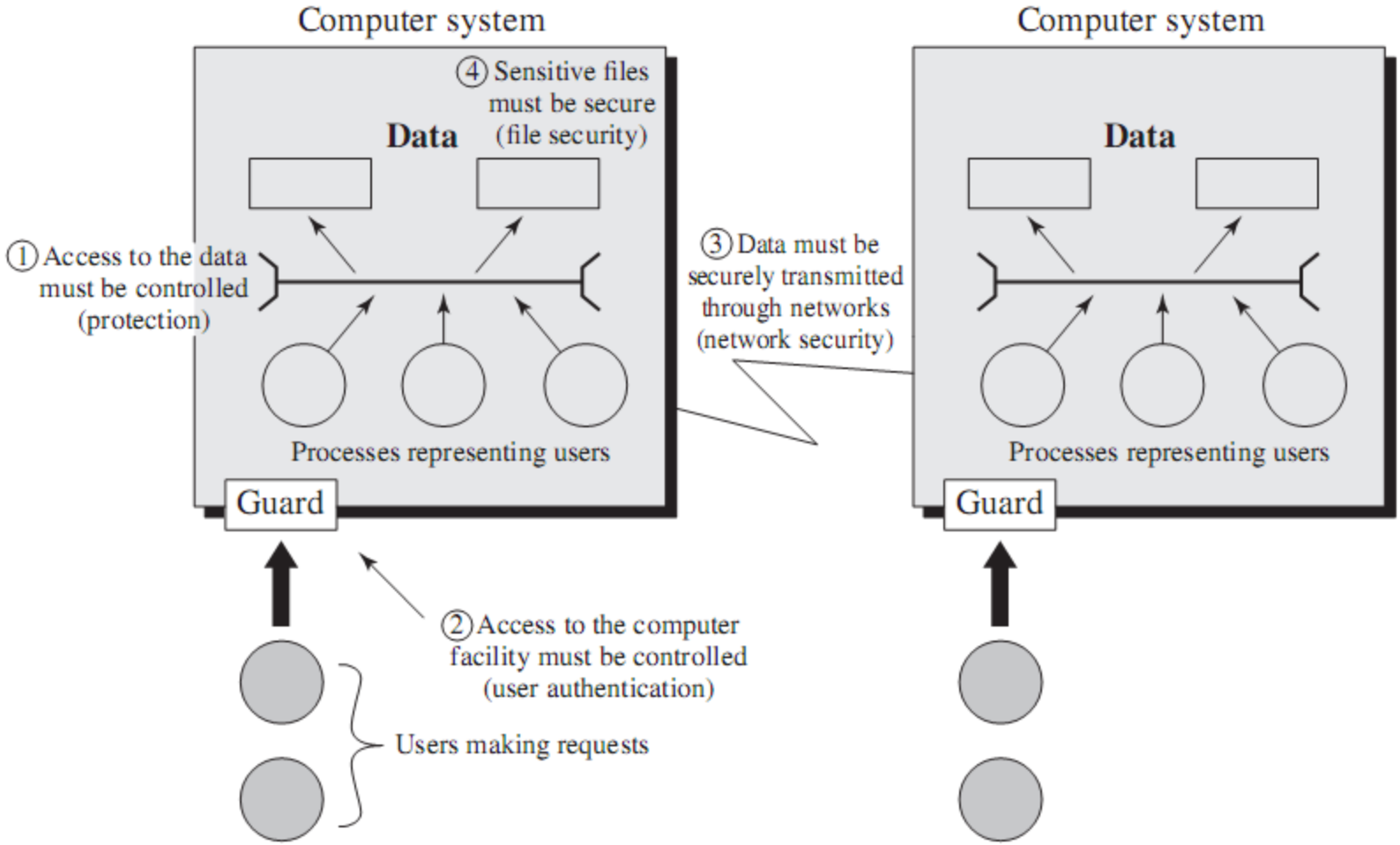
  (*Bảo mật hệ thống máy tính/ thông tin*)

  - The *protection* afforded to an automated information system in order to attain the applicable *objectives* of preserving the *integrity*, *availability*, and *confidentiality* of information system *resources* (includes hardware, software, firmware, information/data, and telecommunications).
    - The NIST Computer Security Handbook, 1995
      - NIST: National Institute of Standards and Technology (US)

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

An overview of database security

- Computer security



Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

An overview of database security

- Threats to computer systems

	Availability	Confidentiality	Integrity
Hardware	Equipment is stolen or disabled, thus denying service.	An unencrypted CD-ROM or DVD is stolen.	
Software	Programs are deleted, denying access to users.	An unauthorized copy of software is made.	A working program is modified, either to cause it to fail during execution or to cause it to do some unintended task.
Data	Files are deleted, denying access to users.	An unauthorized read of data is performed. An analysis of statistical data reveals underlying data.	Existing files are modified or new files are fabricated.
Communication Lines and Networks	Messages are destroyed or deleted. Communication lines or networks are rendered unavailable.	Messages are read. The traffic pattern of messages is observed.	Messages are modified, delayed, reordered, or duplicated. False messages are fabricated.

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

An overview of database security

- Database security (*Bảo mật cơ sở dữ liệu*)
  - Database security refers to (*database*) *protection* from *malicious access*. Absolute protection of the database from malicious abuse is not possible, but the cost to the perpetrator (*tội phạm*) can be made high enough to deter (*ngăn chặn*) most if not all attempts to access the database without *proper authority* (*quyền hợp pháp*).

Source: A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts – 6th Edition, McGraw-Hill, 2006.

An overview of database security

- Database security (Bảo mật cơ sở dữ liệu)
- To protect the database, *security measures* must be taken at several levels:
  - *Database system*. Some database-system users may be authorized to access only a limited portion of the database. Other users may be allowed to issue queries, but may be forbidden to modify the data. It is the responsibility of the database system to ensure that these authorization restrictions are not violated.
  - *Operating system*. No matter how secure the database system is, weakness in operating system security may serve as a means of unauthorized access to the database.
  - *Network*. Since almost all database systems allow *remote access* through terminals or networks, software-level security within the network software is as important as physical security, both on the Internet and in private networks.
  - *Physical*. Sites with computer systems must be physically secured against armed or surreptitious (*bí mật*) entry by intruders.
  - *Human*. Users must be authorized carefully to reduce the chance of any user giving access to an intruder in exchange for a bribe (*hối lộ*) or other favors.

Source: A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts – 6th Edition, McGraw-Hill, 2006.

An overview of database security

- Threats (*Mối nguy hiểm*) to databases
  - *Loss of confidentiality* (*Mất tính bí mật*). Database confidentiality refers to the protection of data from unauthorized disclosure (*sự phơi bày trái phép*). The impact of unauthorized disclosure of confidential information can range from violation of the Data Privacy Act to the jeopardization (*sự nguy hại*) of national security. Unauthorized, unanticipated, or unintentional disclosure could result in loss of public confidence, embarrassment, or legal action against the organization.
  - *Loss of integrity* (*Mất toàn vẹn dữ liệu*). Database integrity refers to the requirement that information be protected from improper modification. Modification of data includes creating, inserting, and updating data; changing the status of data; and deleting data. Integrity is lost if unauthorized changes (*thay đổi trái phép*) are made to the data by either intentional or accidental acts. If the loss of system or data integrity is not corrected, continued use of the contaminated (*bị hư hỏng*) system or corrupted data could result in *inaccuracy*, *fraud*, or *erroneous decisions*.
  - *Loss of availability* (*Mất sự sẵn dùng*). Database availability refers to making objects available to a human user or a program who/which has a legitimate right to those data objects. Loss of availability occurs when the authorized user or program *cannot access* these objects.

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- Threats to databases
  - CIA triad to define database security objectives
  - Confidentiality (*not disclosed to unauthorized ones*)
    - Personal medical records of each patient about their disease and treatment must be kept confidential, only known by each patient and his/ her medication team (doctors/ nurses/ …).
  - Integrity (*timely, accurate, complete, consistent*)
    - The drug use of a patient is not allowed to be changed to something else by other people once given by an in-charge doctor.
  - Availability (*not denied to authorized users*)
    - Upon a visit, a doctor can have access to the medical record of his/ her patient.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures (*biện pháp đối phó*) can be implemented:
  - Access control (*điều khiển truy cập*)
  - Inference control (*điều khiển suy diễn*)
  - Flow control (*điều khiển dòng*)
  - Encryption (*mã hóa*)

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures (*biện pháp đối phó*) can be implemented:
  - Access control (*điều khiển truy cập*)
    - A security mechanism to *control access to data* in a database system
      - Preventing *unauthorized* persons from accessing the system, either to obtain information or to make malicious changes in the database

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures:
  - Access control (*điều khiển truy cập*)
    - Discretionary access control (DAC)
      - *Điều khiển truy cập tùy quyền*
    - Mandatory access control (MAC)
      - *Điều khiển truy cập bắt buộc*
    - Role-based access control (RBAC)
      - *Điều khiển truy cập dựa vào vai trò*
    - Attribute-based access control (ABAC)
      - *Điều khiển truy cập dựa vào thuộc tính*
    - Code-based access control (CBAC)
      - *Điều khiển truy cập dựa vào mã*

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures (*biện pháp đối phó*) can be implemented:
  - Inference control (*điều khiển suy diễn*)
    - A security mechanism to control inferences from statistical databases that *infer certain facts concerning individuals* from queries involving only summary statistics on groups

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures (*biện pháp đối phó*) can be implemented:
  - Flow control (*điều khiển dòng*)
    - A security mechanism to prevent information from flowing in such a way that it reaches unauthorized users

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- To protect databases against these threats, four kinds of countermeasures (*biện pháp đối phó*) can be implemented:
  - Encryption (*mã hóa*)
    - A security mechanism to protect *sensitive data* that is transmitted via some type of communications network by means of *encrypting* techniques
      - The sensitive data, e.g. credit card number, is *encoded* using some coding algorithm. An unauthorized user who accesses encoded data will have difficulty deciphering it, but authorized users are given decoding or decrypting algorithms (or keys) to decipher the data.

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- Database Security and Database Administrator (DBA, *quản trị viên cơ sở dữ liệu*)
  - DBA is the central authority for managing a database system.
  - The responsibility of DBA:
    - Administering the resources
      - Primary resource: the ***database*** ⇒ perform DDL statements (CREATE/ ALTER/ DROP data objects (schema, table, index, trigger, constraint, …) and CREATE/ ALTER/ DROP USER/ ROLE statements, …
      - Secondary resources: database management system (DBMS) and other related softwares
    - Authorizing access to the database ⇒ perform GRANT/ REVOKE
    - Coordinating and monitoring the use of the database
    - Acquiring software and hardware resources as needed
    - Being accountable for problems such as security breaches and poor system response time

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- Database Security and Database Administrator (DBA, *quản trị viên cơ sở dữ liệu*)
  - For database security, DBA has a DBA account in the DBMS, sometimes called a system or superuser account, which provides powerful capabilities that are not made available to regular database accounts and users.
    - root in MySQL
    - sa in MS SQL Server
    - SYS, SYSTEM, DBSNMP in Oracle
      - SYS: An account used to perform database administration tasks
      - SYSTEM: A default generic database administrator account
      - DBSNMP: An account used by the Management Agent component of Oracle Enterprise Manager to monitor and manage the database in cloud control

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- Database Security and Database Administrator (DBA, *quản trị viên cơ sở dữ liệu*)
  - For database security, DBA performs the actions
    - *Account creation* (*tạo tài khoản*). This action creates a new account and password for a user or a group of users to enable access to the DBMS.
    - *Privilege granting* (*cấp quyền*). This action permits the DBA to grant certain privileges to certain accounts.
    - *Privilege revocation* (*thu hồi quyền*) This action permits the DBA to revoke (cancel) certain privileges that were previously given to certain accounts.
    - *Security level assignment* (*gán mức bảo mật*). This action consists of assigning user accounts to the appropriate security clearance level.

Source: R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.

An overview of database security

- Confidentiality (*tính bí mật*)
- Integrity (*tính toàn vẹn*)
- Availability (*sự sẵn dùng*)
- Identification (*định danh*)
- Authentication (*xác thực*)
- Authorization (*định quyền*)
- Auditing (*kiểm toán, kiểm tra sổ ghi nhật ký các truy cập và tác vụ trên cơ sở dữ liệu trong thời gian nhất định*)
- Audit trail (*audit log, sổ ghi nhật ký được dùng cho mục đích bảo mật*)
- Accountability (*trách nhiệm giải trình*)
- Non-repudiation (*chống thoái thác*)
- Privacy (*right of retaining control over personal data, what information is shared with whom, sự riêng tư*)
- Security policy (*chính sách bảo mật*)
- Recovery (*phục hồi*)
- Risk (*rủi ro*)
- Threat (*mối nguy hiểm*)
- Vulnerability (*lỗ hổng*)
- Safeguard (*ban bảo vệ*)
- Asset (*tài sản*)
- Sensitive data (*dữ liệu quan trọng, có tính toàn vẹn/ sẵn dùng phải được đảm bảo*)
- Malicious code (*mã độc hại*)
  - Virus: a code segment that replicates by attaching copies of itself to existing executables
  - Trojan horse: a program that performs a desired task but also includes unexpected functions
  - Worm: a self-replicating program
- Intruder (*kẻ xâm nhập*)
- Intrusion detection (*phát hiện xâm nhập*)

An overview of database security

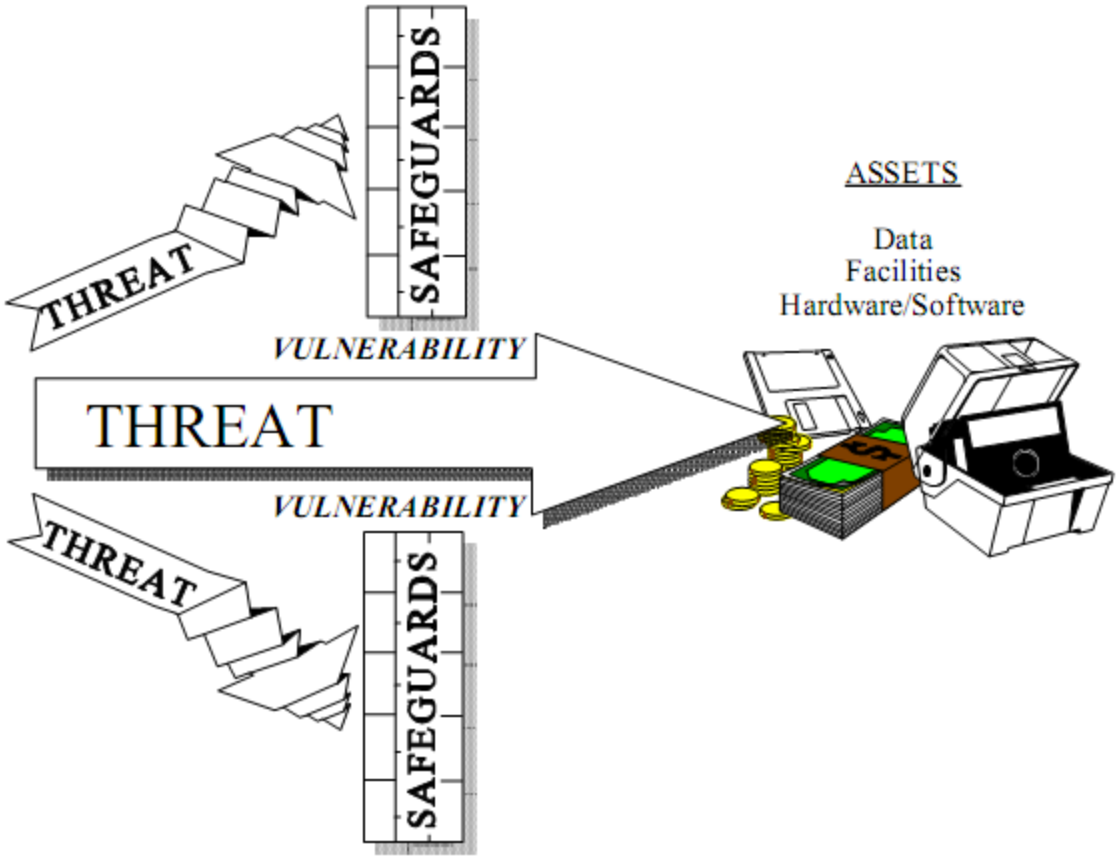
- Information *security policy* is an aggregate of directives, rules, and practices that prescribes how an organization manages, protects, and distributes information.
  - Source: P. Bowen, J. Hash, M. Wilson. Information Security Handbook: A Guide for Managers, NIST Special Publication 800-10, 2006.
- *Identification* and *Authentication* is a technical method that prevents unauthorized people (or unauthorized processes) from entering a system.
  - *Identification* is the means by which a user provides a claimed identity to the system.
  - *Authentication* is the means of establishing the validity of this identity claim.
  - Source: An Introduction to Computer Security: The NIST Handbook, Special Publication 800-12.

An overview of database security

- There are three means of *authenticating a user's identity* which can be used alone or in combination:
  - *something the individual **knows*** (a secret e.g., a password, Personal Identification Number (PIN), or cryptographic key);
  - *something the individual **possesses*** (a token e.g., an ATM card or a smart card);
  - *something the individual **is*** (a biometric e.g., such characteristics as a voice pattern, handwriting dynamics, or a fingerprint).

Source: An Introduction to Computer Security: The NIST Handbook, Special Publication 800-12.

An overview of database security

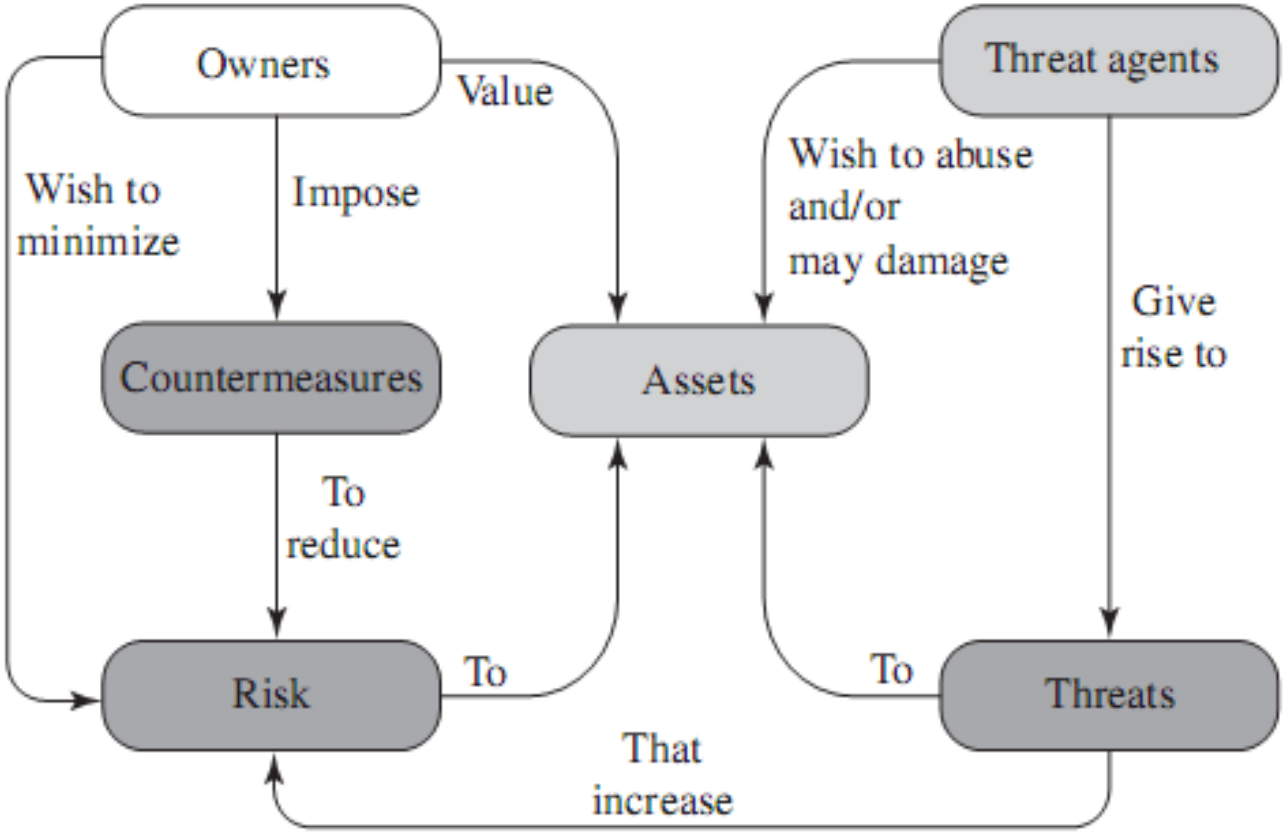


Safeguards prevent threats from harming assets. However, if an appropriate safeguard is not present, a vulnerability exists which can be exploited by a threat, thereby puttting assets at risk.

Source: Figure 7.1 in An Introduction to Computer Security: The NIST Handbook, Special Publication 800-12.

An overview of database security

- Put them altogether for Security



Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

An overview of database security

- A methodical approach to conducting a technical attack
  - 1. Discovering the key elements of the system
  - 2. Scanning for vulnerabilities
  - 4. Hacking the system to gain root/administrator privileges
  - 5. Disabling auditing and removing traces from log files
  - 6. Stealing files, modifying data, and stealing source code or other valuable information
  - 7. Installing back doors and Trojan horses that permit undetectable reentry
  - 8. Returning at will to inflict more damage

An overview of database security

- Legal issues
  - Legal: obeying the law
  - Issues: privacy, intellectual property, free speech, computer crimes, seller/buyer protection, gambling, electronic contracts, taxation, …
- Ethical issues
  - Ethical: obeying the right actions (morals/responsibility) → subjective and circumstance-dependent
  - Issues: privacy, accuracy, property, accessibility

→ Relationship between legal/ethical issues

Discretionary access control based on granting and revoking privileges

- *Discretionary access control* (DAC) is one security scheme in which an entity may be granted access rights that permit the entity, by its own volition, to enable another entity to access some resource.
- A general approach to DAC, as exercised by an operating system or a database management system (DBMS), is that of an *access matrix*.
  - One dimension of the matrix consists of *identified subjects* (*users*) that may attempt data access to the resources.
  - The other dimension lists the *objects* that may be accessed.
    - At the greatest level of detail, objects may be individual data fields. More aggregate groupings, such as records, files, or even the entire database, may also be objects in the matrix.
  - Each entry in the matrix indicates the *access rights* of a particular subject for a particular object.

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

Discretionary access control based on granting and revoking privileges

- An *access matrix* on the COMPANY database with three users: Normal Employee, Manager, Partner

	Normal Employee	Manager	Partner
employee	SELECT, UPDATE	SELECT	
dependent	SELECT, INSERT, DELETE, UPDATE	SELECT	
department	SELECT	SELECT, UPDATE	SELECT
dept_location	SELECT	SELECT, UPDATE	SELECT
project	SELECT	SELECT, INSERT, DELETE, UPDATE	SELECT
works_on	SELECT, UPDATE	SELECT, INSERT, DELETE, UPDATE	

Discretionary access control based on granting and revoking privileges

- In practice, an access matrix is usually sparse and is implemented by decomposition.
  - Decomposition by data objects: *access control lists* (ACLs)
    - This data structure is not convenient for determining the access rights available to a specific user.
  - Decomposition by subjects: *capability tickets*
    - A capability ticket specifies authorized objects and operations for a particular user. Each user has a number of tickets and may be authorized to loan or give them to others.
    - Because tickets may be dispersed around the system, they present a greater security problem than access control lists.

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

Discretionary access control based on granting and revoking privileges

- Many current relational DBMSs use some variation of this technique.
- The main idea for database access control is to include statements in the query language (e.g. SQL) that allow the DBA and selected users to grant and revoke privileges.
  - Statement: GRANT, REVOKE
  - User: an account through which you can log in to the database, and to establish the means by which *DBMS* permits access by the user
  - Privilege: the right to perform some action

Discretionary access control based on granting and revoking privileges

- There are two levels for assigning privileges to use the database system (*DBMS-specific implementation*):
  - The *account* level. At this level, the DBA specifies the particular privileges that each account holds independently of the relations in the database. Some privileges are CREATE SCHEMA, CREATE TABLE, CREATE VIEW, ALTER …, DROP …, …
  - The *relation* (or *table*) level. At this level, the DBA can control the privilege to access each individual relation or view in the database. Some privileges are SELECT, INSERT, DELETE, UPDATE, ...
    - *References* privilege on a table R. This gives the account the capability to reference (or refer to) table R when specifying integrity constraints. This privilege can also be restricted to specific attributes of table R.

Discretionary access control based on granting and revoking privileges

- Levels for assigning privileges to use the database system (*DBMS-specific implementation*):
  - Oracle 19c: *System* privileges and *Object* privileges
  - MS SQL Server: *Server* permissions and *Database* permissions
  - MySQL: *Administrative* privileges, *Database* privileges that apply to a database and to all objects within it, privileges for *database objects* such as tables, indexes, views, and stored routines

Discretionary access control based on granting and revoking privileges

- To control the granting and revoking of relation privileges, each relation R in a database is assigned an *owner account*, which is typically the account that was used when the relation was created in the first place.
- The *owner* of a relation is given *all privileges* on that relation.
- The *owner account holder can pass privileges* on any of the owned relations to other users by granting privileges to their accounts.

DBMS-specific implementation. In MS SQL Server? NOT in MySQL?

Discretionary access control based on granting and revoking privileges

- Specifying Privileges through the Use of Views
  - The mechanism of *views* is an *important* discretionary authorization mechanism.
  - Example 1: If the owner A of a relation R wants another account B to be able to *retrieve only some fields* of R, then A can create a view V of R that includes only those attributes and then grant SELECT on V to B.
  - Example 2: The same applies to limiting B to *retrieving only certain tuples* of R; a view V' can be created by defining the view by means of a query that selects only those tuples from R that A wants to allow B to access.

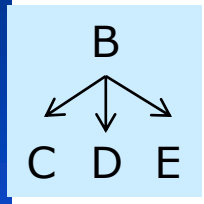
Discretionary access control based on granting and revoking privileges

- Propagation of Privileges Using the GRANT OPTION
  - Whenever the owner A of a relation R grants a privilege on R to another account B, the privilege can be given to B with or without the GRANT OPTION.
  - If the GRANT OPTION is given, this means that B can also grant that privilege on R to other accounts.
  - Suppose that B is given the GRANT OPTION by A and that B then grants the privilege on R to a third account C, also with the GRANT OPTION. In this way, privileges on R can propagate to other accounts without the knowledge of the owner of R. If the owner account A now revokes the privilege granted to B, all the privileges that B propagated based on that privilege should automatically be revoked by the system.
  - A DBMS that allows propagation of privileges must keep track of how all the privileges were granted in the form of some internal log so that revoking of privileges can be done correctly and completely.

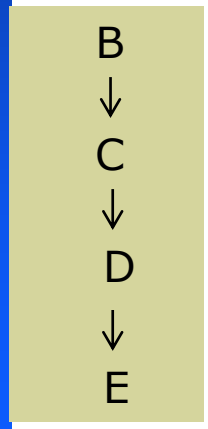
DBMS-specific implementation. In MS SQL Server? In MySQL?

Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Techniques to limit the propagation of privileges have been developed, although they have *not yet been implemented in most DBMSs* and are *not a part of SQL*.



- Limiting horizontal propagation to an integer number *i* means that an account B given the GRANT OPTION can grant the privilege to at most *i* other accounts.



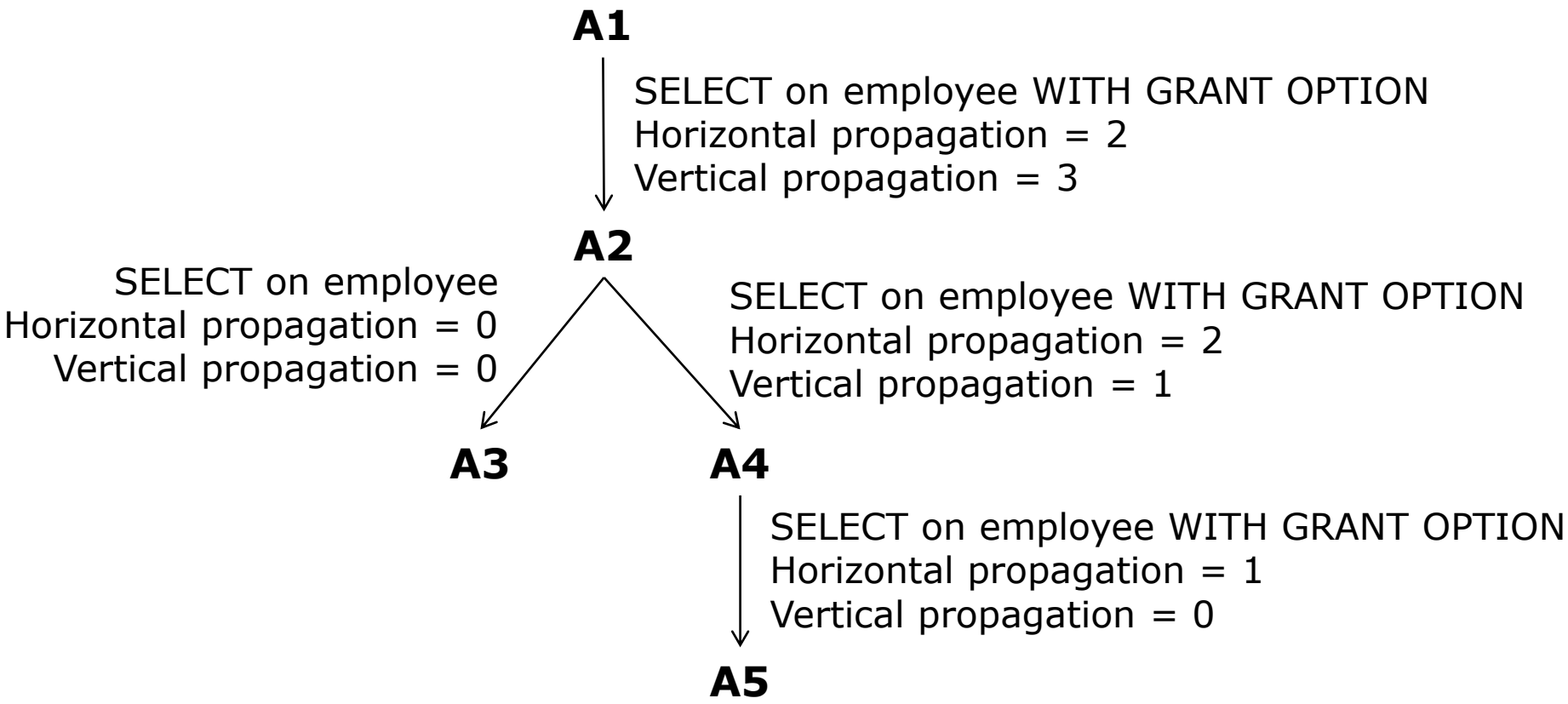
- Vertical propagation is more complicated; it limits the depth of the granting of privileges. Granting a privilege with a vertical propagation of zero is equivalent to granting the privilege with no GRANT OPTION.
- Horizontal and vertical propagations are designed to limit the depth and breadth of propagation of privileges.

Discretionary access control based on granting and revoking privileges

- A1 grants SELECT to A2 on the EMPLOYEE relation with horizontal propagation equal to 2 and vertical propagation equal to 3. A2 can then grant SELECT to at most two accounts because the horizontal propagation limitation is set to 2. Additionally, A2 can grant the privilege to other accounts with vertical propagation set to 0 (no GRANT OPTION), 1, or 2; this is because A2 must *reduce the vertical propagation by at least 1 when passing the privilege* to others.
- In addition, the horizontal propagation must be less than or equal to the originally granted horizontal propagation. For example, if account A grants a privilege to account B with the horizontal propagation set to an integer number $j > 0$, this means that B can grant the privilege to other accounts only with a horizontal propagation *less than* or *equal to j*.

Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges



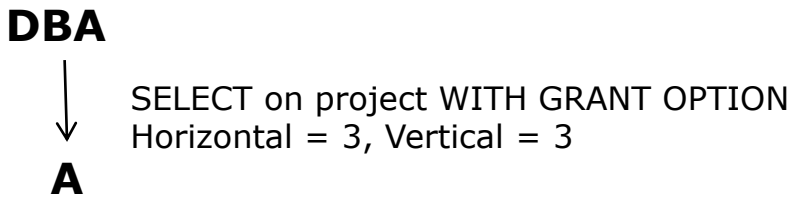
GRANT graph (AUTHORIZATION graph)
that shows propagation of privileges with GRANT OPTION

Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - DBA grants SELECT on PROJECT to user A with GRANT OPTION using horizontal propagation = 3 and vertical propagation = 3.
    - User A grants SELECT on PROJECT to user B with GRANT OPTION using horizontal propagation = 2 and vertical propagation = 1.
    - User A grants SELECT on PROJECT to user C with no GRANT OPTION.
    - User A grant SELECT on PROJECT to user D with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 2.
    - User B grants SELECT on PROJECT to user E with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 0.
    - User D grants SELECT on PROJECT to user F with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 1.

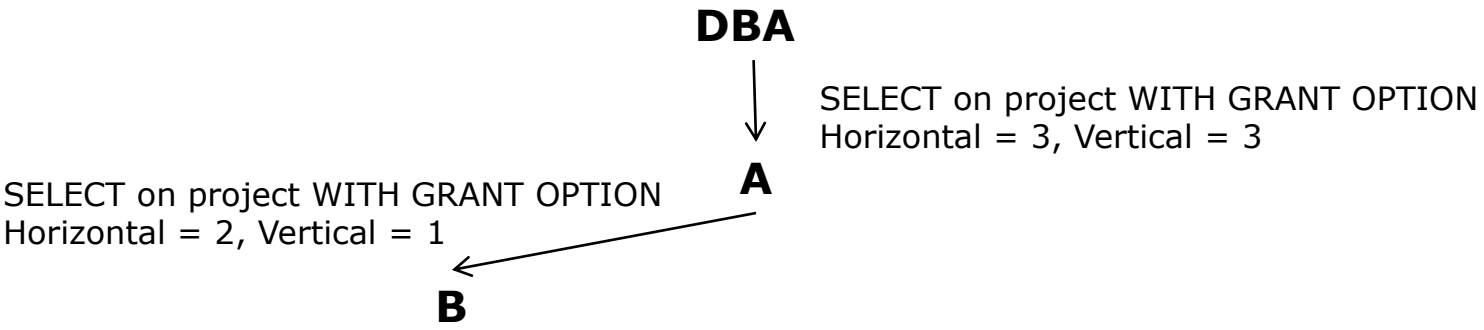
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - DBA grants SELECT on PROJECT to user A with GRANT OPTION using horizontal propagation = 3 and vertical propagation = 3.



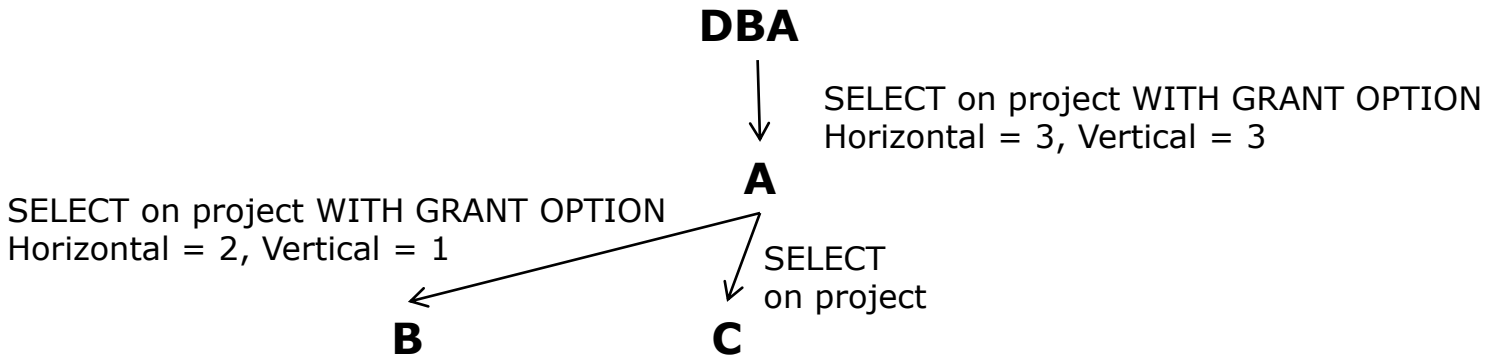
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User A grants SELECT on PROJECT to user B with GRANT OPTION using horizontal propagation = 2 and vertical propagation = 1.



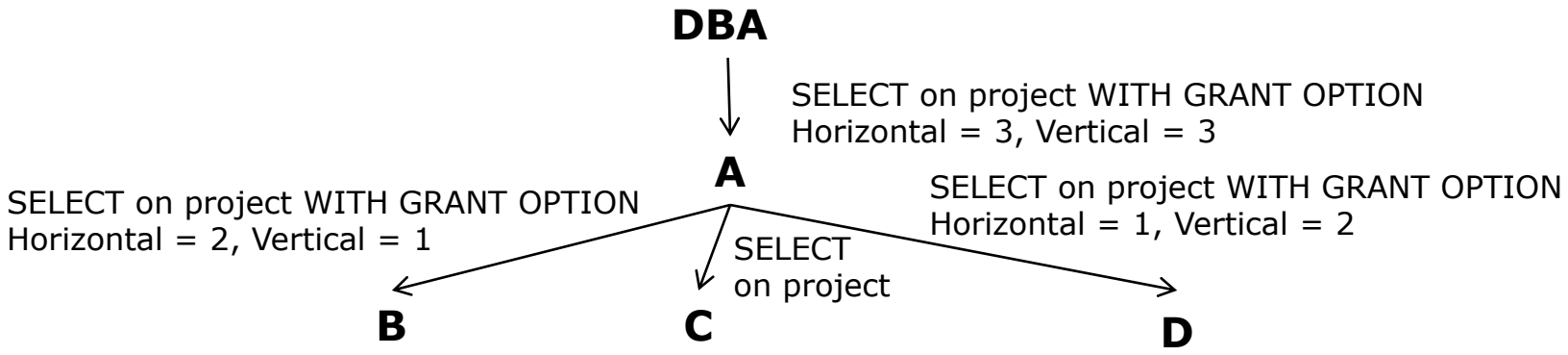
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User A grants SELECT on PROJECT to user C with no GRANT OPTION.



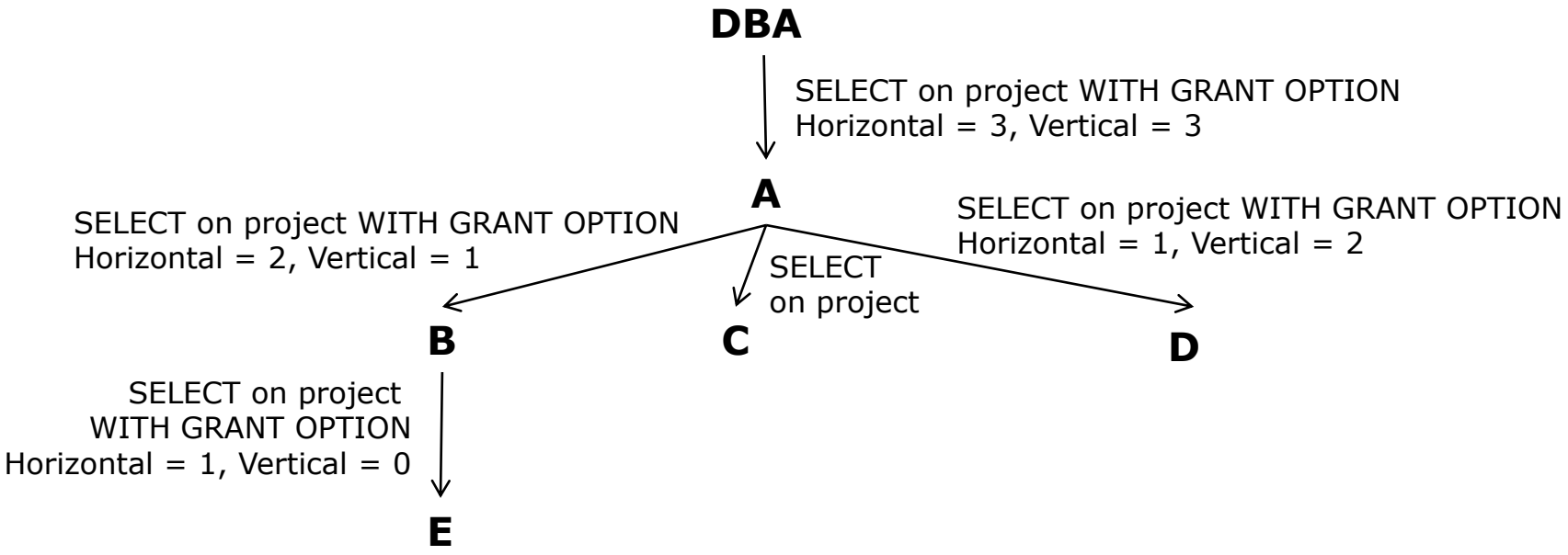
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User A grant SELECT on PROJECT to user D with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 2.



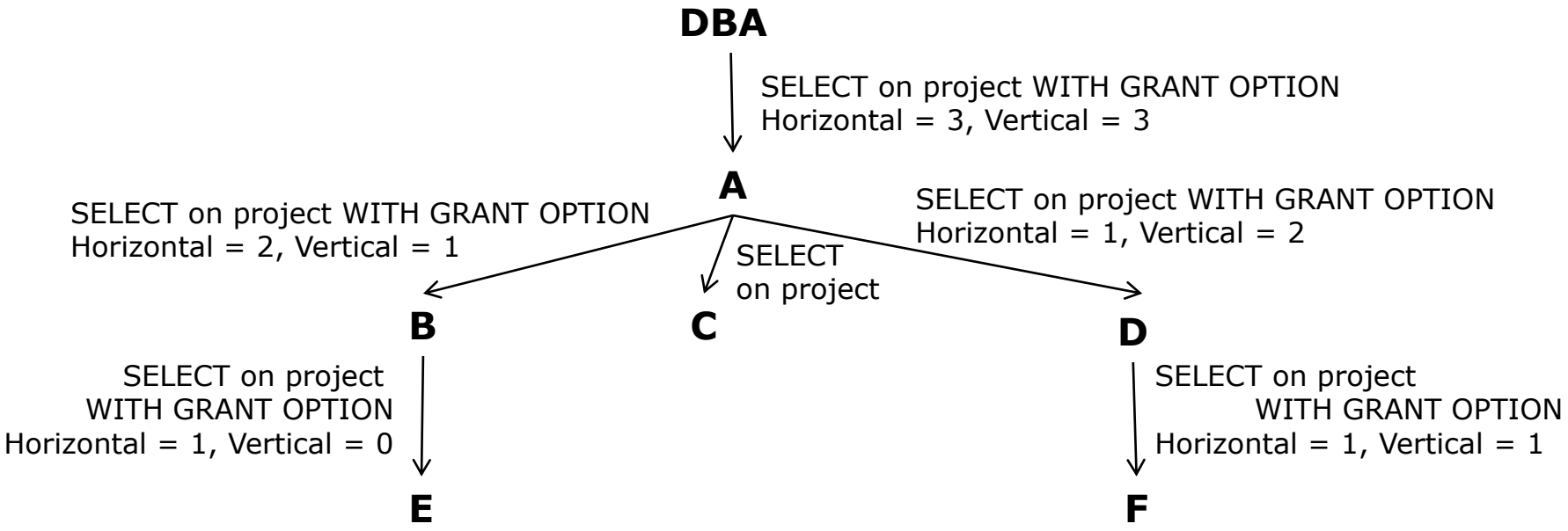
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User B grants SELECT on PROJECT to user E with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 0.



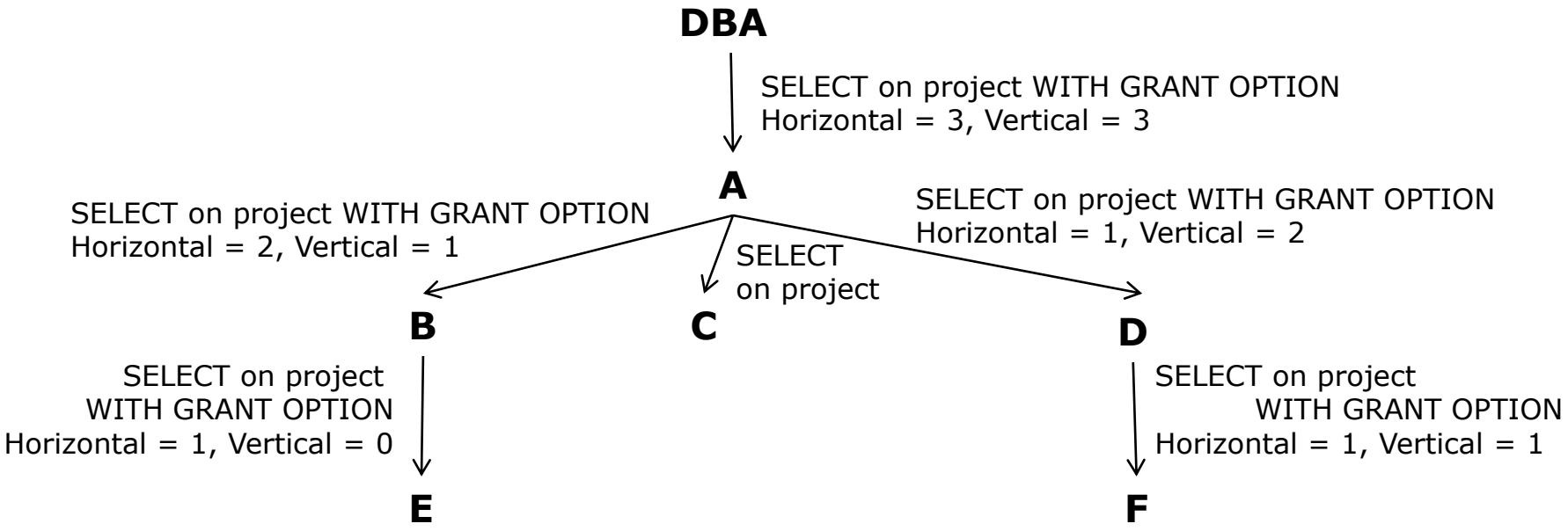
Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User D grants SELECT on PROJECT to user F with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 1.



Discretionary access control based on granting and revoking privileges

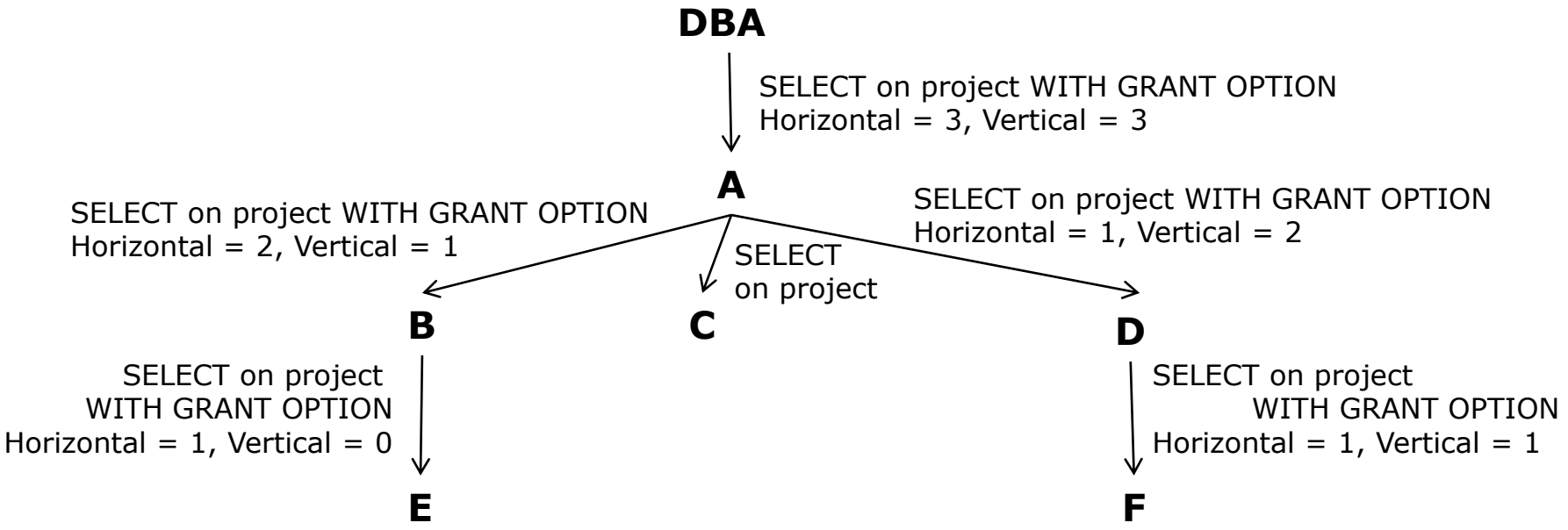
- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User D grants SELECT on PROJECT to user F with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 1.



Who can *keep granting* SELECT on PROJECT *to others*?

Discretionary access control based on granting and revoking privileges

- Specifying Limits on Propagation of Privileges
  - Draw a **grant graph** to show propagation of privileges SELECT on PROJECT of the COMPANY database as follows:
    - User D grants SELECT on PROJECT to user F with GRANT OPTION using horizontal propagation = 1 and vertical propagation = 1.



Who can *keep granting* SELECT on PROJECT *to others*? B, F

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Syntax of GRANT/ REVOKE

```
GRANT
    priv_type [(column_list)]
      [, priv_type [(column_list)]] ...
    ON [object_type] priv_level
    TO user_or_role [, user_or_role] ...
    [WITH GRANT OPTION]
    [AS user
        [WITH ROLE
            DEFAULT
          | NONE
          | ALL
          | ALL EXCEPT role [, role ] ...
          | role [, role ] ...
        ]
    ]
}
```

priv_type : { *static* | *dynamic* privileges }

```
object_type: {
    TABLE
  | FUNCTION
  | PROCEDURE
}
```

```
priv_level: {
    *
  | *.*
  | db_name.*
  | db_name.tbl_name
  | tbl_name
  | db_name.routine_name
}
```

- *Static* privileges: built-in to the server
- *Dynamic* privileges: defined at runtime

Source: MySQL 8.0 Reference Manual, 2020.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Syntax of GRANT/ REVOKE

priv_type : { *static* | *dynamic* privileges }

Table 13.11 Permissible Static Privileges for GRANT and REVOKE

Privilege	Meaning and Grantable Levels
`ALL [PRIVILEGES]`	Grant all privileges at specified access level except `GRANT OPTION` and `PROXY`.
`ALTER`	Enable use of `ALTER TABLE`. Levels: Global, database, table.
`ALTER ROUTINE`	Enable stored routines to be altered or dropped. Levels: Global, database, routine.
`CREATE`	Enable database and table creation. Levels: Global, database, table.
`CREATE ROLE`	Enable role creation. Level: Global.
`CREATE ROUTINE`	Enable stored routine creation. Levels: Global, database.
`CREATE TABLESPACE`	Enable tablespaces and log file groups to be created, altered, or dropped. Level: Global.

Source: MySQL 8.0 Reference Manual, 2020.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Syntax of GRANT/ REVOKE

priv_type : { *static* | *dynamic* privileges }

Table 13.12 Permissible Dynamic Privileges for GRANT and REVOKE

Privilege	Meaning and Grantable Levels
APPLICATION_PASSWORD_ADMIN	Enable dual password administration. Level: Global.
AUDIT_ADMIN	Enable audit log configuration. Level: Global.
BACKUP_ADMIN	Enable backup administration. Level: Global.
BINLOG_ADMIN	Enable binary log control. Level: Global.
BINLOG_ENCRYPTION_ADMIN	Enable activation and deactivation of binary log encryption. Level: Global.
CLONE_ADMIN	Enable clone administration. Level: Global.
CONNECTION_ADMIN	Enable connection limit/restriction control. Level: Global.
ENCRYPTION_KEY_ADMIN	Enable InnoDB key rotation. Level: Global.
FIREWALL_ADMIN	Enable firewall rule administration, any user. Level: Global.

Source: MySQL 8.0 Reference Manual, 2020.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Syntax of GRANT/ REVOKE

```
REVOKE
    priv_type [(column_list)]
      [, priv_type [(column_list)]] ...
    ON [object_type] priv_level
    FROM user_or_role [, user_or_role] ...

REVOKE ALL [PRIVILEGES], GRANT OPTION
    FROM user_or_role [, user_or_role] ...
```

Check partial revocations with MySQL.

Source: MySQL 8.0 Reference Manual, 2020.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - a. Account A can retrieve or modify any relation except DEPENDENT and can grant any of these privileges to other users.
    - b. Account B can retrieve all the attributes of EMPLOYEE and DEPARTMENT except for Salary, Mgr_ssn, and Mgr_start_date.
    - c. Account C can retrieve or modify WORKS_ON but can only retrieve the Fname, Minit, Lname, and Ssn attributes of EMPLOYEE and the Pname and Pnumber attributes of PROJECT controlled by department Research.
    - d. Account D can retrieve any attribute of EMPLOYEE or DEPENDENT and can modify DEPENDENT except for Relationship attribute and can grant any of these privileges to other users.
    - e. Account E can retrieve or modify EMPLOYEE but only for EMPLOYEE tuples of the employees who work for projects and departments in Houston.
    - f. Account F can retrieve or modify DEPARTMENT and DEPT_LOCATIONS but can only retrieve the Pnumber, Dnum, and Plocation attributes of PROJECT.
    - g. X wants to revoke the privileges from user A and the modification privileges on DEPENDENT from user D.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - a. Account A can retrieve or modify any relation except DEPENDENT and can grant any of these privileges to other users.

```
GRANT SELECT, INSERT, DELETE, UPDATE ON company.employee TO A WITH GRANT OPTION;

GRANT SELECT, INSERT, DELETE, UPDATE ON company.department TO A WITH GRANT OPTION;

GRANT SELECT, INSERT, DELETE, UPDATE ON company.dept_locations TO A WITH GRANT OPTION;

GRANT SELECT, INSERT, DELETE, UPDATE ON company.project TO A WITH GRANT OPTION;

GRANT SELECT, INSERT, DELETE, UPDATE ON company.works_on TO A WITH GRANT OPTION;
```

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - b. Account B can retrieve all the attributes of EMPLOYEE and DEPARTMENT except for Salary, Mgr_ssn, and Mgr_start_date.

```
GRANT SELECT(ssn, fname, minit, lname, bdate, sex, address, super_ssn, dno)
                                                ON company.employee TO B;

GRANT SELECT(dnumber, dname) ON company.department TO B;
```

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - c. Account C can retrieve or modify WORKS_ON but can only retrieve the Fname, Minit, Lname, and Ssn attributes of EMPLOYEE and the Pname and Pnumber attributes of PROJECT controlled by department Research.

```
GRANT SELECT, INSERT, DELETE, UPDATE ON company.works_on TO C;

GRANT SELECT(fname, minit, lname, ssn) ON company.employee TO C;

CREATE VIEW cProject AS
SELECT pname, pnumber
FROM project
WHERE dnum IN (SELECT dnumber
                    FROM department
                    WHERE dname = 'Research');

GRANT SELECT ON company.cProject TO C;
```

Check UPDATABLE VIEW in DBMSs for INSERT, DELETE, UPDATE on views.

CREATE VIEW (MySQL)

```
CREATE
[OR REPLACE]
[ALGORITHM = {UNDEFINED | MERGE | TEMPTABLE}]
[DEFINER = user]
[SQL SECURITY { DEFINER | INVOKER }]
VIEW view_name [(column_list)]
AS select_statement
[WITH [CASCADED | LOCAL] CHECK OPTION]
```

Some views are updatable. That is, you can use them in statements such as `UPDATE`, `DELETE`, or `INSERT` to update the contents of the underlying table. For a view to be updatable, there must be a one-to-one relationship between the rows in the view and the rows in the underlying table. There are also certain other constructs that make a view nonupdatable.

The `WITH CHECK OPTION` clause can be given for an updatable view to prevent inserts to rows for which the `WHERE` clause in the *`select_statement`* is not true. It also prevents updates to rows for which the `WHERE` clause is true but the update would cause it to be not true (in other words, it prevents visible rows from being updated to nonvisible rows).

In a `WITH CHECK OPTION` clause for an updatable view, the `LOCAL` and `CASCADED` keywords determine the scope of check testing when the view is defined in terms of another view. The `LOCAL` keyword restricts the `CHECK OPTION` only to the view being defined. `CASCADED` causes the checks for underlying views to be evaluated as well. When neither keyword is given, the default is `CASCADED`.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - d. Account D can retrieve any attribute of EMPLOYEE or DEPENDENT and can modify DEPENDENT except for Relationship attribute and can grant any of these privileges to other users.

```
GRANT SELECT ON company.employee TO D WITH GRANT OPTION;

GRANT SELECT ON company.dependent TO D WITH GRANT OPTION;


GRANT INSERT(essn, dependent_name, bdate, sex),
        DELETE, UPDATE(essn, dependent_name, bdate, sex)
                ON company.dependent TO D WITH GRANT OPTION;
```

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - e. Account E can retrieve or modify EMPLOYEE but only for EMPLOYEE tuples of the employees who work for projects and departments in Houston.

```
CREATE VIEW eEmployee AS
SELECT *
FROM employee
WHERE ssn IN (SELECT DISTINCT essn
                    FROM works_on JOIN project ON pno = pnumber
                    WHERE plocation = 'Houston')
AND dno IN (SELECT dnumber
                    FROM dept_locations
                    WHERE dlocation = 'Houston');

GRANT SELECT, INSERT, DELETE, UPDATE ON company.eEmployee TO E;
```

Check UPDATABLE VIEW in DBMSs for INSERT, DELETE, UPDATE on views.

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - f. Account F can retrieve or modify DEPARTMENT and DEPT_LOCATIONS but can only retrieve the Pnumber, Dnum, and Plocation attributes of PROJECT.

```
GRANT SELECT, INSERT, DELETE, UPDATE ON company.department TO F;

GRANT SELECT, INSERT, DELETE, UPDATE ON company.dept_locations TO F;

CREATE VIEW fProject AS
SELECT pnumber, dnum, plocation
FROM project;

GRANT SELECT ON company.fProject TO F;

-- instead of view fProject
GRANT SELECT (pnumber, dnum, plocation) ON company.project TO F;
```

Discretionary access control based on granting and revoking privileges

- Demonstration on MySQL
  - Some requirements on the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user *root*, who wants to grant/ revoke the following privileges to/ from user accounts A, B, C, D, E, and F:
    - g. *root* wants to revoke the privileges from user A and the modification privileges on WORKS_ON from user C.

```
REVOKE SELECT, INSERT, DELETE, UPDATE ON company.employee FROM A;
REVOKE SELECT, INSERT, DELETE, UPDATE ON company.department FROM A;
REVOKE SELECT, INSERT, DELETE, UPDATE ON company.dept_locations FROM A;
REVOKE SELECT, INSERT, DELETE, UPDATE ON company.project FROM A;
REVOKE SELECT, INSERT, DELETE, UPDATE ON company.works_on FROM A;

-- REVOKE ALL PRIVILEGES ON company.* FROM A;


REVOKE INSERT, DELETE, UPDATE ON company.works_on FROM C;
```

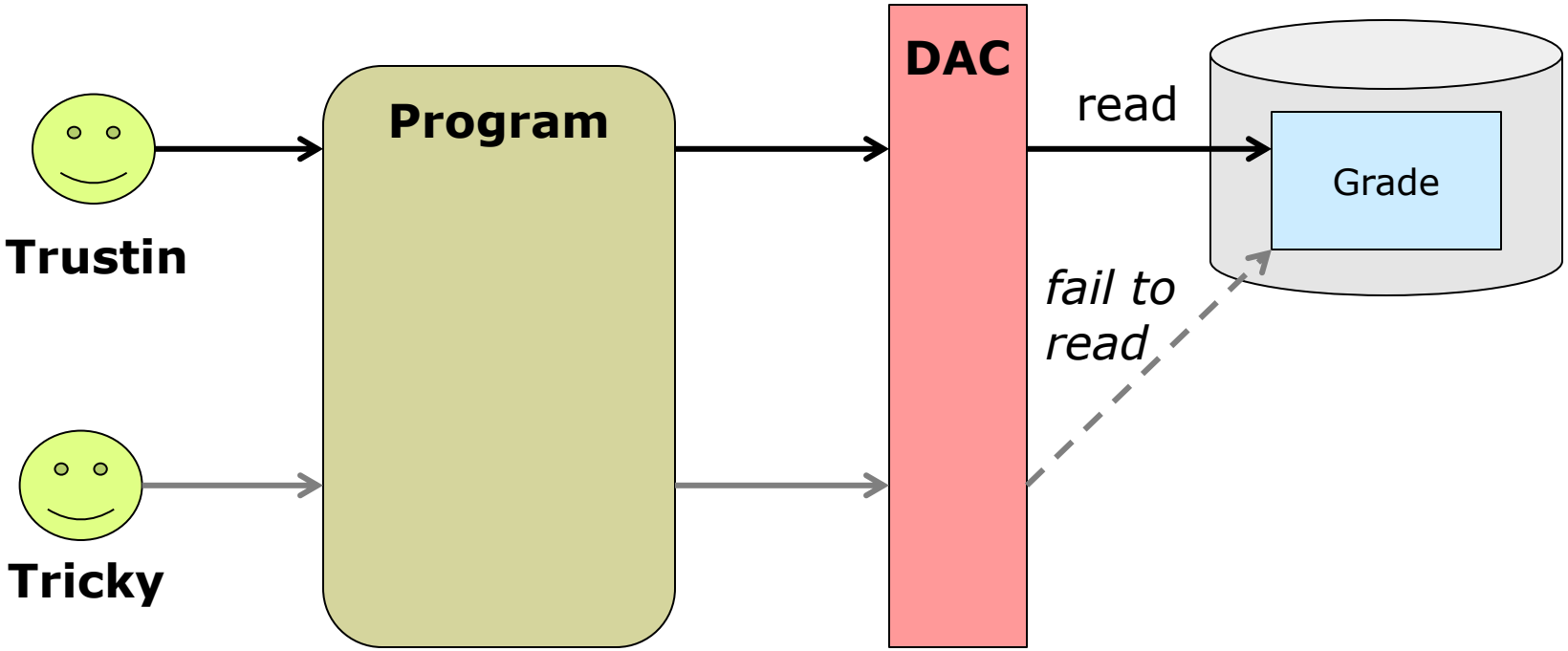
-- Check how well MySQL works with ***GRANT OPTION***.

Discretionary access control based on granting and revoking privileges

- *Strengths* of DAC for database security
  - Ensure data availability for *authorized* users
- *Weaknesses* of DAC for database security
  - An *all-or-nothing* method: A user either has or does not have a certain privilege.
    - A *finer* privilege granularity
  - *Vulnerability* to malicious attacks, such as *Trojan horses* embedded in application programs
    - Discretionary authorization models *do not impose any control on how information is propagated and used* once it has been accessed by users authorized to do so.

Discretionary access control based on granting and revoking privileges

- *Weaknesses* of DAC for database security
  - *Trojan horses* embedded in application programs

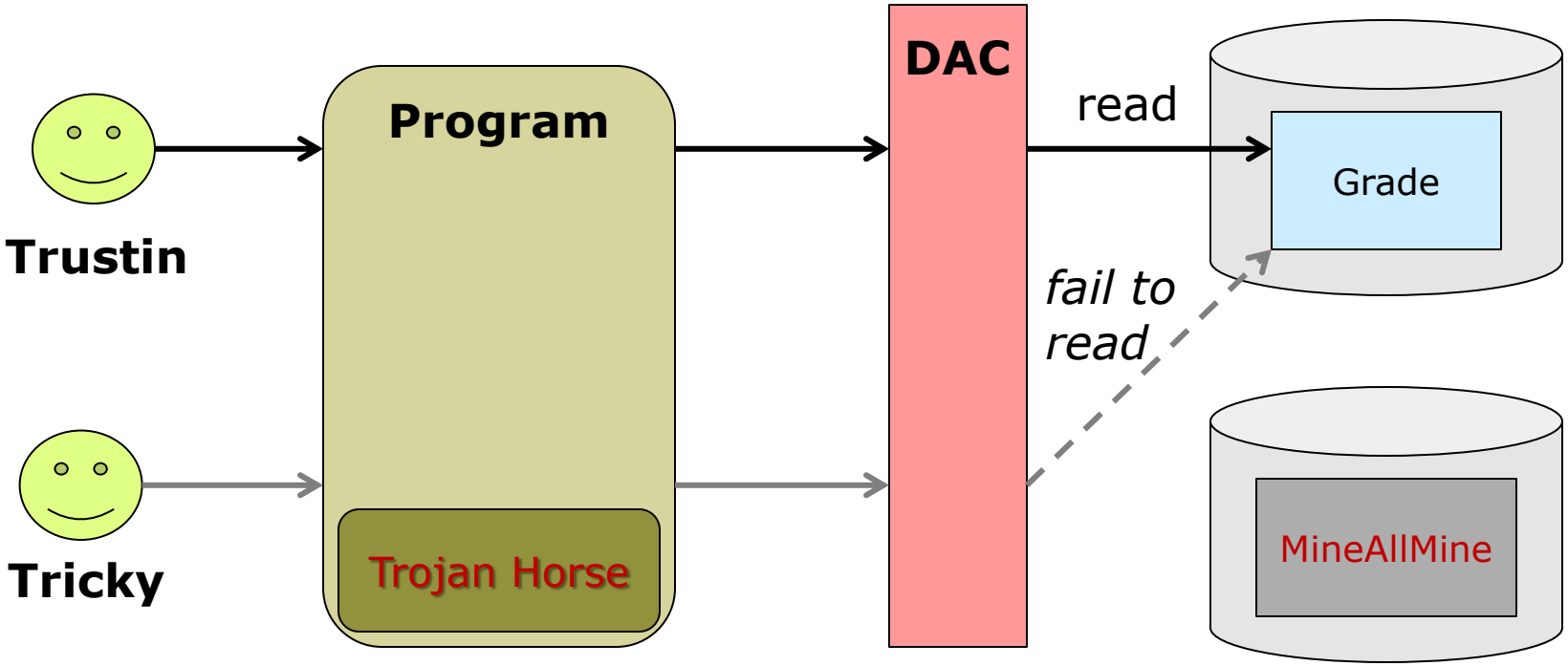


Tricky *failed to read* Grade table!

Source: R. Ramakrishnan, J. Gehrke, *Database Management Systems - 2nd Edition,* McGraw-Hill, 2000.

Discretionary access control based on granting and revoking privileges

- *Weaknesses* of DAC for database security
  - *Trojan horses* embedded in application programs

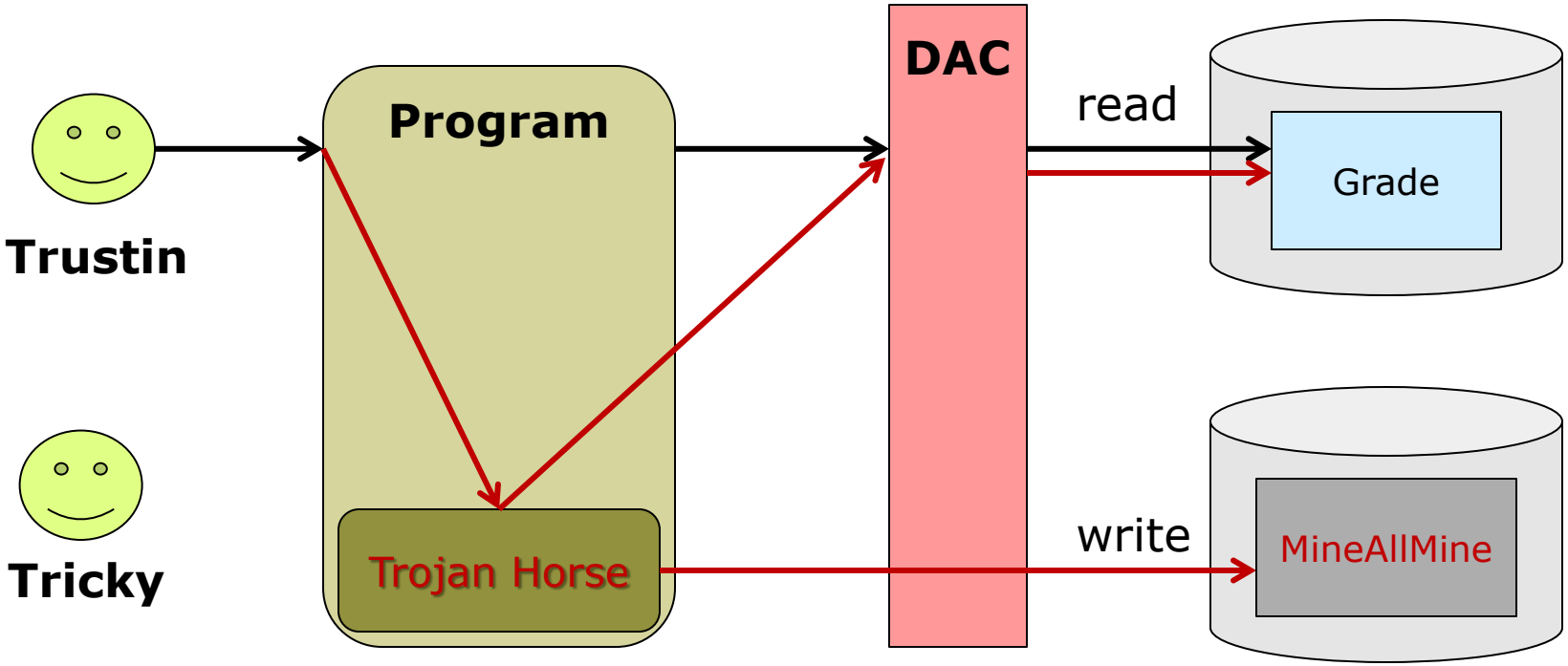


Tricky modified Program with *Trojan Horse* and *MineAllMine*.

Source: R. Ramakrishnan, J. Gehrke, *Database Management Systems - 2nd Edition,* McGraw-Hill, 2000.

Discretionary access control based on granting and revoking privileges

- *Weaknesses* of DAC for database security
  - *Trojan horses* embedded in application programs



Tricky is now *able to read* Grade table via *MineAllMine* table!

Source: R. Ramakrishnan, J. Gehrke, *Database Management Systems - 2nd Edition,* McGraw-Hill, 2000.

Mandatory access control and role-based access control for multilevel security

- *Mandatory access control* (MAC): Controls access based on comparing security labels (which indicate how sensitive or critical system resources are) with security clearances (which indicate system entities are eligible to access certain resources).
  - This policy is termed *mandatory* because an entity that has clearance to access a resource may not, just by its own volition (i.e. *the power to make own decisions*, *quyền tự quyết*), enable another entity to access that resource.
- *Role-based access control* (RBAC): Controls access based on the roles that users have within the system and on rules stating what accesses are allowed to users in given roles.
- *Attribute-based access control* (ABAC): Controls access based on attributes of the user, the resource to be accessed, and current environmental conditions.

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)
  - An approach that adds *security classes of data and users* into the discretionary access control mechanism
    - *Fine-grained* access modes instead of the *binary* mode
    - *Multilevel security* (MLS, *bảo mật đa mức*)
  - The commonly used model for multilevel security, known as the **Bell-LaPadula model**
    - Further reading:

      Bell, D., and LaPadula, L. "Secure Computer Systems: Mathematical Foundations." MTR–2547, Vol. I, The MITRE Corporation, Bedford, MA, 1 March 1973.

      Bell, D. "Looking Back at the Bell-Lapadula Model." Proceedings of the 21st Annual IEEE Computer Security Applications Conference, 2005.

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)
  - For simplicity, four security classification levels are used: *top secret* (TS), *secret* (S), *confidential* (C), and *unclassified* (U), where TS is the highest level and U the lowest.

$$TS \geq S \geq C \geq U$$

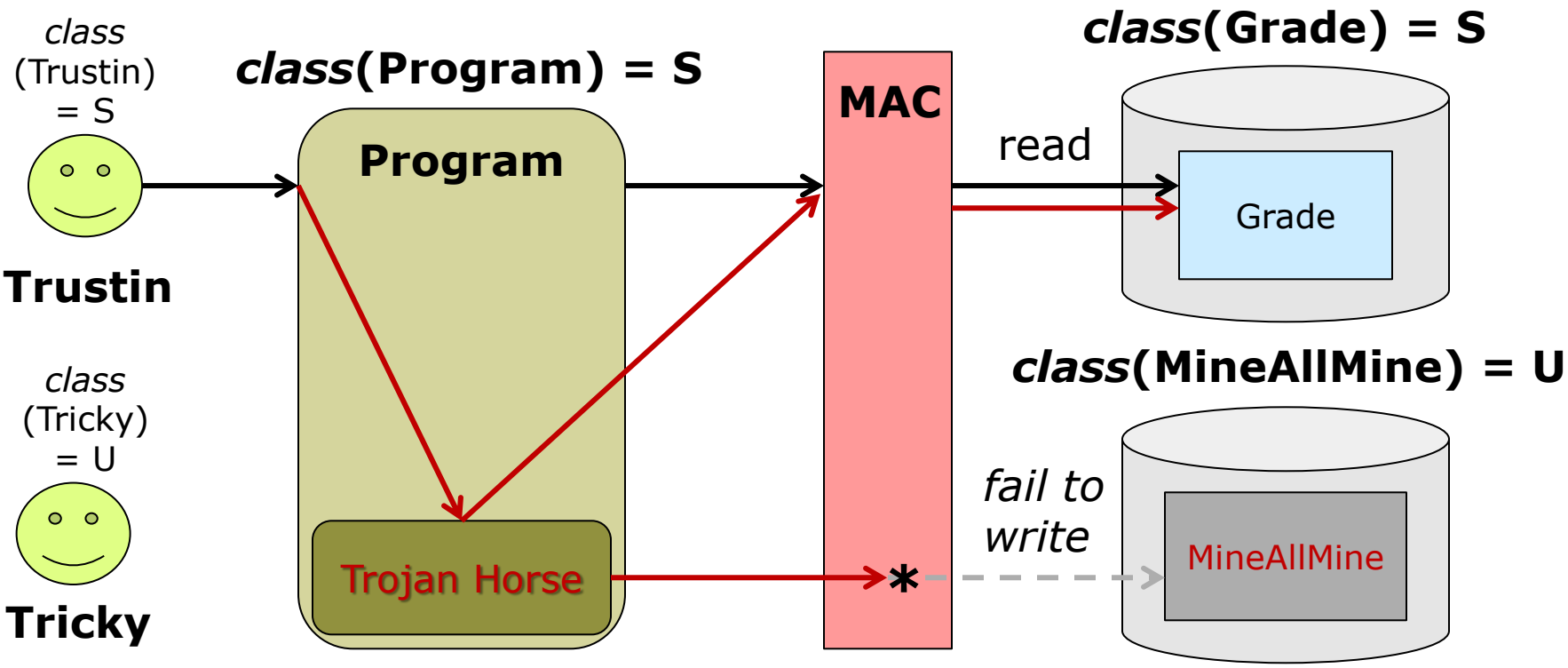
  - Each subject (user, account, program) and object (relation, tuple, column, view, operation) are classified into one of the security classifications TS, S, C, or U.
    - The clearance (classification) of a subject S is *class*(S).
      - Decided according to the *trustworthiness* of each subject
    - The classification of an object O is *class*(O).
      - Decided according to the *sensitivity* of each object

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)
  - The requirement for *confidentiality-centered multilevel security* is that *a subject at a high level may not convey information to a subject at a lower level* unless that flow accurately reflects the will of an authorized user as revealed by an authorized declassification.
    - *No information flows from higher to lower classifications*.
  - Multilevel security enforces:
    - No read up: A subject S can *only read* an object O of *less or equal* security level: *class*(S) ≥ *class*(O).
      - simple security property (ss-property)
    - No write down: A subject S can *only write* into an object O of *greater or equal* security level: *class*(S) ≤ *class*(O).
      - star-property (*-property)

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)



***-property** prevents Trojan Horse from writing more secure data into less secure one: *class*(Program) = S > *class*(MineAllMine) = U.

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)
  - The multilevel model

    = the relational data model + multilevel security
  - A *multilevel relation* schema R with n attributes would be represented as:

    $$R(A_1, C_1, A_2, C_2, \ldots, A_n, C_n, TC)$$

    - Each C_i represents the classification attribute associated with attribute A_i.
    - TC = a classification for the tuple *t* = the highest of all attribute classification values in *t* = max $\{C_1, C_2, \ldots, C_n\}$
    - The *apparent key* of a multilevel relation (*khóa biểu kiến của một quan hệ đa mức*) = the set of attributes that would have formed the *primary key* in a regular (single-level) relation

Mandatory access control and role-based access control for multilevel security

- The multilevel model
  - A multilevel relation will appear to contain different data to subjects (users) with different clearance levels.
  - In some cases, it is possible to store a single tuple in the relation at a higher classification level and produce the corresponding tuples at a lower-level classification through a process known as *filtering* (*quá trình lọc*).
    - After filtering, data query/ manipulation processing is normally conducted.
  - In other cases, it is necessary to store two or more tuples at different classification levels with the same value for the apparent key.
  - *Polyinstantiation* (*tính đa hình*): several tuples can have the same apparent key value but have different attribute values for users at different clearance levels.

Mandatory access control and role-based access control for multilevel security

- The multilevel model
  - The *entity integrity* rule for multilevel relations states that all attributes that are members of the apparent key must *not be null* and must have the *same security classification within each individual tuple*.
  - *All other attribute values in the tuple must have a security classification greater than or equal to the security classification of the apparent key*.
    - This constraint ensures that a user can see the key if the user is permitted to see any part of the tuple.
  - Other integrity rules, called *null integrity and interinstance integrity*, informally ensure that if a tuple value at some security level can be filtered (derived) from a higher-classified tuple, then it is sufficient to store the higher-classified tuple in the multilevel relation.

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an *apparent key*.

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

class(TC(tuple Smith)) = S = max {U, C, S}
class(TC(tuple Brown)) = S = max {C, S, C}

class(Name(tuple Smith)) = U = min {U, C, S}
class(Name(tuple Brown)) = C = min {C, S, C}

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an *apparent key*.

-- issued by TS users
SELECT * FROM EMPLOYEE;

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Appearance of EMPLOYEE after *filtering* for classification *TS* users

NO READ UP: class(Subject) ≥ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an *apparent key*.

-- issued by S users
SELECT * FROM EMPLOYEE;

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Appearance of EMPLOYEE after *filtering* for classification *S* users

EMPLOYEE

NO READ UP: class(Subject) ≥ class(Object)

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an *apparent key*.

-- issued by C users
SELECT * FROM EMPLOYEE;

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Appearance of EMPLOYEE after *filtering* for classification *C* users

EMPLOYEE

NO READ UP: class(Subject) ≥ class(Object)

Name	Salary	JobPerformance	TC
Smith U	40000 C	NULL C	C
Brown C	NULL C	Good C	C

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an *apparent key*.

-- issued by U users
SELECT * FROM EMPLOYEE;

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Appearance of EMPLOYEE after *filtering* for classification *U* users

NO READ UP: class(Subject) ≥ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	NULL U	NULL U	U

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an apparent key.

-- issued by TS users
UPDATE EMPLOYEE
SET Job_performance = 'Excellent'
WHERE Name = 'Smith';

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

No update for the 'Smith' tuple at the higher classification level *TS* of the user because the user is not allowed to overwrite the existing value of Job_performance at the lower classification level *S*

NO WRITE DOWN: class(Subject) ≤ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an apparent key.

-- issued by S users
UPDATE EMPLOYEE
SET Job_performance = 'Excellent'
WHERE Name = 'Smith';

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Update the 'Smith' tuple at the same classification level *S* of the user because the user is allowed to overwrite the existing value of Job_performance at the same classification level *S*

NO WRITE DOWN: class(Subject) ≤ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Excellent S	S
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an apparent key.

-- issued by C users
UPDATE EMPLOYEE
SET Job_performance = 'Excellent'
WHERE Name = 'Smith';

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Create a *polyinstantiation* for the 'Smith' tuple at the lower classification level *C* because the user is not allowed to overwrite the existing value of Job_performance at the higher classification level

NO WRITE DOWN: class(Subject) ≤ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- A multilevel relation: EMPLOYEE
  - Name is an apparent key.

-- issued by U users
UPDATE EMPLOYEE
SET Job_performance = 'Excellent'
WHERE Name = 'Smith';

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Brown C	80000 S	Good C	S

Create a *polyinstantiation* for the 'Smith' tuple at the lower classification level *U* because the user is not allowed to overwrite the existing value of Job_performance at the higher classification level

NO WRITE DOWN: class(Subject) ≤ class(Object)

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	NULL U	Excellent U	U
Brown C	80000 S	Good C	S

Mandatory access control and role-based access control for multilevel security

- Mandatory access control (MAC)
  - Mandatory policies ensure a high degree of protection—in a way, they *prevent any illegal flow of information* for *data confidentiality*.
  - MAC is suitable for military and high-security types of applications, which require a higher degree of protection.
  - Weaknesses
    - MAC is too rigid in that a *strict classification of subjects and objects into security levels* is required.
    - MAC is applicable to *few environments.*
    - *Few DBMSs* implemented MAC.

Mandatory access control and role-based access control for multilevel security

- Role-based access control (RBAC)
  - *Role*: a *named* set of privileges that can be granted to users or to other roles
  - Its basic notion is that privileges and other permissions are associated with organizational roles rather than with individual users.
  - Each user may have several different roles. However, they cannot be used simultaneously by the user. They are mutually exclusive.
  - Mutual exclusion of roles can be categorized into two types, namely authorization time exclusion (static) and runtime exclusion (dynamic).
    - In authorization time exclusion, two roles that have been specified as mutually exclusive cannot be part of a user's authorization at the same time.
    - In runtime exclusion, both these roles can be authorized to one user but cannot be activated by the user at the same time.

Mandatory access control and role-based access control for multilevel security

- Role-based access control (RBAC)
  - Multiple individuals can be assigned to each role.
  - Security privileges that are common to a role are granted to the role name, and any individual assigned to this role would automatically have those privileges granted.
  - RBAC can be used with traditional discretionary and mandatory access controls.
  - RBAC ensures that only authorized users in their specified roles are given access to certain data or resources.
  - Many DBMSs have allowed the concept of roles, where privileges can be assigned to roles.
    - Oracle, MS SQL Sever, MySQL, HBase, MongoDB, …

Inference control

- Statistical databases are used mainly to produce statistics about various populations.
- The database may contain confidential data about individuals; this information should be protected from user access.
- Users are permitted to retrieve statistical information about the populations, such as averages, sums, counts, maximums, minimums, and standard deviations.
- Users are not allowed to retrieve individual data.
- PROBLEM: Inference can be made from statistical information to *derive individual data*.

Inference control

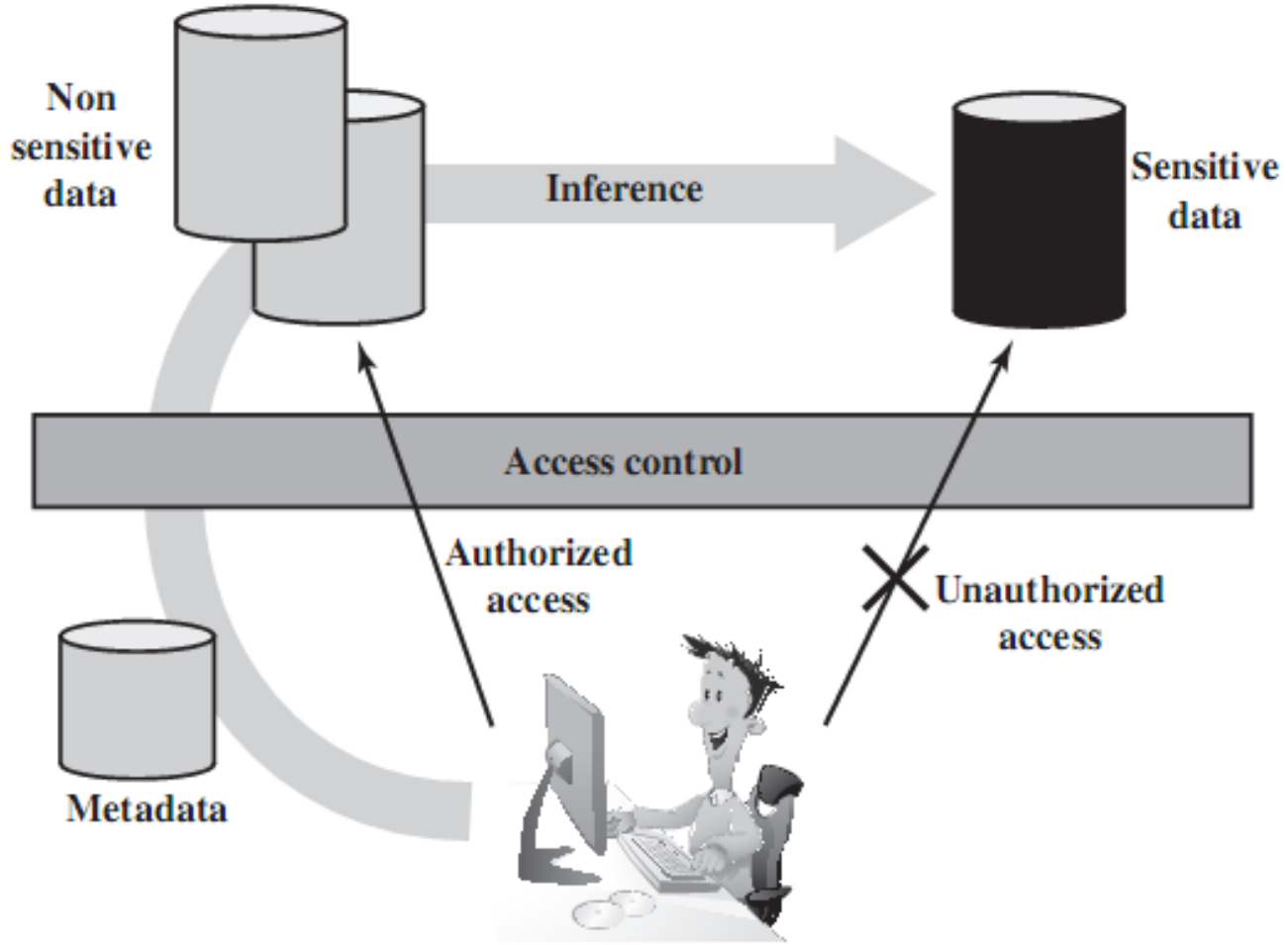


Figure 5.7 **Indirect Information Access via Inference Channel**

Source: W. Stallings, L. Brown, Computer Security - Principles and Practice, Third Edition, Pearson Education, Inc., 2015.

Inference control

- Inference control is needed to prohibit the retrieval of individual data from statistical databases.
- This is related to privacy protection of users in the statistical database.
- For example, the COMPANY database is not allowed to be used for the queries on individual employees' salaries, but the aggregation queries.

  What happens if we obtain the number of employees and their average salary in department 1?

Inference control

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

What happens if we obtain the number of employees and their average salary in department 1?

```
SELECT COUNT(*), AVG(salary)
FROM employee
WHERE dno = 1;
```

NumberOfEmployees	AverageSalary
1	55000.000000

If we know Borg with department 1, the average salary is his!

Inference control

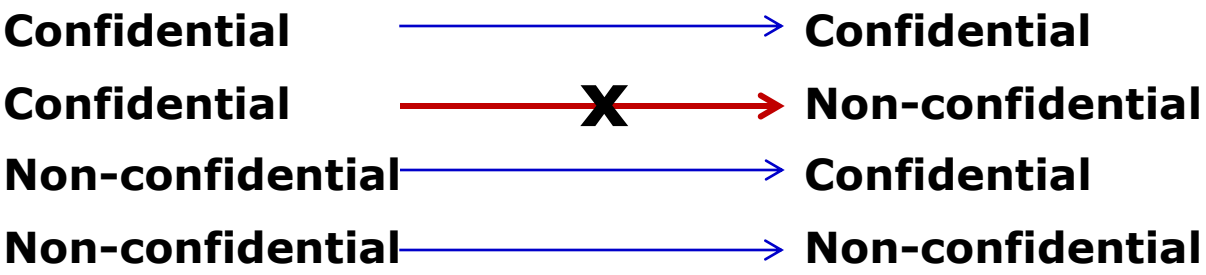
- *Solutions* to inference control
  - To disallow statistical queries whenever the *number of tuples in the population* specified by the selection condition falls below some threshold.
  - To prohibit sequences of queries that refer *repeatedly to the same population of tuples*.
  - To introduce slight *inaccuracies or noise* into the results of statistical queries deliberately, to make it difficult to deduce individual information from the results.
  - *Partitioning* of the database: records are stored in groups of some minimum size; queries can refer to any complete group or set of groups, but *never to subsets of records within a group*.

Flow control

- Flow control regulates the distribution or flow of information among accessible objects.
- Flow controls check that information contained in some objects does not flow explicitly or implicitly into *less* protected objects.
  - A flow between object X and object Y occurs when a program reads values from X and writes values into Y.
  - Flow control ensures that a user cannot get indirectly in Y what he or she cannot get directly in X.

Flow control

- Most flow controls employ some concept of security class; the transfer of information from a sender to a receiver is allowed only if the receiver's security class is at least as privileged as the sender's.
  - Examples of a flow control include preventing a service program from leaking a customer's confidential data, and blocking the transmission of secret military data to an unknown classified user.
- A flow policy specifies the channels along which information is allowed to move.
  - For example, the simplest flow policy specifies just two classes of information—confidential (C) and non-confidential (N)—and allows all flows except those from class C to class N.



Flow control

- Flow controls can be enforced by an extended access control mechanism, which involves *assigning a security class* (usually called the clearance) *to each running program*.
  - The program is allowed to read a particular memory segment only if its security class is as high as that of the segment.
  - The program is allowed to write in a segment only if its class is as low as that of the segment.
  - This automatically ensures that no information transmitted by the person can move from a higher to a lower class.
  - For example, a military program with a *secret* clearance can only *read* from objects that are *secret*, *unclassified, or confidential* and can only *write* into objects that are *secret and top secret*.

***Secret* program** —*read*→ ***Secret*/ *Unclassified*/ *Confidential* objects**

***Secret* program** —*write*→ ***Secret*/ *Top secret* objects**

Flow control

- Two types of flow can be distinguished:
  - *explicit* flows, which occur as a consequence of assignment instructions: Y:= $f(X_1, X_n)$
  - *implicit* flows, which are generated by conditional instructions: if $f(X_{m+1}, \ldots, X_n)$ then Y := $f(X_1, X_m)$
- Flow control mechanisms must verify that only authorized flows, both explicit and implicit, are executed.
- A *covert channel* (*kênh ẩn*) allows a transfer of information that violates the security or the policy, i.e. allows information to pass from a higher classification level to a lower classification level through improper means.
  - A *timing* channel: a channel where the information is conveyed (*giao tiếp*) by the timing of events or processes.
  - A *storage* channel: a channel where information is conveyed by accessing system information or what is otherwise inaccessible to the user.
- Covert channels are not a major problem in well-implemented robust database implementations.

Encryption and Public Key Infrastructure

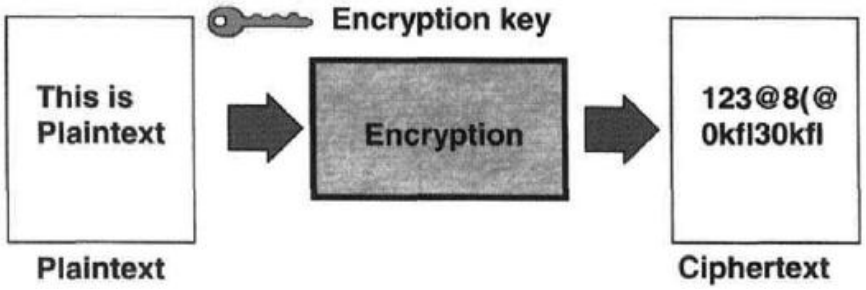
- Suppose data are being *transferred* (*communicated*), but data falls into the hands of a *nonlegitimate* user (*người dùng bất hợp pháp*).
- By using encryption, the message can be disguised (*bị thay đổi*) so that even if the transmission is diverted (*bị đổi hướng*), the message will not be revealed (*được biết*).
- Encryption is the *conversion of data* into a form, called a *ciphertext*, that cannot be easily understood by *unauthorized* persons. *Confidentiality* is ensured.
- Encryption enhances *security and privacy* when access controls are bypassed, because in cases of data loss or theft, encrypted data cannot be easily understood by unauthorized persons.

Encryption and Public Key Infrastructure

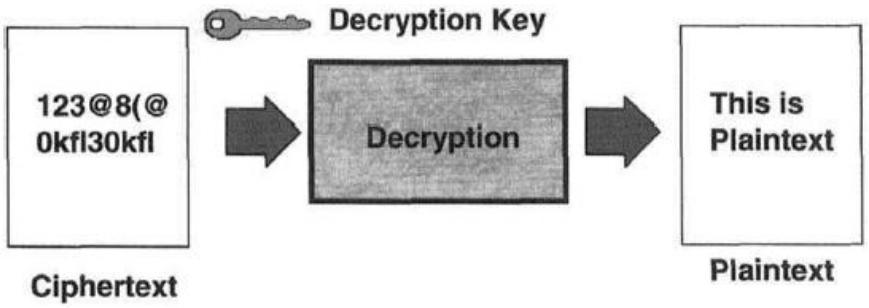
- Encryption/Cryptography
  - Definition: change of electronic information or signals into a secret code (=system of letters, numbers or symbols) that people cannot understand or use on normal equipment
  - Four parts of cryptography
    - Plaintext (*bản rõ*)
    - Ciphertext (*bản mã*)
    - Encryption algorithm (cipher/cryptosystem) (*mã hóa*)
    - Key (*khóa*)

Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption process (*quá trình mã hóa*)



  - Decryption process (*quá trình giải mã*)



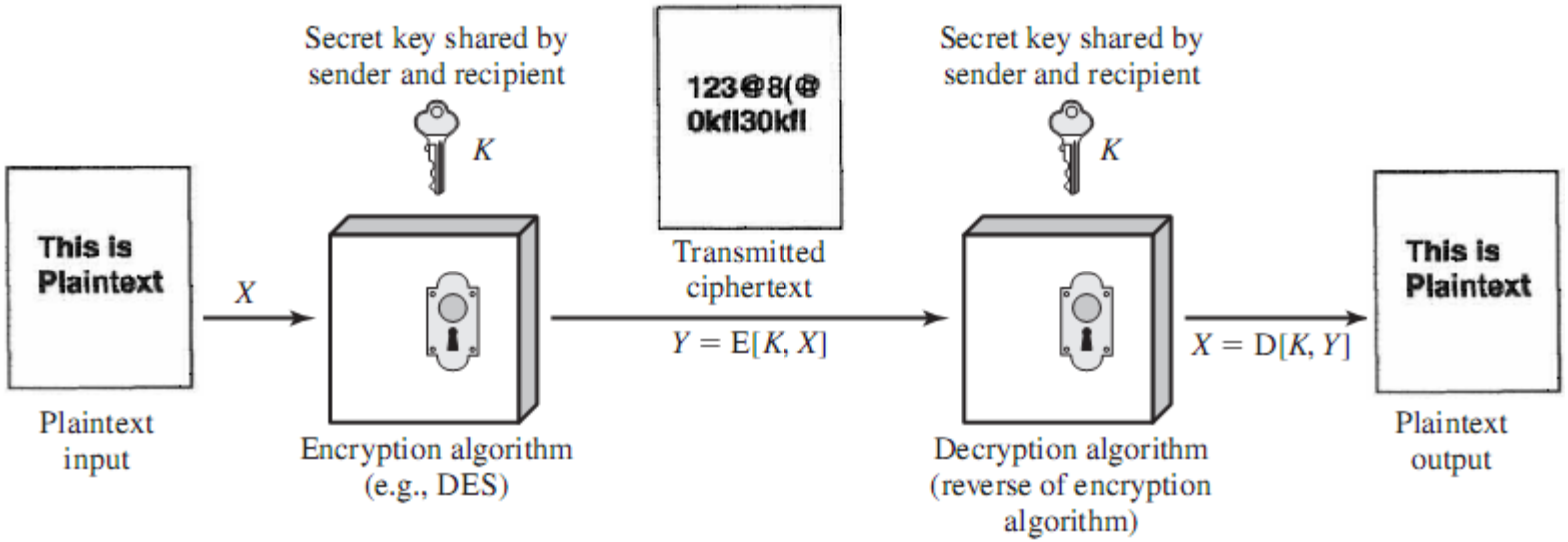
Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption types
    - Symmetric (private/secret) key encryption
      - The same secret key to encrypt/decrypt a message
      - Sender/receiver must exchange the key securely.
      - Algorithm: the Data Encryption Standard (DES) with a key length of 56 bits, Triple DES, the Advanced Encryption Standard (AES) by NIST, …

Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption types
    - Symmetric (private/secret) key encryption

(*mã hóa khóa đối xứng*)



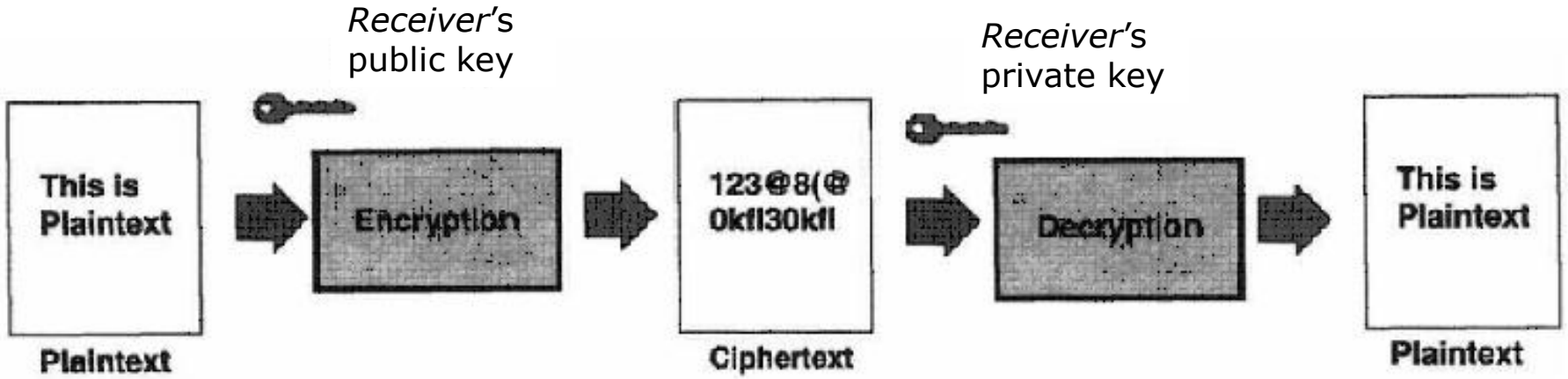
Key transfer and management?

Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption types
    - Asymmetric (public) key encryption
      - Two keys for encryption/decryption: a public key and a private key
      - The private key is kept secret by its owner.
      - The public key is freely distributed.
      - If the public key is used to encrypt a message, only the corresponding private key can decrypt the corresponding ciphertext, and vice versa.
      - Algorithm: the RSA public key algorithm (www.rsasecurity.com) with a key length of 512-1024 bits
        - One of the first public key schemes was introduced in 1978 by Ron Rivest, Adi Shamir, and Len Adleman at MIT.

Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption types
    - Asymmetric (public) key encryption
      (*mã hóa khóa bất đối xứng*)

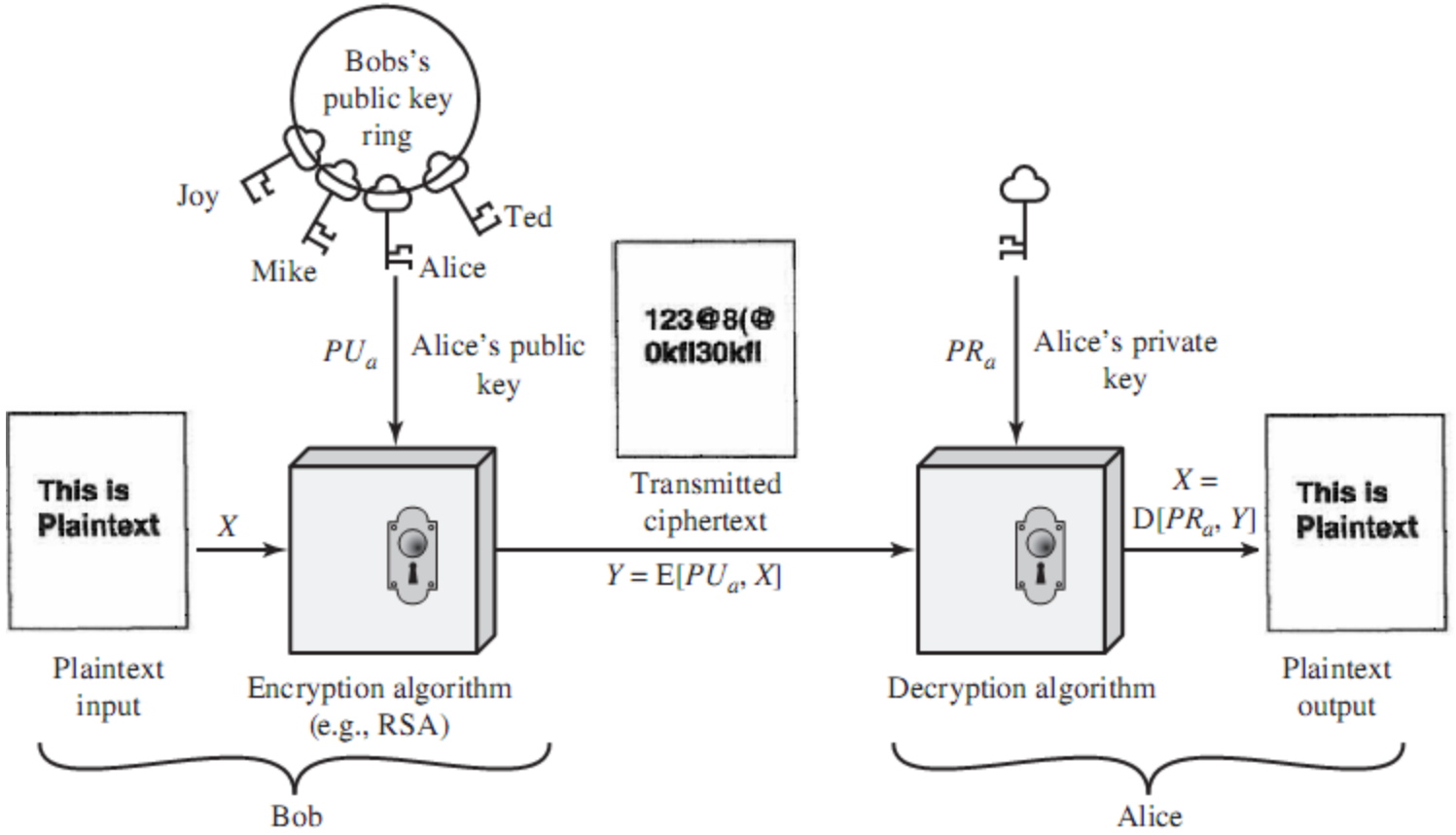


- Receiver's public key & receiver's private key
- Sender's private key & sender's public key

More complexity, slower than Symmetric Key Encryption!

Encryption and Public Key Infrastructure

- Asymmetric (public) key encryption

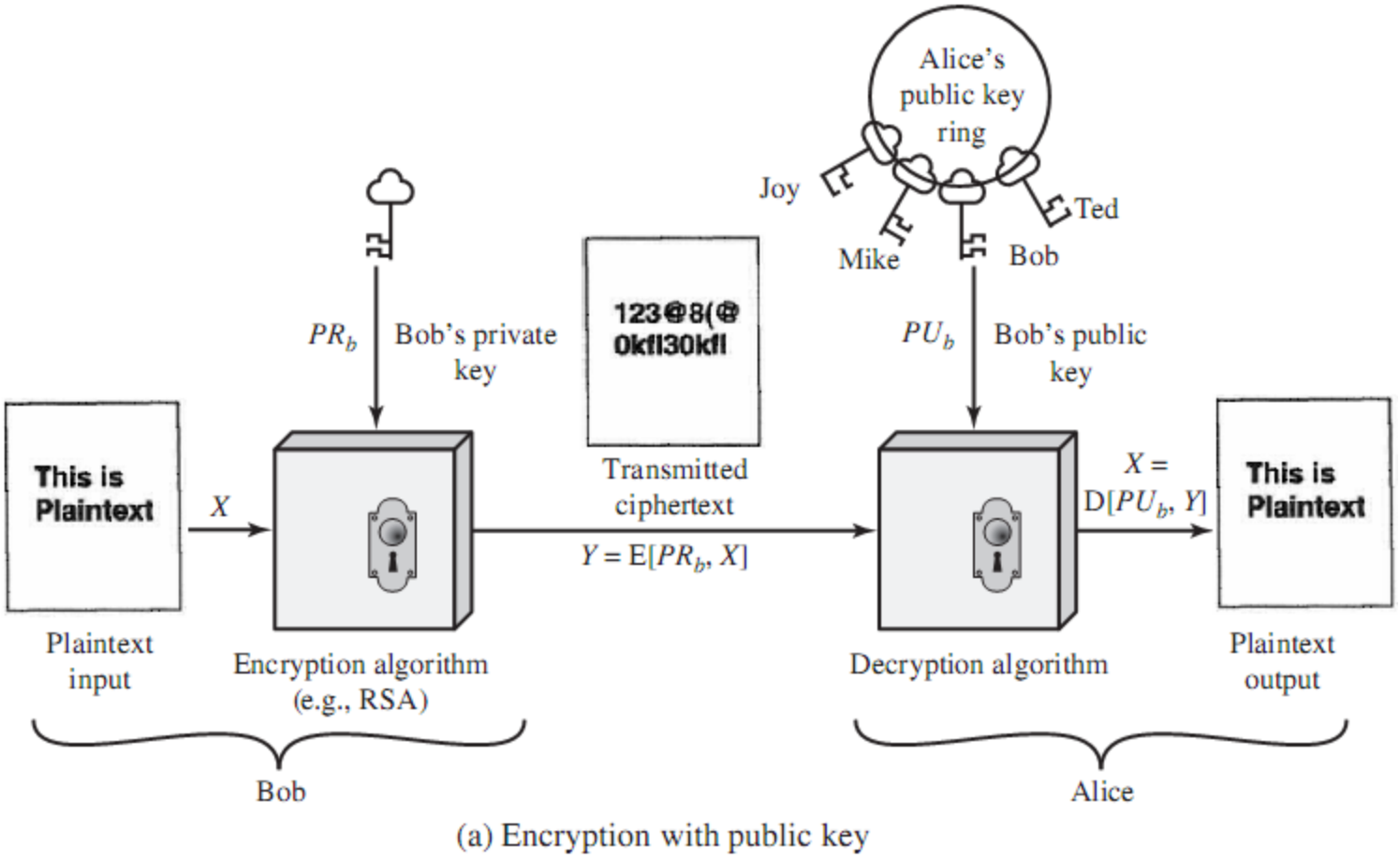


(a) Encryption with public key

Data Confidentiality

Encryption and Public Key Infrastructure

- Asymmetric (public) key encryption

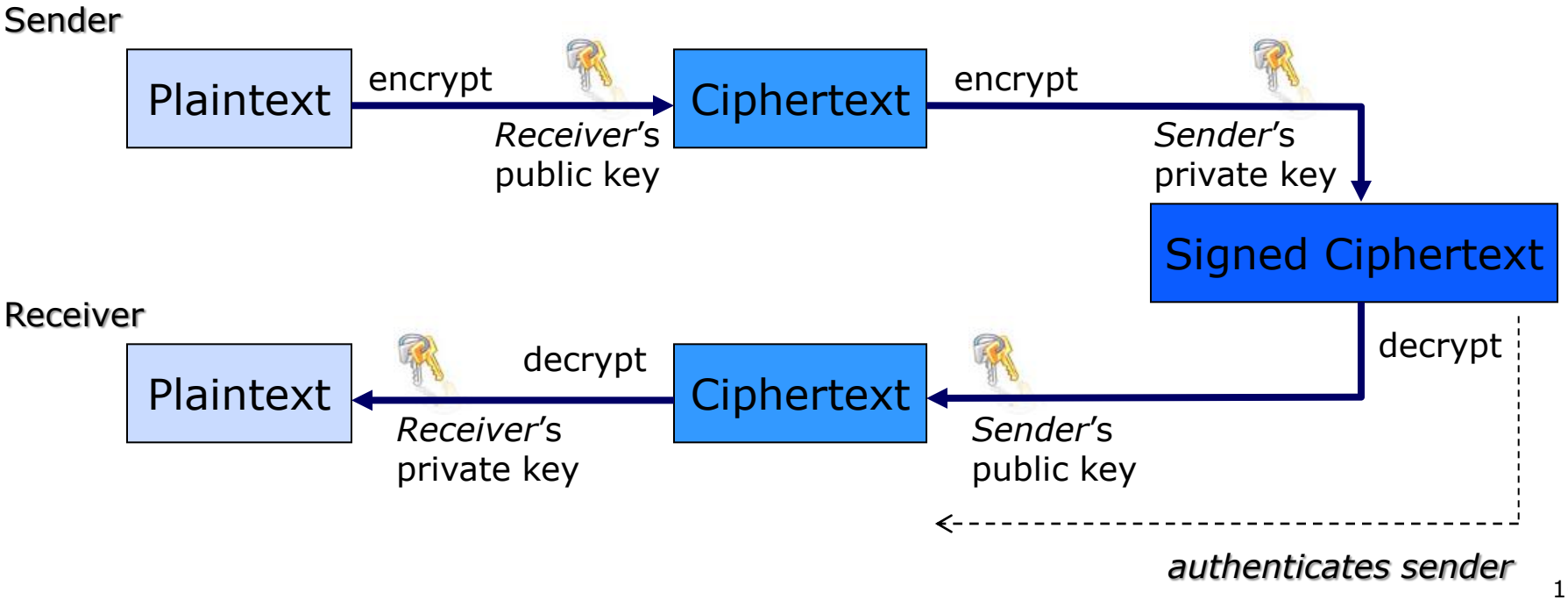


(a) Encryption with public key

Authentication / Data Integrity

Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Encryption types
    - Asymmetric (public) key encryption for ***authentication***

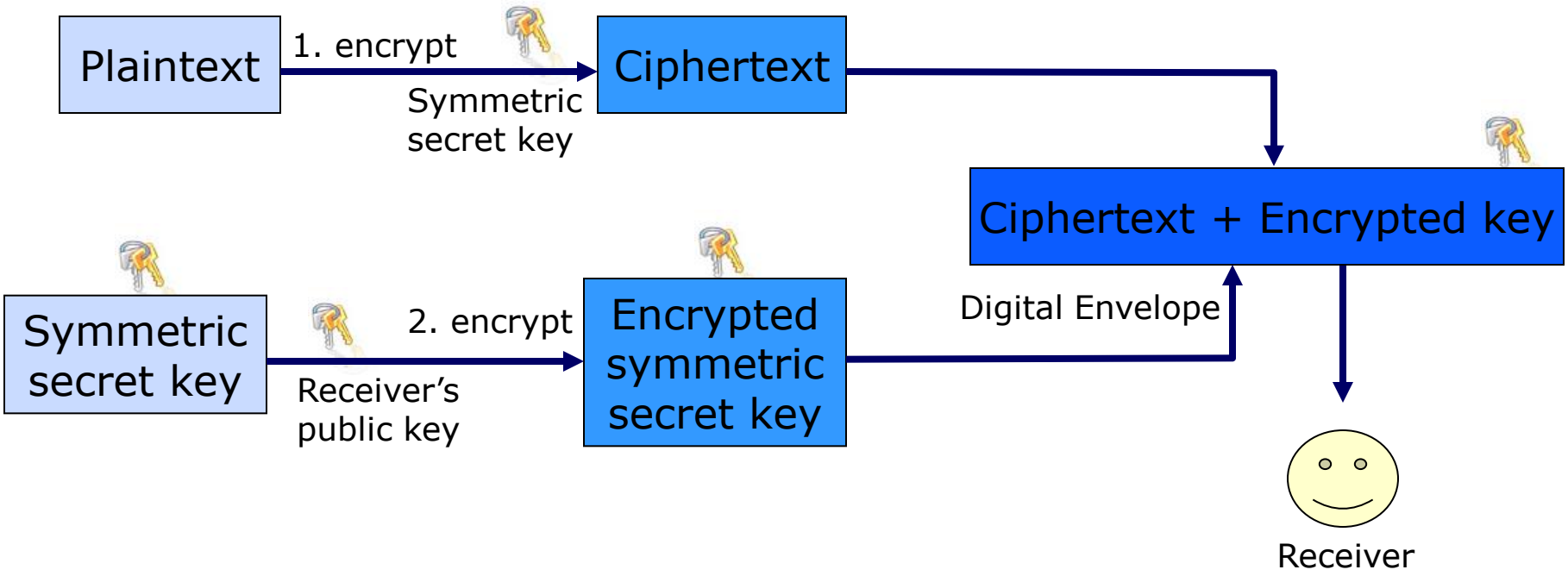


Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Key agreement protocols
    - Combination of symmetric and public key encryption
    - Rules for communication: exactly what encryption algorithm(s) is (are) going to be used?
    - The most common key agreement protocol to *exchange keys over an unsecure medium*: a digital envelope

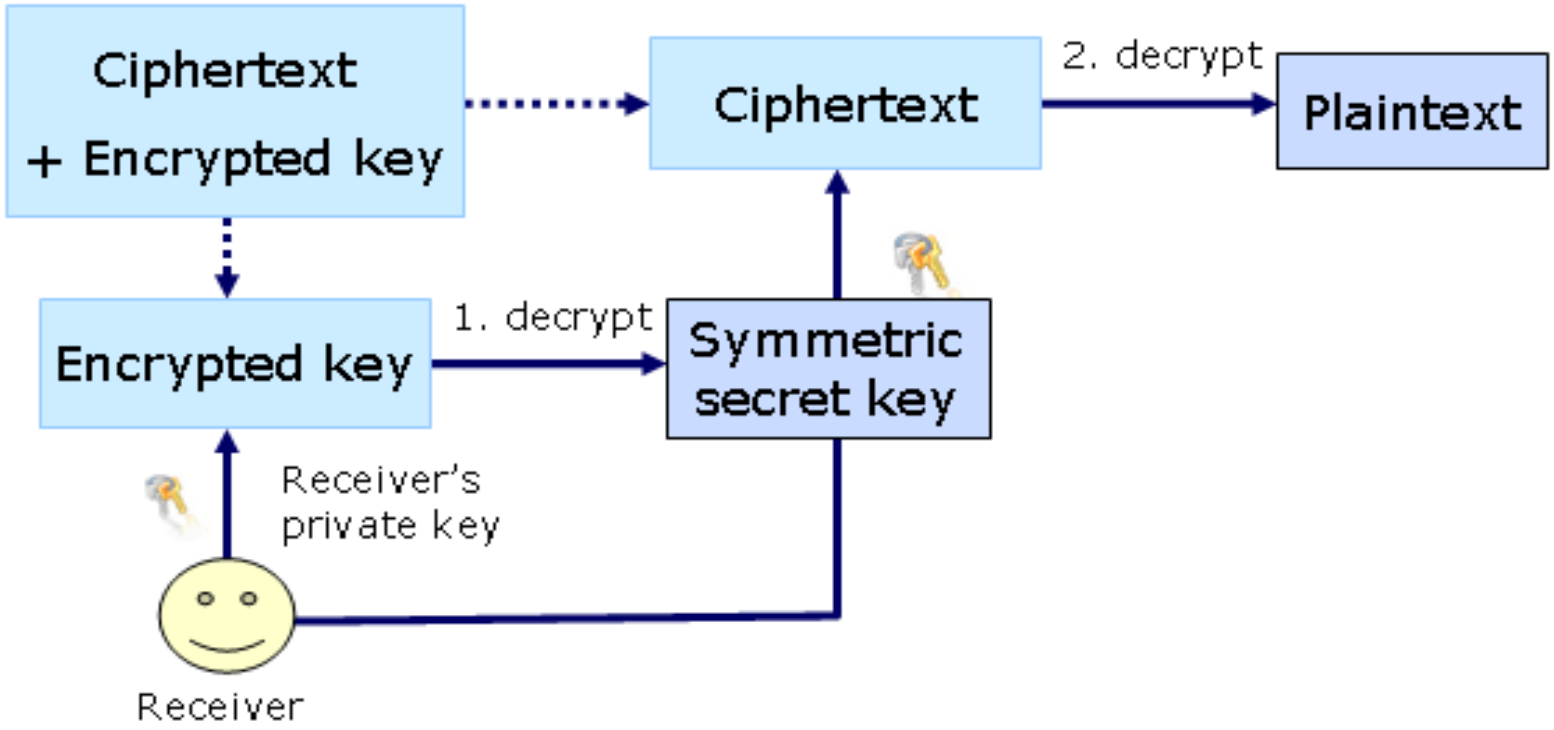
Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Digital Envelope (*phong bì số*)



Encryption and Public Key Infrastructure

- Encryption/Cryptography
  - Digital Envelope



Encryption and Public Key Infrastructure

- Electronic signatures (e-signatures, *chữ ký số*)
  - Definition: the electronic equivalent of written signatures for ***authentication***
  - Types of e-signatures
    - Signatures based on characters, digits, scanned images, sound, …
    - Signatures based on passwords, PIN codes, biometrics, …
    - Signatures based on public keys
  - *Public Key Infrastructure* (PKI), *Digital Certificates*, *Certification Authorities* (CA) to *authenticate* parties in a transaction

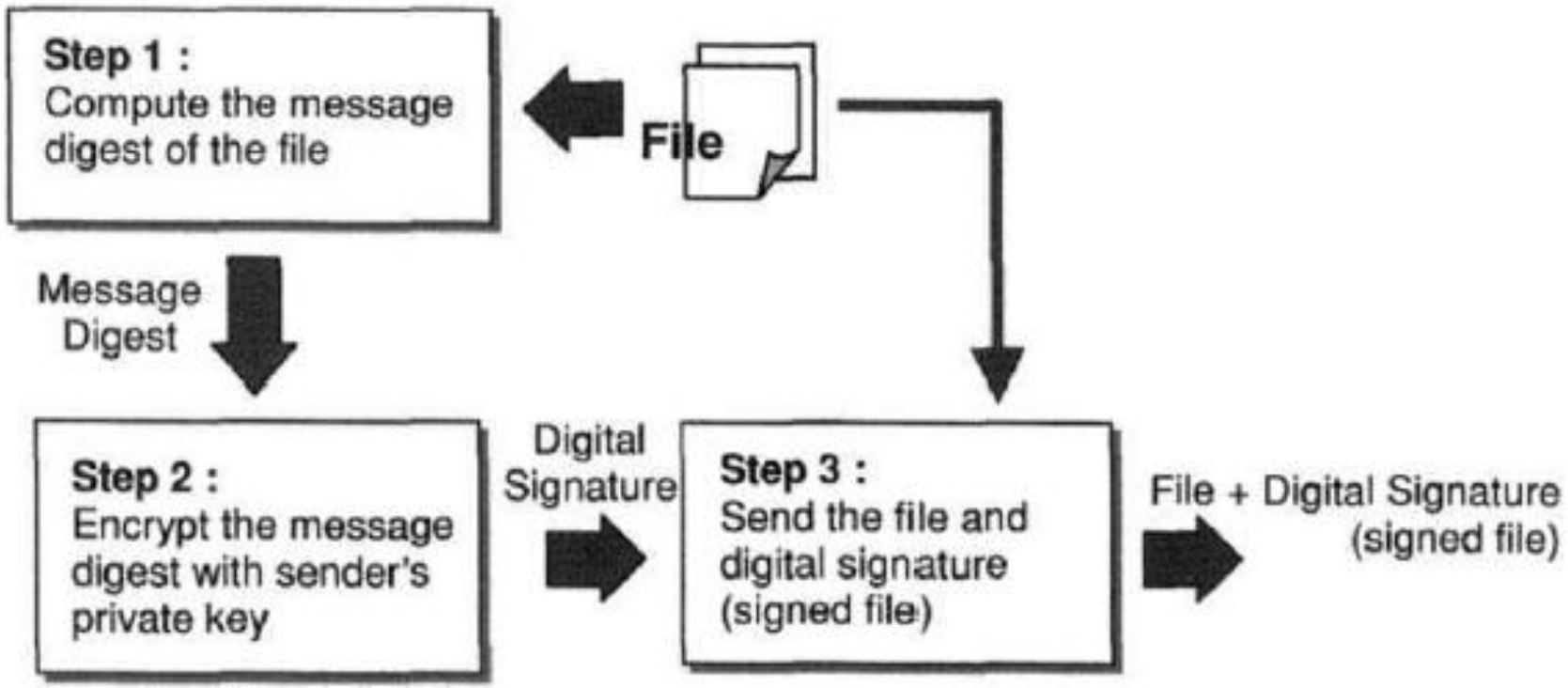
Encryption and Public Key Infrastructure

- Electronic signatures (e-signatures, *chữ ký số*)
  - Definition: the electronic equivalent of written signatures for ***authentication***
  - Types of e-signatures: Signatures based on public keys
  - *Public Key Infrastructure* (PKI, hạ tầng khóa công khai), *Digital Certificates (chứng thực số)*, *Certification Authorities* (CA, *bên cung cấp chứng thực số*) to *authenticate* parties in a transaction
    - A digital certificate is used to combine the value of a public key with the identity of the person or service that holds the corresponding private key into a digitally signed statement.
    - Certificates are issued and signed by a certification authority (CA), trusted by the parties.

Encryption and Public Key Infrastructure

- Digital signatures
  - Electronic signatures based on public keys for authentication, integrity, and nonrepudiation
  - Created using the contents of the document
  - Different for each document digitally signed
  - Associated with timestamping for nonrepudiation

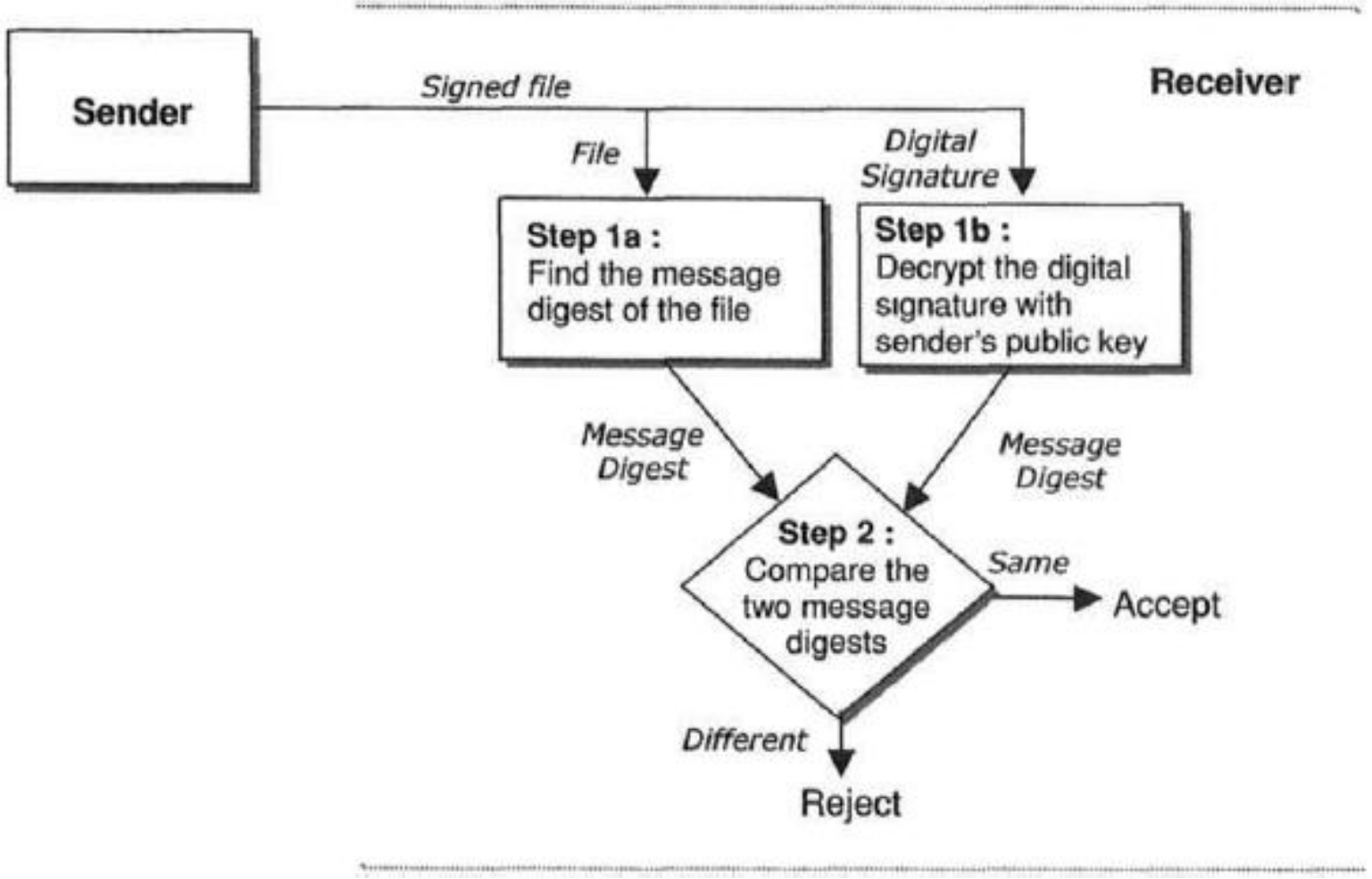
Encryption and Public Key Infrastructure



Steps in digital signature generation

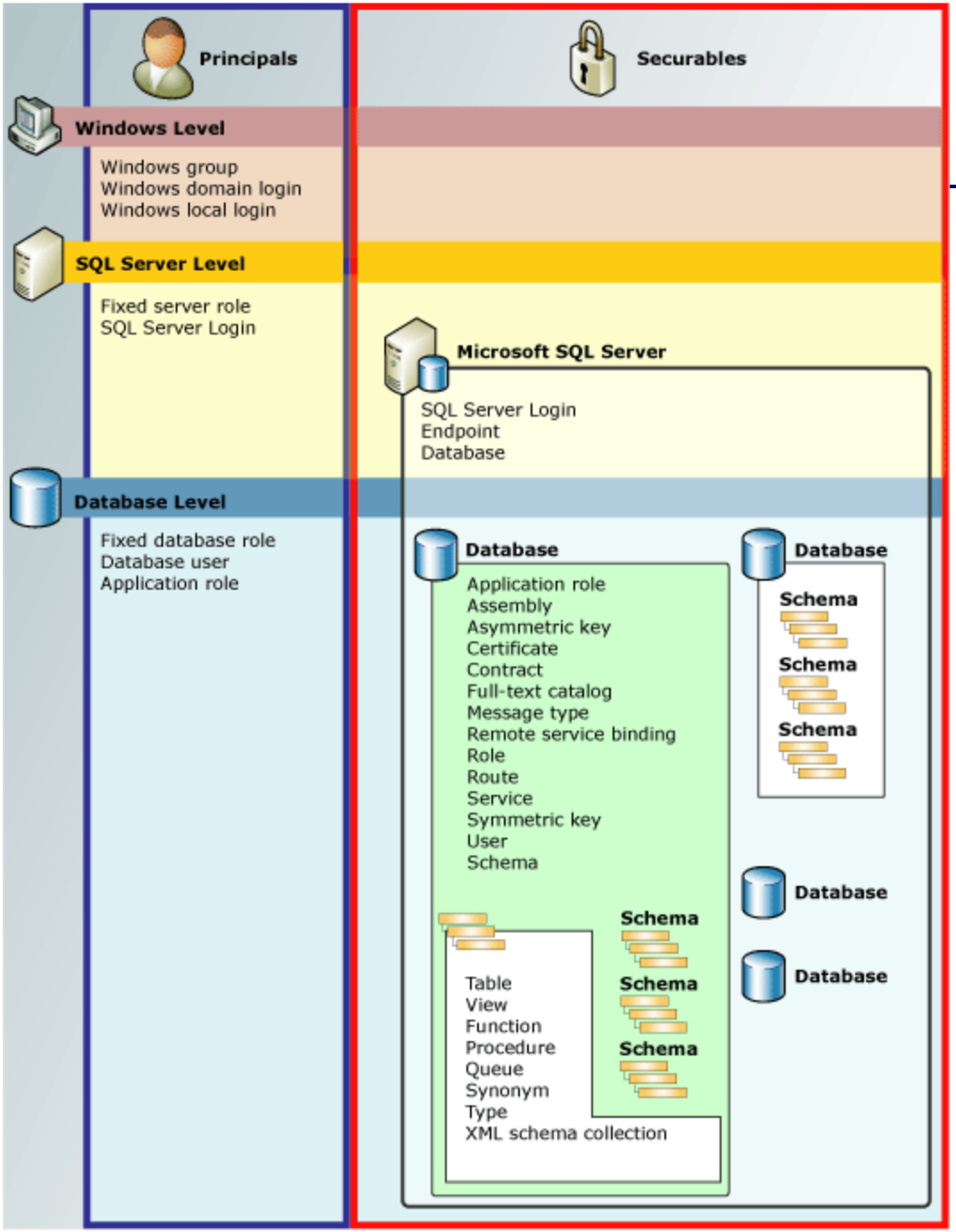
- Hashing the file to obtain the message digest

Encryption and Public Key Infrastructure



Steps in digital signature verification

Security in new DBMSs



Principals:

- Entities that can request SQL Server resources,
- Arranged in a hierarchy
- Each has a security identifier.

Securables: The resources to which the SQL Server Database Engine authorization system regulates access

Endpoint: Each database mirroring server instance requires a unique listener port that is assigned to the database mirroring endpoint of the instance.

Source: MS SQL Server Books Online

Security in new DBMSs

- Oracle 19c
  - Oracle Database can authenticate users attempting to connect to a database by using information stored in that database itself.
  - To configure Oracle Database to use database authentication, each user must be created with an associated password.
  - Oracle Database generates a one-way hash of the user's password and stores it for use when verifying the provided login password.
    - the salted SHA-1 hashing algorithm
    - the salted PKBDF2 SHA-2 SHA-512 hashing algorithm
  - Oracle Database records the password versions in the DBA_USERS data dictionary view.

Source: Oracle Database Security Guide, 19c, E96299-10, 2020.

Security in new DBMSs

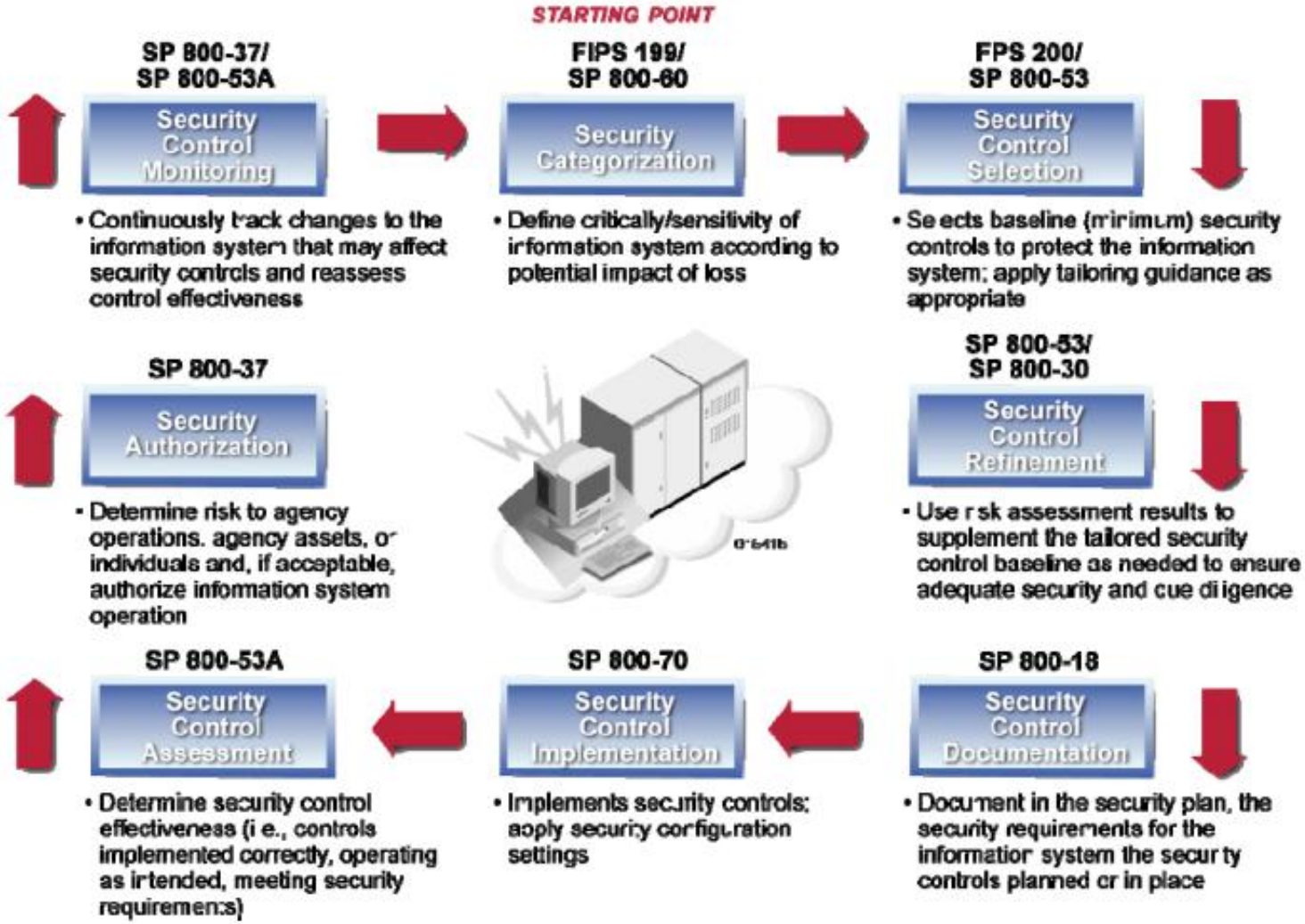
- MySQL 8.0
  - SHA-256 Pluggable Authentication: two authentication plugins that implement SHA-256 hashing for user account passwords:
    - sha256_password: Implements basic SHA-256 authentication.
      - In the name sha256_password, "sha256" refers to the 256-bit digest length the plugin uses for encryption.
    - caching_sha2_password: Implements SHA-256 authentication (like sha256_password), but uses caching on the server side for better performance and has additional features for wider applicability.
      - The default authentication plugin rather than mysql_native_password.
      - In the name caching_sha2_password, "sha2" refers more generally to the SHA-2 class of encryption algorithms, of which 256-bit encryption is one instance. The latter name choice leaves room for future expansion of possible digest lengths without changing the plugin name.

Source: MySQL 8.0 Reference Manual, 2020.

Security in new DBMSs

- NoSQL DBMSs
  - HBase (The Hadoop Database, hbase.apache.org)
    - RBAC: controls which users or groups can read and write to a given HBase resource or execute a coprocessor endpoint, using the familiar paradigm of roles.
    - GRANT/ REVOKE
    - …
  - MongoDB (mongodb.org)
    - RBAC: enabled or disabled (default) to govern each user's access to database resources and operations
    - …

Risk Management in the System Security Life Cycle



Source: P. Bowen, J. Hash, M. Wilson. Information Security Handbook: A Guide for Managers, NIST Special Publication 800-10, 2006.

Summary

- Database security refers to protection from malicious access:
  - Unauthorized reading of data (theft of information),
  - Unauthorized modification of data,
  - Unauthorized destruction of data.
- To protect the database, security measures must be taken into account at several levels:
  - Database system: authorization to data access
  - Operating system: authorization to database system access
  - Network: secured remote access via terminals and networks
  - Physical: physically secured sites with computer systems
  - Human: Users must be authorized carefully to reduce the chance of any user giving access to an intruder.

Summary

- Integrity constraints ensure that changes made to the database by authorized users do not result in a loss of data consistency.
- The data stored in the database need to be protected from unauthorized access, malicious destruction or alteration, and accidental introduction of inconsistency.
- Threats to databases
  - Loss of integrity
  - Loss of availability
  - Loss of confidentiality

Summary

- Four main control measures are used to provide security of data in databases:
  - Access control
  - Inference control
  - Flow control
  - Data encryption
- In a multiuser database system, the DBMS must provide techniques to enable certain users or user groups to access selected portions of a database without gaining access to the rest of the database.
  - Discretionary security mechanisms
    - Discretionary access control based on granting and revoking privileges
  - Mandatory security mechanisms
    - Mandatory access control and role-based access control for multilevel security

Summary

- The database administrator (DBA) is the central authority for managing a database system.
  - The DBA's responsibilities include granting privileges to users who need to use the system and classifying users and data in accordance with the policy of the organization.
  - The DBA has a DBA account in the DBMS, sometimes called a system or superuser account, which provides powerful capabilities that are not made available to regular database accounts and users.
  - DBA-privileged commands include commands for granting and revoking privileges to individual accounts, users, or user groups and for performing the following types of actions:
    - Account creation
    - Privilege granting
    - Privilege revocation
    - Security level assignment

Chapter 7: Database Security

Review

- 1. Discuss what is meant by each of the following terms: database authorization, access control, data encryption, privileged (system) account, database audit, audit trail, authentication.
- 2. What is the purpose of having separate categories for index authorization and resource authorization?
- 3. What are advantages of encrypting data stored in the database?
- 4. Perhaps the most important data items in any database system are the passwords that control access to the database. Suggest a scheme for the secure storage of passwords. Be sure that your scheme allows the system to test passwords supplied by users who are attempting to login to the system.
- 5. Which account is designated as the owner of a relation? What privileges does the owner of a relation have?
- 6. Discuss the types of privileges at the account level and those at the relation level.

Review

- 7. Views are used to simplify access to the database by users who need to see only part of the database. Views are also used as a security mechanism. Do these two purposes for views ever conflict? Explain your answer.
- 8. How is the view mechanism particularly used as an authorization mechanism?
- 9. What is meant by granting a privilege? What is meant by revoking a privilege?
- 10. Discuss the system of propagation of privileges and the restraints imposed by horizontal and vertical propagation limits.

Review

- 11. What is the difference between discretionary and mandatory access control?
- 12. What are the typical security classifications? Discuss the simple security property and the *-property, and explain the justification behind these rules for enforcing multilevel security.
- 13. Describe the multilevel relational data model. Define the following terms: apparent key, polyinstantiation, filtering.
- 14. What are the relative merits of using DAC or MAC?
- 15. What is role-based access control? In what ways is it superior to DAC and MAC?
- 16. What are the two types of mutual exclusion in role-based access control?
- 17. What is meant by row-level access control?

Review

- 18. What are the different types of SQL injection attacks?
- 19. What risks are associated with SQL injection attacks?
- 20. What preventive measures are possible against SQL injection attacks?
- 21. What is a statistical database? Discuss the problem of statistical database security.
- 22. How is privacy related to statistical database security? What measures can be taken to ensure some degree of privacy in statistical databases?
- 23. What is flow control as a security measure? What types of flow control exist?
- 24. What are covert channels? Give examples of covert channels.
- 25. What is the goal of encryption? What process is involved in encrypting data and then recovering it at the other end?
- 26. What is the public key infrastructure scheme? How does it provide security?

Review

- 27. Consider the COMPANY database. Suppose that all the relations were created by (and hence are owned by) user X, who wants to grant the following privileges to user accounts A, B, C, D, and E:
  - a. Account A can retrieve or modify any relation except DEPENDENT and can grant any of these privileges to other users.
  - b. Account B can retrieve all the attributes of EMPLOYEE and DEPARTMENT except for Salary, Mgr_ssn, and Mgr_start_date.
  - c. Account C can retrieve or modify WORKS_ON but can only retrieve the Fname, Minit, Lname, and Ssn attributes of EMPLOYEE and the Pname and Pnumber attributes of PROJECT.
  - d. Account D can retrieve any attribute of EMPLOYEE or DEPENDENT and can modify DEPENDENT.
  - e. Account E can retrieve any attribute of EMPLOYEE but only for EMPLOYEE tuples that have Dno = 3.

Write SQL statements to grant these privileges. Use views where appropriate.

Review

- 28. Suppose that privilege (a) of Review 27 is to be given with GRANT OPTION but only so that account A can grant it to at most five accounts, and each of these accounts can propagate the privilege to other accounts but without the GRANT OPTION privilege. What would the horizontal and vertical propagation limits be in this case?
- 29. Consider the relation EMPLOYEE below. How would it appear to a user with classification U? Suppose that a classification U user tries to update the salary of 'Smith' to $50,000; what would be the result of this action?

EMPLOYEE

Name	Salary	JobPerformance	TC
Smith U	40000 C	Fair S	S
Smith U	40000 C	Excellent C	C
Brown C	80000 S	Good C	S

A multilevel relation to illustrate multilevel security. Assume that the Name attribute is the apparent key, and consider the query: SELECT * FROM EMPLOYEE.

Polyinstantiation of the Smith tuple.