DATABASE SECURITY

Content

1.1 Definition

1.2 Scope of database security: why take threats seriously?
- 1.2.1 Situations related to database security
- 1.2.2 Threats

1.3 Countermeasures: computer-based controls (BRAVE)
- 1.3.1 **B**ackup and recovery
- 1.3.2 **R**AID technology
- 1.3.3 **A**uthorization
- 1.3.4 **V**iews
- (1.3.5 **E**ncryption – will be covered in next lecture)

1.4 Summary

Database security

Protection of the data against accidental or intentional loss, destruction, or misuse

- Includes:
  - hardware
  - software
  - people
  - data

Security Overview

- There are four key issues in the security of databases just as with all security systems
  - Availability
  - Authenticity
  - Integrity
  - Confidentiality

Availability

- Data needs to be available at all necessary times
- Data needs to be available to only the appropriate users
- Need to be able to track who has access to and who has accessed what data

Authenticity

- Need to ensure that the data has been edited by an authorized source
- Need to confirm that users accessing the system are who they say they are
- Need to verify that all report requests are from authorized users
- Need to verify that any outbound data is going to the expected receiver

Integrity

- Need to verify that any external data has the correct formatting and other metadata
- Need to verify that all input data is accurate and verifiable
- Need to ensure that data is following the correct work flow rules for your institution/corporation
- Need to be able to report on all data changes and who authored them to ensure compliance with corporate rules and privacy

Confidentiality

- Need to ensure that confidential data is only available to correct people
- Need to ensure that entire database is security from external and internal system breaches
- Need to provide for reporting on who has accessed what data and what they have done with it
- Mission critical and Legal sensitive data must be highly security at the potential risk of lost business and litigation

Threats to database security

- **Threat:**
  - any situation or event (intentional or accidental) that may adversely affect a system and thus the organization.
- **Tangible threat:**
  - loss of hardware, software or data.
- **Intangible threat:**
  - loss of credibility or client confidence.

Threats and Damage

Extent of damage depends on the existence of

- countermeasures, and
- contingency plans.

Example: A hard disk crash; all processing must stop until the problem is solved. Recovery time depends on factors such as:

- when the last back-ups were taken, and
- how easy it is to restore the system.

Organization needs to:

- identify types of threats and their significance.
- develop appropriate, cost-effective plans and countermeasures.

Threats

Hardware
Fire/flood/bombs
Data corruption due to power loss or surge
Failure of security mechanisms giving greater access
Theft of equipment
Physical damage to equipment
Electronic interference and radiation

DBMS and Application Software
Failure of security mechanism giving greater access
Program alteration
Theft of programs

Communication networks
Wire tapping
Breaking or disconnection of cables
Electronic interference and radiation

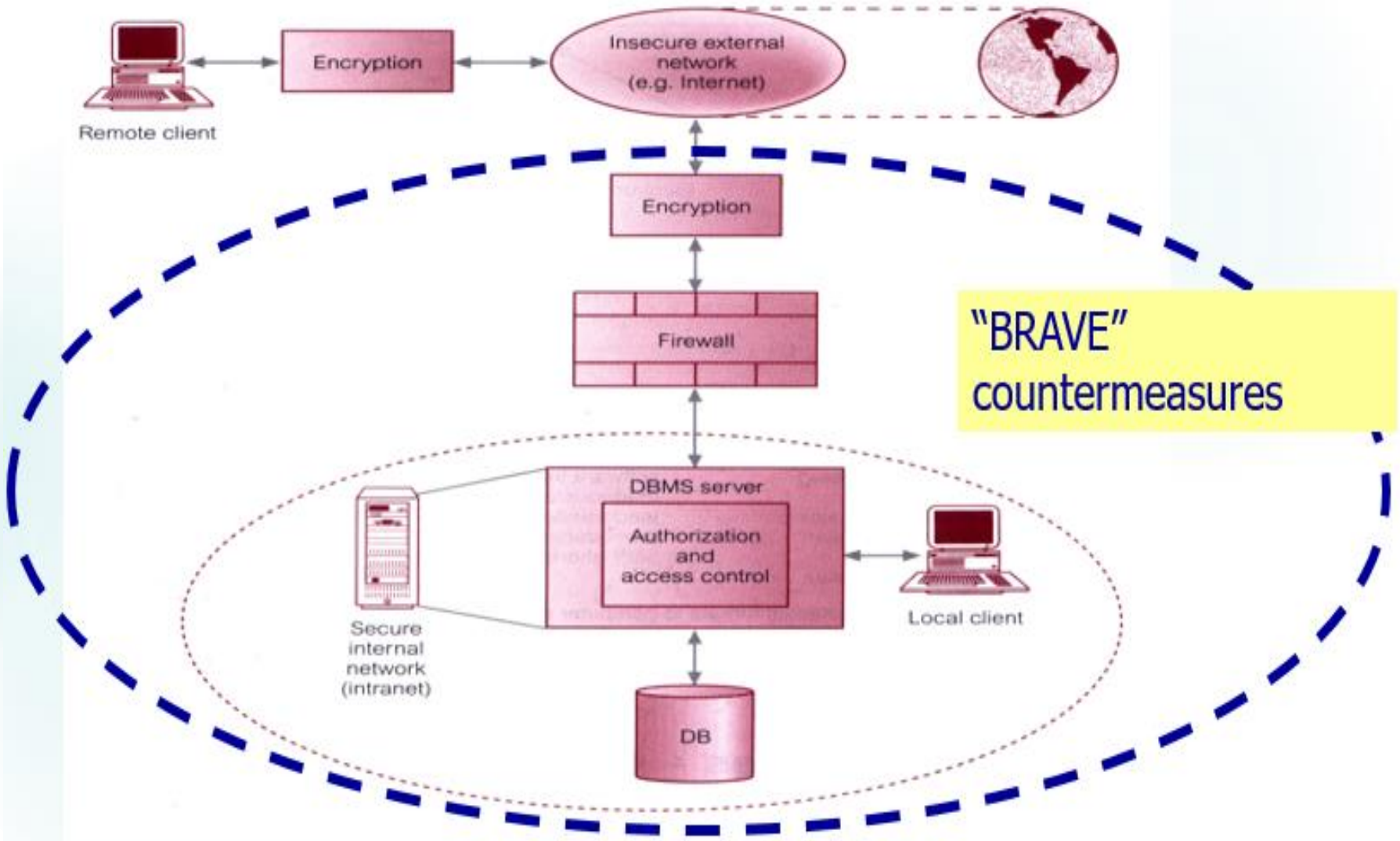
Database
Unauthorized amendment or copying of data
Theft of data
Data corruption due to power loss or surge

Users
Using another person's means of access
Viewing and disclosing unauthorized data
Inadequate Staff training
Illegal entry by hacker
Blackmail
Introduction of viruses

Programers/Operators
Creating trapdoors
Program alteration (such as creating software that is insecure)
Inadequate staff training
Inadequate security policies and procedures
Staff shortages or strikes

Data/Database Administrator
Inadequate security policies and procedures

Countermeasures: computer-based controls



Countermeasures: computer-based controls

"**BRAVE"**

- B ackup and recovery
- R AID technology
- A uthorization
- V iews
- E ncryption

Types of countermeasures range from:

- *Physical controls to*
- *Administrative procedures*

NB: despite the wide range of controls, generally a DBMS is only as secure as its operating system!

Backup and recovery

- **Backup:** the process of periodically (at frequent and regular intervals) taking a copy of the database and log files (and possibly programs) on to offline storage media kept in a secure location, suitable for rapid recovery from a fault.
- **Journaling:** the process of keeping and maintaining online a log file (journal) containing all the changes made to the database to enable recovery to be undertaken effectively in the event of failure.
- **Archiving:** the process of periodically taking a copy of the database and possibly programs on to offline secondary storage media, suitable for infrequent reference.
- DBMSs provide backup facilities to assist with recovery of the database in the event of failure.

Backup and recovery

- In the event of a failure, the database can be recovered to its last known consistent state using either:
  1. The log file created by the journaling facility, or
  2. The database backup file, or
  3. Both.

- Without a log file the database would have to be recovered solely from the database backup. This means that any changes made after the last backup to the database will be lost.

- Details of how to restore the database from the log files are given in II.II Discussed in Transaction Management Topic.

Integrity

- *Integrity constraints contribute to maintaining a secure database by preventing the data from becoming invalid and giving misleading or incorrect results.*

- *Integrity constraints were discussed in the prerequisite topics.*

RAID Technology

- DBMS hardware needs to be *fault tolerant:*
  - Disk drives, disk controllers, CPU, power supplies and cooling fans.
  - Disk drives are the most vulnerable.

- **RAID** (Redundant Array of Independent Disks)
  - several independent unreliable disks "look like" one big reliable disk
  - organized to increase performance and improve reliability.

- Performance is increased through *data striping*:
  - Data is segmented into equal-size partitions (the *striping unit*).
  - These are spread across multiple disks
  - Striping improves overall I/O performance via parallelism.

RAID Technology

- Reliability is increased through storing redundant information across disks using a *parity* or *error-correcting* scheme, such as **Reed-Solomon codes** (Pless, 1989).

1. Parity schemes:
   - each byte may have a parity bit associated with it that records whether the number of bits set to 1 is even or odd.
   - If the number of bits get corrupted, the new parity bit will not match the stored parity. Similarly, if the stored parity bit gets corrupted.

2. Error-correcting schemes
   - store two or more additional bits, and can reconstruct the original data if a single bit gets corrupted. These schemes can be used through striping bytes across disks.

RAID Technology

- **RAID 0 –Non redundant.** Maintains no redundant data. Has the best write performance since updates do not have to be replicated. Data striping performed at level of blocks.
- **RAID 1 -Mirrored.** Maintains two identical copies of the data across different disks (*mirrors*). Writes not performed simultaneously to maintain consistency in the event of disk failure. Most expensive and slowest storage solution, but very safe.
- **RAID 0+1 –Non redundant and Mirrored.** combines striping and mirroring.
- **RAID 2 -Memory-Style Error-Correcting Codes.** The striping unit is a single bit, redundancy scheme used are hamming codes.
- **RAID 3 -Bit-Interleaved Parity.** Redundancy provided by storing parity information on a single disk. This can be used to recover the data on the other disks should they fail. Parity disk can become a bottleneck though the overall level uses less storage space than RAID 1.
- **RAID 4 -Block-Interleaved Parity.** Striping unit is a disk block. A parity block is maintained on a separate disk for corresponding blocks from a number of other disks. If one of the disks fails, the parity block is used with the corresponding blocks from other disks to restore the blocks on the failed disk.
- **RAID 5 -Block-Interleaved Distributed Parity.** Parity data for redundancy is used similarly to RAID 3. The parity data is striped across all disks. Alleviating the bottleneck.
- **RAID 6 -P+Q Redundancy.** Similar to RAID 5. Additional redundant data is maintained to protect against multiple disk failures. Error-correcting codes are used instead of parity.

Authorization

Authorization: the granting of a right or privilege that enables a subject to have legitimate access to a system or a systems object.

Authorization controls (*access controls*) are built into the software. They govern:

- what systems or objects a specified user can access (e.g. database tables, views, procedures, triggers) and
- what the user may do with them.

Authorization involves *authentication* of the subjects (potential user or program) requesting access to objects.

Authentication: a mechanism that determines whether a user is who he or she claims to be. A *system administrator* is usually responsible for allowing users access by creating user accounts. Each user is given a unique *identifier* The identifier has a *password* associated with it, chosen by the user and known to the operating system.

Authorization

Procedure allows access to the computer system but not necessarily to the DBMS or other applications. A separate but similar procedure may be associated with gaining access to these further applications.

The *database administrator* (DBA) is responsible for authorizing users.

The DBA sets up individual user accounts and passwords using the DBMS itself.

Some DBMSs maintain a list of user identifiers and password *distinct* from those in the operating systems list.

Others have entries which are validated against the operating systems list based on the current user's identifier. This prevents a user logging on to the DBMS with a different name to that used when logging on to the operating system.

Authorization Privileges

Certain *Privileges* may be associated with Permission to use a DBMS. For example, the right to access or create certain database objects or relations or to run certain DBMS utilities.

Privileges are granted specific to the requirements of the users job. Excessive granting of privileges can compromise security.

Closed systems: though authorized to use the DBMS, authorization is needed to access specific objects, granted by either the DBA or the owners of the specific objects.

Open systems: users are allowed complete access to all objects within the database. Privileges will have to be explicitly *removed* to control access.

Authorization Ownership & Privileges

- The DBMS owns some objects, usually in the form of a specific *superuser* such as the DBA.

- Ownership gives the owner all appropriate privileges on the object owned.

- The creator of an object owns the object, and can assign the appropriate privileges.

For example: though the owner owns a view they may be only authorized to query it. (owners set the privileges allowed, and are also subjected to them). Privileges can be passed on to other authorized users.

- In SQL, when a user passes on a privilege, *they can indicate* whether the recipient can pass it on.

Authorization Ownership & Privileges

•Different types of identifier may have different associated priorities. For example a DBMS may permit both individual and group user identifiers, and the *individual* user may have higher priority than the *group*.

Privileges (Select, Update, Insert, Delete or All) usually have a binary value associated with them (differs from one system to another)

e.g. Select = 0100, Update = 0010, All = 1111. Summing them, for a particular object, indicates the privileges for that user type.

•The different privileges for the different types of user specific to each database object are stored in an *access control matrix*.

Authorization Ownership & Privileges

Example:

User group identifiers

User Identifier	Type	Group	Member Identifier
SG37	User	Sales	SG37
SG14	User	Sales	SG14
SG5	User		
Sales	Group		

Access Control Matrix

User Identifier	PropertyNo	type	price	ownerNo	staffNo	branchNo	Query Row Limit
Sales	0001	0001	0001	0000	0000	0000	15
SG37	0101	0101	0111	0101	0111	0000	100
SG5	1111	1111	1111	1111	1111	1111	none

Views

- **View:** a view is the dynamic result of the interaction of one or more relational operations with the base relations, which produce another relation. It is a *virtual relation* in that it does not exist in the database but is produced upon request by a given user at a particular time.

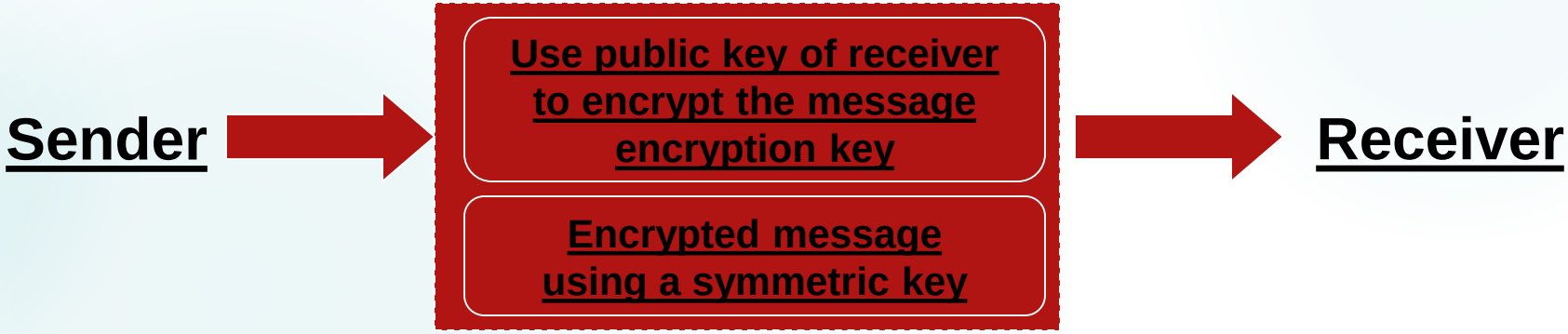
- The view is a powerful and flexible security mechanism as it hides parts of the databases from certain users.

Encryption

- The encoding of the data by a special algorithm that renders the data unreadable by any program without the decryption key
- Symmetric cryptography: sender and receiver use the same key
- Asymmetric cryptography: encryption & decryption keys

Encryption

- Encryption key: public key
- Decryption key: private key
- Asymmetric techniques: more secure but expensive in terms of computational costs



Encryption & PKI (Public Key Infrastructure)

- How does PKI work?