CHAPTER 16

DATABASE SECURITY

CHAPTER OBJECTIVES

- Establish the goals and objectives of a database security system
- Examine potential security problems and review solution options
- Understand the principles and applications of access control
- Distinguish between authentication and authorization and learn how these are performed in a database system
- Study encryption and its application to databases
- Review special security considerations such as security of statistical databases and the use of views for database security

Think of a storage place for a manufacturing organization where it keeps all the important raw materials required for making its products. The organization uses these materials to run its production. Various types of materials form parts of different products. Some types of materials are more critical and sensitive to the production process. The organization cannot perform its day-to-day operations without access to the materials. Will the organization allow anyone to walk into the storage facility off the street and have access to the materials? Unless the place is properly secured, unauthorized persons can enter and steal or destroy the materials. Not even all authorized persons may be permitted access to all parts of the storage area. Sensitive and critical materials must be off-limits to most people.

For a modern organization, its database system is even more significant than many other types of assets. Many organizations such as financial institutions and

Database Design and Development: An Essential Guide for IT Professionals by Paulraj Ponniah
ISBN 0-471-21877-4

travel companies cannot survive even a single day without their database systems. Any type of destruction of or unauthorized access to the database system has serious impact. Obviously, an organization must ensure that its database system is adequately guarded against accidental breaches of security or theft, misuse, and destruction through malicious intent.

Every organization must protect its database system from intentional and unintentional threats. To do so, it must employ both computer-based and other types of controls. The DBMS must include a proper security system to protect the database from unauthorized access.

SECURITY ISSUES

What are we trying to protect by ensuring database security? What levels of information need to be safeguarded and how? What are the types of problems and threats that deserve special attention? Can we distinguish between threats from outside and internal threats? Do these require different types of protection mechanisms? What are the solution options? How is protection of privacy related to database security?

Let us address these broad questions before getting into specific access control techniques. Many organizations are opening up their database systems for access over the Internet. This openness results in great advantages but, at the same time, makes the database system vulnerable to threats from a much wider area. Web security demands special attention.

Goals and Objectives

Specifically, what are we trying to protect? How are we planning to protect? These questions form the basis for discussions on database security. Let us consider the primary goals and objectives. Figure 16-1 provides an overview of the security system for a database.

Note the following three broad goals of database security highlighted in the figure.

- Denial of access to the database by unauthorized users
- Guarantee of access to all authorized users
- Protection of privacy of data

In a broad sense, you understand database security and what protection means. However, let us get into specific objectives. What are the particular objectives to deal with individual types of threats? Here is a list of specific objectives of a security system:

Shield from destruction. Shield the database from fire or any other such disaster.

Safeguard from theft. Safeguard the database from malicious schemes of competitors or profiteers to steal the data content.

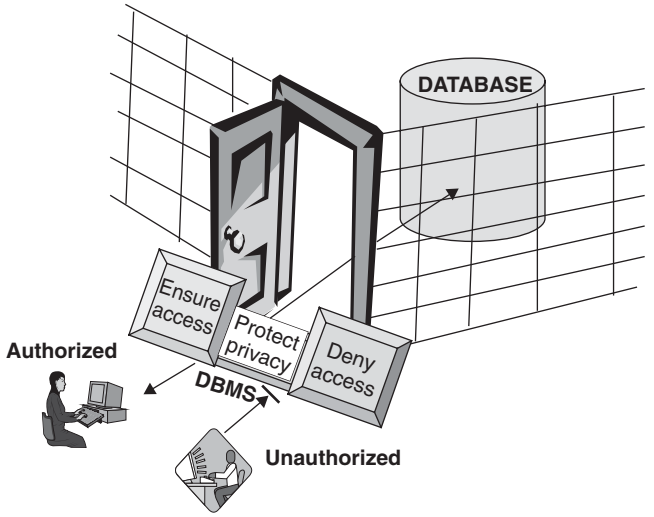

Figure 16-1 Database security system.

Defense from vandalism. Defend the database from the attempts of ingenious, disgruntled professionals intending to tamper with and vandalize the database.

Provide safety from fraud. Keep the database safe from persons with intentions to commit fraud or to misuse its contents .

Shelter of privacy. Shelter the privacy of individuals and institutions about whom data reside in the database.

Identification of users. Be able to positively identify authorized users.

Authorization of users. Guarantee access to authorized users.

Scope of authorization. Be able to authorize individual users for specific portions of the database as needed.

Levels of authorization. Provide individual users with particular authorization levels to read, update, add, or delete data.

Monitoring of usage. Be able to monitor access by authorized users to keep audit trails for tracing actions.

Security Problems

Many aspects of security problems require attention in a database environment. Legal, social, and ethical aspects are involved. Does the person requesting for particular information have a legal right to that piece of information? Also, there are policy questions about who decides on what types of access authorizations must be granted

to whom and when. Operational and administrative aspects need to be considered. How do you allocate passwords, maintain them, and preserve confidentiality?

What about physical controls to prevent problems? Should workstations and servers be guarded with physical lock-and-key schemes? Are hardware controls available in your environment to be used for database security? Are there security schemes in the operating system itself? Finally, what are the security provisions in your DBMS, and to what extent can your environment take advantage of these provisions?

To come up with solution options, first it will be worthwhile to classify the types of security problems likely to be encountered. When you are able to classify the threats, you will be able to find solutions to each type of problem. Broadly, we may classify the types of security exposure in a database environment as follows:

Natural disasters. Fire, floods, and other such catastrophes.

Human carelessness. Unintended damage caused by authorized users, especially while running jobs in batch.

Malicious damage. Sabotage, vandalism, actions of malicious programmers, technical support staff, and persons performing database administration functions.

Crime. Theft, embezzlement, industrial espionage, and employees selling a company's secrets and data for mailing lists.

Privacy invasion. Casual curiosity, data lookup by competitors, obtaining data for political or legal reasons.

Let us put together the components of the problems of database protection and summarize the potential threats. Figure 16-2 presents a summary of threats to database security. Note each component showing the type of threat and its source.

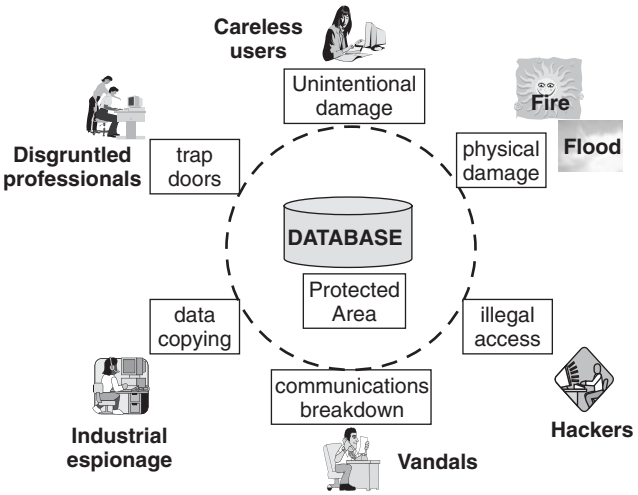

Figure 16-2 Threats to database security.

Solution Options

We have looked at the types of potential threats to a database system. Various types of sources pose different threats. How do you make provisions to protect your database system? When you consider each type of threat or problem, adopt a three-level approach to problem resolution:

- Minimize the probability of the problem happening. Establish enough protection rings to enclose the database system. Take all the necessary protective measures and institute strong deterrents.
- Diminish the damage if it happens. If an intruder manages to penetrate the outer layer of protection, make it progressively difficult to cut through the inner layers. Guard the most sensitive portions of the database with the most stringent security measures.
- Devise precise recovery schemes. If a vandal manages to destroy some parts of the database, have a tested method to recover from the damage. If a fire destroys your database, plan to be able to restore from a copy stored off-site.

When you examine the types of threats, you will notice that most of the recovery solutions must be a combination of general control procedures and computer-based techniques. Let us explore the nature of these two types of solution methods.

General Control Procedures These are matters of broad security policy and general procedures. Although these procedures deal with the security of the database in a computer system, most of these do not involve the direct use of computers. Many of these relate to planning and policy-making. Some are physical controls, and a few others involve outside agencies. The following is a list of such security measures.

Physical controls. Include physical access to buildings, monitoring of visitors at entrances and exits, and guarding of workstations and servers.

Human controls. Safeguard against threats from information system professionals and specialists by proper security clearance to work on sensitive data.

Control of equipment. Includes secure placement of equipment such as laptops loaded with sensitive data and printers that are designated to print critical data.

Security through outside agencies. Refers to third-party storage areas to keep backup copies of database and outside installations that can be used for disaster recovery.

Contingency Plans. Intended to be adopted in case of fire or bomb alerts. Plans must include designation of responsibilities and procedures for recovery.

Security Policy. An essential element of the security system to address the scope of the security schemes, the duties and responsibilities of employees, the procedures

to be followed, and disciplinary action in the event of noncompliance with policy and procedures.

Computer-Based Techniques Now let us turn our attention to the types of countermeasures that are executed through the use of the computer system including the DBMS. Here is a list of the major techniques:

Authorization of users. Includes authentication of authorized users and granting of access privileges to them.

Tailoring authorization through views. Defining user views to have the ability to authorize users for specific portions of the database.

Backup and recovery. Creation of backup copies of the database at regular intervals and also testing and implementing recovery procedures.

Protection of sensitive data. Use of encryption technology to protect sensitive data.

All DBMSs have security systems to guarantee database access to authorized users. Commonly, these security mechanisms are referred to as discretionary and mandatory security mechanisms. Let us define the scope of this division:

Discretionary security mechanisms. Used for granting and revoking data access privileges to users for accessing specific parts of a database in any of the access modes of read, update, add, and delete.

Mandatory security mechanisms. Used for establishing security at multiple levels by classifying users into distinct groups and grouping data into distinct segments and, thereafter, assigning access privileges for particular user groups to data segments.

From our discussions so far, you must have concluded that database security is critical but also difficult. You must look toward enforcing database security at different levels. Security mechanisms must exist at several layers such as within the database system itself, at the level of the operating system, the network, the application, the hardware, and so on. Figure 16-3 clearly illustrates the layers of control for database security.

Privacy Issues

Businesses and government agencies collect and store large volumes of information about customers, suppliers, distributors, and employees. Data privacy concerns those kinds of information that relate to individuals and external organizations that are part of the company's database. Who owns this information—the company that has the database or the individuals and organizations to whom the information relates? Who can access this information? Can this information be sold to others? What are the regulations?

Data privacy fits into data security in an unorthodox manner. Data security is generally thought of as the protection of a company's data from unauthorized access. Who authorizes access, and who decides on how and to whom access must

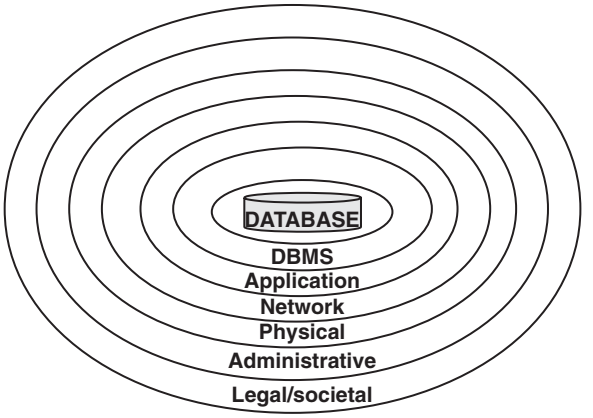

Figure 16-3 Database security: layers of control.

be granted? Of course, the company does this because it is deemed that the company owns the data in the database. In the same way, data privacy may be thought of as protecting information about employees, customers, suppliers, and distributors from unauthorized access. Who decides on this authorization? Naturally, the owners must make the decision. Who are the owners—the company or those about whom information is collected and stored?

Privacy issues are becoming more and more sensitive in North America, as they have been in Europe for some time. Legislation about privacy and confidentiality of information varies from region to region. Some basic rights are available to those about whom data is retained in corporate databases. Individuals and institutions may inquire about what information about them is stored and may demand to correct any information about them. Privacy concerns escalate with the widespread use of the Internet. Although formal regulations may not be adequate, organizations are ethically obliged to prevent misuse of the information they collect about individuals and third-party institutions.

Web Security

While discussing database security, it is important to mention security mechanisms as they relate to the DBMS and the Web. We will discuss these security options in Chapter 19, which is dedicated to the topic of the database and the Web. Security options include firewalls, proxy servers, digital signatures, and so on.

ACCESS CONTROL

Essentially, database security rests on controlling access to the database system. Controlling physical access forms one part of database security. The other major part consists of controlling access through the DBMS.

Let us consider two primary dimensions of access control. One dimension of access control deals with levels of data access. A single user or a category of users

may be granted access privileges to database objects at various levels of detail. Another dimension of access control refers to the modes or types of access granted to a single user or to a category of users. How do you grant access privileges to a single user or user category? This leads to the two basic approaches to access control.

As noted above, the DBMS provides two basic approaches to access control: discretionary control and mandatory control. Discretionary access control refers to the granting of privileges or rights to individual users. Although discretionary access control is fairly effective, it is possible for an unauthorized user to gain privileges through an unsuspecting authorized user. Mandatory access control is more effective in overcoming the defects of discretionary access control.

We will first discuss how data access control pertains to levels of database objects and access types or modes. Data levels and access types form a grid, and access privileges may be granted at the intersections of data levels and access types. Our discussion will continue on the mechanisms for granting access privileges under the discretionary or mandatory access control approaches. In this section, you will also study the two important topics of authentication and authorization of users.

Levels and Types of Data Access

Let us grasp the significance of data levels for the purpose of granting access privileges. Consider the following database relation containing data about a worker in a construction company:

WORKER (WorkerId, Name, Address, City, State, Zip, SuperId, WageRate)

Examine the following list of possible ways of granting of access privileges to a specific user:

- User has unlimited access privileges to the entire WORKER relation.
- User has no access privileges of any kind to any part of the WORKER relation.
- User may only read any part of WORKER relation but cannot make any changes at all.
- User may read only his or her row in the relation but cannot change any columns in that row.
- User may read only his or her row in the relation but can change only the Name and Address columns.
- User may read only the WorkerId, Name, Address, and SuperId columns of any record but can change only the Name and Address columns.
- User may read only the WorkerId and WageRate columns of any record but can modify the WageRate column only if the value is less than 5.00.
- User may read all columns of any record but can modify the WageRate only if the SuperId column value is the value of WorkerId of that user.

The above list is in no way exhaustive. Yet you can readily observe that a general method of security enforcement must possess a great range and flexibility. A flexi-

ble security system in a DBMS must be able to grant privileges at the following data levels:

The whole database
  Individual relation; all rows and all columns
    All rows but only specific columns of a relation
    All columns but only specific rows of a relation
      Specific rows and specific columns of a relation

Now let us move on to the consideration of modes or types of data access. You are familiar with access types or modes of create, read, update, and delete (sometimes indicated by the acronym CRUD). Let us expand the list of access types to include all types:

Insert or Create. Add data to a file without destroying any data.

Read. User may read and copy data from the database into the user's environment through an application program or a database query.

Update. Write updated values.

Delete. Delete and destroy specific data objects.

Move. Move data objects without the privilege of reading the contents.

Execute. Run a program or procedure with implied privileges needed for the execution.

Verify Existence. Verify whether a specific database object exists in the database.

You have noted the various access types and also the levels of data eligibility based on which access privileges may be granted. What is your observation from this discussion? What are the implications? You can easily realize the immense flexibility needed for giving access privileges. Although numerous variations are possible, most commonly access privileges are granted to single relations in the CRUD modes.

Discretionary Control

As mentioned above, in this approach, individual users are granted privileges or rights to access specific data items in one or more designated modes. On the basis of the specification of privileges, a user is given the discretion to access a data item in the read, update, insert, or delete modes. A user who created a database object automatically derives all privileges to access the object including the passing on of privileges to other users with regard to that object.

While discussing SQL data control examples in Chapter 13, we introduced the SQL commands for granting and revoking access privileges. This is how SQL supports discretionary access control. Now we will explore the fundamental concepts of discretionary access control and go over a few more examples.

Basic Levels There are two basic components or levels for granting or revoking access privileges:

Database Objects	Data item or data element, generally a base table or view
Users	A single user or a group of users identifiable with some authorization identifier

With these two components, access privileges may be granted as shown in the following general command:

```
GRANT privileges ON database object TO users
```

At the level of *users,* access privileges include the following:

CREATE privilege	To create a database schema, a table or relation, or a user view
ALTER privilege	To alter and make changes to schema such as adding or eliminating columns
DROP privilege	To delete a table or view
SELECT privilege	To retrieve data
MODIFY privilege	To add or insert, update, or delete data
REFERENCES privilege	To define foreign keys and reference columns in another table

At the level of *database objects*, the access privileges listed above apply to the following:

Base table	All data in the table or relation
View	All data defined in the virtual relation or view
Column	Data in the specified column only

Authorization Matrix We can think of the two levels of *users* and *database objects* forming a matrix for the purpose of granting access privileges. Set the users as columns and the database objects as rows. Then in the cells formed by the intersection of these columns and rows we can specify the type of privilege granted.

Figure 16-4 presents an example of a type of authorization matrix. Note how this type of presentation makes it easy to review the access privileges in a database environment.

Owner Account Each database table or relation has an owner. This user account that created the table possesses all access privileges on that table. The DBA can assign an owner to an entire schema and grant the appropriate access privileges. The owner of a database object can grant privileges to another user. This second user can then pass along the privileges to a third user and so on. The DBMS keeps track of the cycle of granting of privileges.

SUBJECT or USER	DATABASE OBJECT	PRIVILEGE	CONSTRAINT
Rogers	Department record	Modify	None
Miller	Department record	Create	DeptNo NOT EQUAL 101
Chen	Employee record	Select	None
Goldstein	Employee record	Create	None
Jenkins	Department record	Modify	Address ONLY
Gonzales	Employee record	Drop	EmployeePosition = 'Staff'

Figure 16-4 Authorization matrix.

Here is an example of a cycle of privileges passed along from Rogers, who is the owner of table EMPLOYEE:

By Rogers	GRANT ALL PRIVILEGES ON EMPLOYEE TO Miller WITH GRANT OPTION
By Miller	GRANT ALL PRIVILEGES ON EMPLOYEE TO Chen WITH GRANT OPTION
By Chen	GRANT ALL PRIVILEGES ON EMPLOYEE TO Williams WITH GRANT OPTION
By Rogers	GRANT SELECT ON EMPLOYEE TO Goldstein WITH GRANT OPTION
By Goldstein	GRANT SELECT ON EMPLOYEE TO Rodriguez WITH GRANT OPTION
By Rogers	REVOKE ALL PRIVILEGES ON EMPLOYEE FROM Miller CASCADE

Figure 16-5 illustrates this cycle of privileges with an authorization graph. Note how the privileges are passed along and how the revoking of privileges with cascade option works.

REFERENCES Option The REFERENCES privilege is not the same as the SELECT privilege. Let us take an example. Suppose Nash is the owner of the DEPARTMENT table as indicated below:

DEPARTMENT (DeptNo, DeptName, DeptLocation)

Nash can authorize Miller to create another table EMPLOYEE with a foreign key in that table to refer to the DeptNo column in the DEPARTMENT table. Nash can do this by granting Miller the REFERENCES privilege with respect to the DeptNo column. Note the EMPLOYEE table shown below:

EMPLOYEE (EmployeeNo, FirstName, LastName, Address, Phone, Employee Position, Salary, EmployeeCode, DeptNo)

Foreign Key: DeptNo REFERENCES DEPARTMENT

If Miller loses the REFERENCES privilege with respect to the DeptNo column in the DEPARTMENT table, the foreign key constraint in the EMPLOYEE

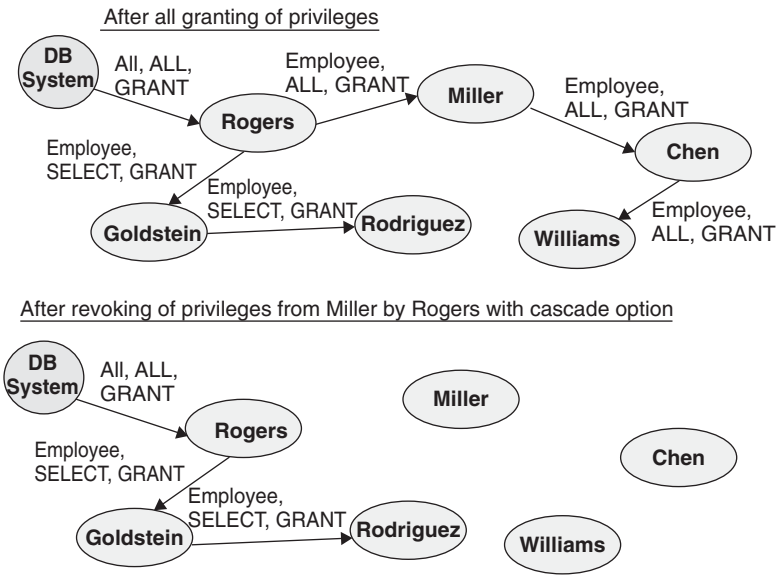

Figure 16-5 Authorization graph.

table will be dropped. The EMPLOYEE table itself, however, will not be dropped.

Now suppose Miller has the SELECT privilege on the DeptNo column of the DEPARTMENT table, not the REFERENCES privilege. In this case, Miller will not be allowed to create the EMPLOYEE table with a foreign key column referring to DeptNo in the DEPARTMENT table.

Why not grant Miller the SELECT privilege and allow him to create the EMPLOYEE table with a foreign key column referring to the DeptNo column in the DEPARTMENT table? If this is done, assume that Miller creates the table with a foreign key constraint as follows:

EMPLOYEE (EmployeeNo, FirstName, LastName, Address, Phone, Employee Position, Salary, EmployeeCode, DeptNo)

Foreign Key: DeptNo REFERENCES DEPARTMENT ON DELETE NO ACTION

With the NO ACTION option in the foreign key specification, Nash is prevented from deleting rows from the DEPARTMENT table even though he is the owner. For this reason, whenever such a restrictive privilege needs to be authorized, the more stringent privilege REFERENCES is applied. The SELECT privilege is therefore intended as permission just to read the values.

Use of Views

Earlier we had discussions on user views. A user view is like a personalized model of the database tailored for individual groups of users. If a user group, say, in the marketing department, needs to access only some columns of the DEPARTMENT

and EMPLOYEE tables, then you can satisfy their information requirements by creating a view comprising just those columns. This view hides the unnecessary parts of the database from the marketing group and shows them only those columns they require.

Views are not like tables in the sense that they do not store actual data. You know that views are just like windows into the database tables that store the data. Views are virtual tables. When a user accesses data through a view, he or she is getting the data from the base tables, but only from the columns defined in the view.

Views are intended to present to the user exactly what is needed from the database and to make the rest of the data content transparent to the user. However, views offer a flexible and simple method for granting access privileges in a personalized manner. Views are powerful security tools. When you grant access privileges to a user for a specific view, the privileges apply only to those data items defined in the views and not to the complete base tables themselves.

Let us review an example of a view and see how it may be used to grant access privileges. For a user to create a view from multiple tables, the user must have access privileges on those base tables. The view is dropped automatically if the access privileges are dropped. Note the following example granting access privilege to Miller for reading EmployeeNo, FirstName, LastName, Address, and Phone information of employees in the department where Miller works.

```
CREATE VIEW MILLER AS
   SELECT EmployeeNo, FirstName, LastName, Address, Phone
      FROM EMPLOYEE
         WHERE DeptNo =
            (SELECT DEPARTMENT.DeptNo
               WHERE DEPARTMENT.DeptNo =
                  EMPLOYEE.DeptNo AND
                  (EMPLOYEE.LastName = 'Miller' )) ;

GRANT SELECT ON MILLER TO Miller;
```

SQL Examples

In Chapter 13, we considered a few SQL examples on granting and revoking of access privileges. Now we will study a few more examples. These examples are intended to reinforce your understanding of discretionary access control. We will use the DEPARTMENT and EMPLOYEE tables shown above for our SQL examples.

DBA gives privileges to Miller to create the schema:

GRANT CREATETAB TO Miller;

Miller defines schema, beginning with create schema statement:

CREATE SCHEMA EmployeeDB AUTHORIZATION Miller; (other DDL statements follow to define DEPARTMENT and EMPLOYEE tables)

Miller gives privileges to Rodriguez for inserting data in both tables:

GRANT INSERT ON DEPARTMENT, EMPLOYEE TO Rodriguez;

Miller gives Goldstein privileges for inserting and deleting rows in both tables, allowing permission to propagate these privileges:

```
GRANT INSERT, DELETE ON DEPARTMENT, EMPLOYEE TO Goldstein
    WITH GRANT OPTION;
```

Goldstein passes on the privileges for inserting and deleting rows in the DEPARTMENT table to Rogers:

```
GRANT INSERT, DELETE ON DEPARTMENT TO Rogers;
```

Miller gives Williams the privilege to update only the salary and position columns in the EMPLOYEE table:

```
GRANT UPDATE ON EMPLOYEE (Salary, EmployeePosition) TO Williams
```

DBA gives Shady privilege to create tables:

```
GRANT CREATETAB TO Shady;
```

Shady creates table MYTABLE and gives Miller privilege to insert rows into MYTABLE:

```
GRANT INSERT ON MYTABLE TO Miller;
```

Mandatory Control

Discretionary access control provides fairly adequate protection. This has been the traditional approach in relational databases. A user either has certain access privileges or he or she does not. The discretionary access control method does not support variations based on the sensitivity of parts of the database. Although the method is sufficient in most database environments, it is not bulletproof. An ingenious professional can drill holes into the protection mechanism and gain unauthorized access.

Note the actions of user Shady indicated in the last few statements of the previous subsection. Shady has created a private table MYTABLE of which he is the owner. He has all privileges on this table. All he has to do is somehow get sensitive data into MYTABLE. Being a clever professional, Shady may temporarily alter one of Miller's programs to take data from the EMPLOYEE data and move the data into MYTABLE. For this purpose, Shady has already given privileges to Miller for inserting rows into the MYTABLE table. This scenario appears as too unlikely and contrived. Nevertheless, it makes the statement that discretionary access control has its limitations.

Mandatory access control overcomes the shortcomings of discretionary access control. In the mandatory access control approach, access privileges cannot be granted or passed on by one user to another in an uncontrolled manner. A well-defined security policy dictates which classes of data may be accessed by users at which clearance levels. The most popular method is known as the Bell–LaPadula model. Many of the commercial relational DBMSs do not currently provide for mandatory access control. However, government agencies, defense departments, financial institutions, and intelligence agencies do require security mechanisms based on the mandatory control technique.

Classes and Clearances The mandatory access control model is based on the following components:

Objects	Database objects such as tables, views, rows, and columns
Subjects	Users, programs, and modules that need access privileges
Classes	Security classes for objects
Clearances	Security clearances for subjects

Each database object is assigned a security class. Typical classes are (TS) top secret, (S) secret, (C) confidential, and (U) unclassified. The data sensitivity sequence is as follows: TS > S > C > U. Each subject is assigned clearance for a specific security class. We may represent these by the following notation:

Class (O)	Security class for an object O
Class (S)	Security clearance for a subject S

How Mandatory Control Works The model enforces two basic restrictions on all reads and writes:

1. ***Simple Security Property***
   Subject S is not permitted to have read access to an object O unless Class (S) $\geqslant$ Class (O).
2. ****-Property (Star Property)***
   Subject S is not permitted to have write access to an object O unless Class (S) $\leqslant$ Class (O).

Look at the first property, which is fairly intuitive. This property allows a subject to read an object only if the subject's clearance level is higher than or equal to that of the object. Try to understand what the second property is meant to prevent. The second property prohibits a subject from writing to an object in a security class lower than the clearance level of the subject. Otherwise, information may flow from a higher class to a lower class. Consider a user with S clearance. Without the enforcement of the star property, this user can copy an object in S class and rewrite it as a new object with U classification so that everyone will be able to see the object.

Get back to the case of Shady trying to access data from the EMPLOYEE table by tricking Miller. The mandatory access control method would spoil Shady's plan as follows:

- Classify EMPLOYEE table as S.
- Give Miller clearance for S.
- Give Shady lower clearance for C.

Shady can therefore create objects of C or lower classification. MYTABLE will be in class C or lower. Miller's program will not be allowed to copy into MYTABLE because

Class (MYTABLE) < Class (Miller), violation of star property.

SPECIAL SECURITY CONSIDERATIONS

We have covered several topics on database security so far. You know the common types of security threats. You have explored solution options. You have reviewed the two major approaches to database access control—discretionary and mandatory. Before we finish our discussion of the significant topics, we need to consider just a few more.

In our discussion on granting access privileges, we have been referring to individual users or user groups that need access privileges. Who are these users, and how do you identify them to the database system? This is an important question we need to address. Another obvious question is, Where is the DBA in all of these database security provisions, and what is the role of the DBA? Finally, we will inspect what are known as statistical databases and consider special security problems associated with these.

Authorization

The security mechanism protecting a database system is expected to prevent users from performing database operations unless they are authorized to do so. Authorization for data access implies access control. We have discussed discretionary and mandatory access control approaches. Let us now complete the discussion by touching on a few remaining topics.

Profiles To authorize a subject that may be a user, a group of users, a program, or a module, an account is assigned to the subject. Let us confine our discussion to a subject who is a user. User Samantha Jenkins is eligible to have access to the human resources database. So first, Jenkins must be assigned an account or user-identification.

The DBMS maintains a user profile for each user account. The profile for Jenkins includes all the database objects such as tables, views, rows, and columns that she is authorized to access. In the user profile, you will also find the types of access privileges such as read, update, insert, and delete granted to Jenkins.

Alternatively, the DBMS may maintain an object profile for each database object. An object profile is another way of keeping track of the authorizations. For example, in the object profile for the EMPLOYEE table, you will find all the user accounts that are authorized to access the table. Just like a user profile, an object profile also indicates the types of access privileges.

Authorization Rules The user profile or the object profile stipulates which user can access which database object and in what way. These are the authorization rules. By examining these rules, the DBMS determines whether a specific user may be permitted to perform the operations of read, update, insert, or delete on a particular database object. You have already looked at an example of an authorization matrix in Figure 16-4. This matrix tends to be exhaustive and complex in a large database environment.

Many DBMSs do not implement elaborate information matrices as presented in Figure 16-4 to enforce authorization rules. Instead, they adopt simpler versions to implement authorization rules. Authorization rules may be represented as an autho-

PRIVILEGE	Department Record	Employee Record
Read	Y	Y
Update	N	N
Insert	N	Y
Delete	N	N

Authorization Rule
Subject: Samantha Jenkins

PRIVILEGE	Samantha Jenkins (password JNK271)	Will Rogers (password WRG346)
Read	Y	Y
Update	N	N
Insert	Y	N
Delete	N	N

Authorization Rule
Object: Employee Record

Figure 16-6 Implementation of authorization rules.

rization table for either subjects or users. On the other hand, an authorization table for objects can also do the job. A DBMS may implement authorization rules through either of the two tables or both.

Figure 16-6 presents both options for implementing authorization rules. Note how you can derive the same rule authorizing Samantha Jenkins to access the EMPLOYEE table with read and insert privileges.

Enforcing Authorization Rules We have authorization rules in an authorization matrix or in the form of authorization tables for users or objects. How are the rules enforced by the DBMS? A highly protected privilege module with unconstrained access to the entire database exists to enforce the authorization rules. This is the arbiter or security enforcer module, although it might not go by those names in every DBMS. The primary function of the arbiter is to interrupt and examine every database operation, check against the authorization matrix, and either allow or deny the operation.

The arbiter module goes through a sequence of interrogations to determine the action to be taken about an operation. Figure 16-7 provides a sample interrogation list.

Suppose after going through the interrogation sequence, the arbiter has to deny a database operation. What are the possible courses of action? Naturally, the particular course of action to be adopted depends on a number of factors and the circumstances. Here are some basic options provided in DBMSs:

- If the sensitivity of the attempted violation is high, terminate the transaction and lock the workstation.
- For lesser violations, send appropriate message to user.
- Record attempted security breaches in the log file.

		ACTION	
TEST	INTERROGATION	YES	NO
1	Unconditional access to all relations in request?	GRANT	
2	Any relation in request unconditionally prohibited?	DENY	Next test
3	Unconditional access to all attributes in request?	GRANT	
4	Any attribute in request unconditionally prohibited?	DENY	Next test
5	Unconditional access to all groups of attributes in request?	GRANT	
6	Any group of attributes in request unconditionally prohibited?	DENY	Inform user

Figure 16-7 Arbiter interrogation list.

- Monitor for continued attempts of security breaches by same user for possible censure.

Authentication

Let us return to the authorization of access privileges to Samantha Jenkins. The authorization matrix contains security authorization rules for her. When she attempts to perform any database operation, the DBMS, through its arbiter module, can verify authorization rules as applicable and either allow or deny the operation. When Samantha Jenkins signs on to the database system with her user-id, in effect, she declares that she is Samantha Jenkins. All authorization she can have relates to the user known to the system as Samantha Jenkins.

Now when she signs on with her user-id and declares that she is Samantha Jenkins, how does the system know that she is really who she says she is? How can the system be sure that it is really Samantha Jenkins and not someone else signing on with her user-id? How can the system authenticate her identity? Authentication is the determination of whether the user is who he or she claims to be or declares he or she is through the user-id.

It is crucial that the authentication mechanism be effective and failsafe. Otherwise, all the effort and sophistication of the authorization rules will be an utter waste. How can you ensure proper authentication? Let us examine a few of the common techniques for authentication.

Passwords. Passwords, still the most common method, can be effective if properly administered. Passwords must be changed fairly often to deter password thefts. They must be stored in encrypted formats and be masked while being entered. Password formats need to be standardized to avoid easily detectable combinations. A database environment with highly sensitive data may require one-time-use passwords.

Personal information. The user may be prompted with questions for which the user alone would know the answers such as mother's maiden name, last four digits of social security number, first three letters of the place of birth, and so on.

Biometric verification. Verification through fingerprints, voiceprints, retina images, and so on. Smartcards recorded with such biometric data may be used.

Special procedures. Run a special authentication program and converse with the user. System sends a random number m to the user. The user performs a simple set of operations on the random number and types in the result n. System verifies n by performing the same algorithm on m. Of course, m and n will be different each time and it will be hard for a perpetrator to guess the algorithm.

Hang-up and call-back. After input of user-id, the system terminates the input and reinitiates input at the workstation normally associated with that user. If the user is there at that customary workstation and answers stored questions for the user-id, then the system allows the user to continue with the transaction.

Role of the DBA

The database administrator plays a pivotal role in security administration. Along with user representatives and senior management staff including the DA, the DBA develops a security policy appropriate for the environment. The DBA is the central authority executing the security policy by setting up user accounts and granting access privileges. The DBA has a user account that has extensive access privileges on all of the database objects.

Let us summarize the responsibilities of the DBA.

Creation of new accounts. Assign user-id and password to each user or group of users.

Creation of views. Create user views as needed for the purpose of tailoring security provisions for specific user groups.

Granting of privileges. Grant access privileges for users or user groups to perform database operations on database objects in accordance with security policy.

Revocation of privileges. Cancel access privileges originally assigned to users or user groups.

Assignments of security levels. Assign user accounts to proper security classification for mandatory access control. Designate security levels to database objects.

Maintenance of audit trail. Extend log file record to include updates with user-ids.

Statistical Databases

Statistical databases pose a great and interesting challenge in the matter of data security. Statistical databases are usually large databases intended to provide statistical information and not information about individual entities. Statistical databases may contain data about large populations of entities.

A census database contains information about the people in specific geographic areas. The database system of a large international bank holds information about the savings and checking account activities of significant strata of the population. Databases of large financial institutions contain profiles of investors. Databases used

in data warehousing and data mining may be considered as statistical databases in some significant sense.

Need for Data Access Statistical databases serve critical purposes. They store rich data content providing population statistics by age groups, income levels, household sizes, education levels, and so on. Government statisticians, market research companies, and institutions estimating economic indicators depend on statistical databases. These professionals select records from statistical databases to perform statistical and mathematical functions. They may count the number of entities in the selected sample of records from a statistical database, add up numbers, take averages, find maximum and minimum amounts, and calculate statistical variances and standard deviations.

All such professionals need access to statistical databases. However, there is one big difference between users of an operational database needing access privileges and professionals requiring access privileges to a statistical database. Users of an operational database need information to run the day-to-day business—to enter an order, to check stock of a single product, to send a single invoice. That is, these users need access privileges to individual records in the database. On the other hand, professionals using statistical databases need access privileges to access groups of records and perform mathematical and statistical calculations from the selected groups. They are not interested in single records, only in samples containing groups of records.

Security Challenge So what is the problem with granting access privileges to professionals to use a statistical database just the way you would grant privileges to use any other type of database? Here is the problem: The professionals must be able to read individual records in a select sample group for performing statistical calculations but, at the same time, must not be allowed to find out what is in a particular record.

For example, take the case of the international bank. The bank's statisticians need access to the bank's database to perform statistical calculations. For this purpose, you need to grant them access privileges to read individual records. But, at the same time, you cannot allow them to see Jane Doe's bank account balance. The challenge in the case of the bank is this: How can you grant access privileges to the statisticians without compromising the confidentiality of individual bank customers?

Perhaps one possible method is to grant access privileges to individual records because the statistician needs to read a group of records for the calculations but restrict the queries to perform only mathematical and statistical functions such as COUNT, SUM, AVG, MAX, MIN, variance and standard deviations.

Although this method appears to be adequate to preserve the confidentiality of individual customers, a clever professional can run a series of queries and narrow the intersection of the query result to one customer record. This person can infer the values in individual rows by running a series of ingenuous queries. Each query produces a result set. Even though only statistical functions are permitted, by combining the different results through a series of clever queries, information about a single entity may be determined. Figure 16-8 illustrates how, by using different predicates in queries from a bank's statistical database, the bank balance of a single

Query A:

```
SELECT COUNT (*) FROM  CUSTOMER
WHERE CustCity = 'Any City' AND CustGender = 'F' AND
        IncomeLevel = 'High"
```

Query B:

```
SELECT AVG (Balance) FROM  CUSTOMER
WHERE CustCity = 'Any City' AND CustGender = 'F' AND
        IncomeLevel = 'High"
```

If result from Query A is 1, then Query B gives the account balance for the specific customer.

If result from Query A is 2 or 3, issue additional queries by including statistical functions of MAX, MIN, SUM to get a precise range for the account balance for the specific customer.

Figure 16-8 Ingenuous queries isolating individual records.

customer Jane Doe may be determined. Assume that the infiltrator knows some basic information about Jane Doe.

Solution Options Safeguarding privacy and confidentiality in a statistical database proves to be difficult. The standard method of granting access privileges does not work. In addition to discretionary and mandatory techniques, other restrictions must be enforced on queries.

Here is a list of some solution options. None of them is completely satisfactory. Combinations of some of the options seem to be effective. Nevertheless, protection of privacy and confidentiality in statistical databases is becoming more and more essential.

Only statistical functions. Allow only statistical functions in queries.

Same sample. Reject series of queries to the same sample set of records.

Query types. Allow only those queries that contain statistical or mathematical functions.

Number of queries. Allow only a certain number of queries per user per unit time.

Query thresholds. Reject queries that produce result sets containing fewer than n records, where n is the query threshold.

Query combinations. The result set of two queries may have a number of common records referred to as the intersection of the two queries. Impose a restriction saying that no two queries may have an intersection larger than a certain threshold number.

Data pollution. Adopt data swapping. In the case of the bank database, swap balances between accounts. Even if a user manages to read a single customer's

record, the balances may have been swapped with balances in another customer's record.

Introduce noise. Deliberately introduce slight noise or inaccuracies. Randomly add records to the result set. This is likely to show erroneous individual records, but statistical samples produce approximate responses quite adequate for statistical analysis.

Log queries. Maintain a log of all queries. Maintain a history of query results and reject queries that use a high number of records identical to those used in previous queries.

ENCRYPTION

We have discussed the standard security control mechanisms in detail. You studied the discretionary access control method whereby unauthorized persons are kept away from the database and authorized users are guaranteed access through access privileges. You have also understood the mandatory access control method, which addresses some of the weaknesses of the discretionary access control scheme. Now you are confident that these two standard schemes provide adequate protection and that potential intruders cannot invade the database.

However, the assumption is that an infiltrator or intruder tries to break into the system through normal channels by procuring user-ids and passwords through illegal means. What if the intruder bypasses the system to get access to the information content of the database? What if the infiltrator steals the database by physically removing the disks or backup tapes? What if the intruder taps into the communication lines carrying data to genuine users? What if a clever infiltrator runs a program to retrieve the data by breaking the defenses of the operating system?

The normal security system breaks down in such cases. Standard security techniques fall short of expectations to protect data from assaults bypassing the system. If your database contains sensitive financial data about your customers, then you need to augment your security system with additional safeguards. In today's environment of electronic commerce on the Internet, the need for dependable security techniques is all the more essential. Encryption techniques offer added protection.

What is Encryption?

Simply stated, encryption is a method of coding data to make them unintelligible to an intruder and then decoding the data back to their original format for use by an authorized user. Some commercial DBMSs include encryption modules; a few others provide program exits for users to code their own encryption routines. Currently, encryption techniques are widely used in applications such as electronic fund transfers (EFT) and electronic commerce.

An encryption scheme needs a cryptosystem containing the following components and concepts:

- An encryption key to code data (called plaintext)
- An encryption algorithm to change plaintext into coded text (called ciphertext)

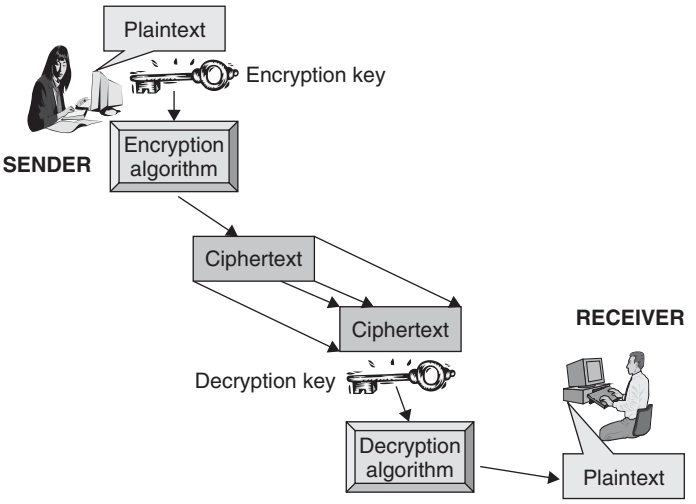

Figure 16-9 Elements of encryption.

- A decryption key to decode ciphertext
- A decryption algorithm to change ciphertext back into original plaintext

Figure 16-9 shows the elements of encryption. Note the use of the keys and where encryption and decryption take place.

The underlying idea in encryption dictates the application of an encryption algorithm to plaintext where the encryption algorithm may be accessible to the intruder. The idea includes an encryption key specified by the DBA that has to be kept secret. Also is included a decryption algorithm to do the reverse process of transforming ciphertext back into plaintext.

A good encryption technique, therefore, must have the following features:

- Fairly simple for providers of data to encrypt
- Easy for authorized users to decrypt
- Does not depend on the secrecy of the encryption algorithm
- Relies on keeping the encryption key a secret from an intruder
- Extremely difficult for an intruder to deduce the encryption key

Just to get a feel for an encryption scheme, let us consider a simple example before proceeding further into more details about encryption. First, we will use a simple substitution method. Second, we will use a simple encryption key. Let us say that the plaintext we want to encrypt is the following plaintext:

```
ADMINISTRATOR
```

Simple Substitution Use simple substitution by shifting each letter in the plaintext to three spaces to the right in the alphabetic sequence. A becomes D, D becomes G; and so on. The resulting ciphertext is as follows:

DGPLQLVWUDWRU

If the intruder sees a number of samples of the ciphertext, he or she is likely to deduce the encryption algorithm.

Use of Encryption Key This is a slight improvement over the simple substitution method. Here, let us a use a simple encryption key stipulated as "SAFE." Apply the key to each four-character segment of the plaintext as shown below:

ADMINISTRATOR
SAFESAFESAFES

The encryption algorithm to translate each character of the plaintext is as follows:

Give each character in the plaintext and the key its position number in the alphabetic scheme. The letter "a" gets 1, the letter "z" gets 26, and a blank in the plaintext gets 27. Add the position number of each letter of plaintext to the position number of the corresponding letter of the key. Then apply division modulus 27 to the sum. This calculation means dividing a number by 27 and using the remainder to applying to the algorithm. Use the number resulting from the division modulus 27 to find the letter to be substituted in the ciphertext.

Let us apply the algorithm to the first position in our plaintext.

Plaintext letter	A	Key letter	S
Position number	1	Position number	19

Sum of position numbers = (1 + 19)
Result of division modulus 27 = 20 divided by 27, remainder 20

Letter in position 20 is T. Therefore, the first letter "A" in the plaintext is substituted by "T" in the ciphertext.

After doing all the substitutions, the ciphertext reads as follows:

TESNFJYYbBYTb (b indicates a blank)

Now compare the ciphertexts produced by the two methods and note how even a simple key and fairly unsophisticated algorithm could improve the encryption scheme.

Original plaintext	ADMINISTRATOR
Ciphertext using simple substitution	DGPLQLVWUDWRU
Ciphertext using simple key	TESNFJYYbBYTb

Encryption Methods

Three basic methods are available for encryption:

Encoding. Most simple and inexpensive method. Here, for important fields, the values of are encoded. For example, instead of storing the names of bank branches, store codes to represent each name.

Substitution. Substitute, letter for letter, in the plaintext to produce the ciphertext.

Transposition. Rearrange characters in the plaintext using a specific algorithm.

Usually a combination of substitution and transposition works well. However, techniques without encryption keys do not provide adequate protection. The strength of a technique depends on the key and the algorithm used for encryption and decryption. However, with plain substitution or transposition, if an intruder reviews sufficient number of encoded texts, he or she is likely to decipher.

On the basis of the use and disposition of encryption keys, encryption techniques fall into two categories.

Symmetric Encryption This technique uses the same encryption key for both encryption and decryption. The key must be kept a secret from possible intruders. The technique relies on safe communication to exchange the key between the provider of data and an authorized user. If the key is to be really secure, you need a key as long as the message itself. Because this is not efficient, most keys are shorter. The Data Encryption Standard (DES) is an example of this technique.

Asymmetric Encryption This technique utilizes different keys for encryption and decryption. One is a public key known openly, and the other is a private key known only to the authorized user. The encryption algorithm may also be known publicly. The RSA model is an asymmetric encryption method.

Data Encryption Standard

This technique, designed and developed by IBM in 1977, was adopted by the National Bureau of Standards as the official Data Encryption Standard (DES). Since then, various industry agencies have adopted DES. The technique uses a single key for both encryption and decryption. The key has to be kept secret from potential intruders. Similarly, the encryption algorithm must not be publicly known. The algorithm consists of character substitutions and transpositions or permutations.

Figure 16-10 illustrates this single-key encryption technique.

How it Works DES divides plaintext into blocks of 64 bits each. A 64-bit key is used to encrypt each block. Although the key is 64 bits long, effectively, it is only a 56-bit key because 8 bits are used as parity bits. Even with 56 bits in a key, there are 2^{56} possible distinct keys. So there are a huge number of choices for establishing a key.

Encryption takes place in the following sequence:

- Apply an initial permutation to each block using the key.
- Subject each transformed or permuted block to a sequence of 16 complex substitution steps.
- Finally, apply another permutation algorithm, the inverse of the initial permutation.

The decryption algorithm is identical to the encryption algorithm, except that the steps are in the reverse order.

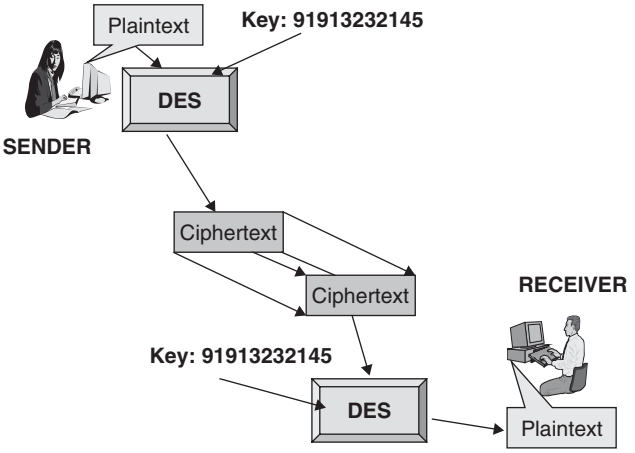

Figure 16-10 DES: single-key encryption.

Weaknesses Despite the complexity of the encryption algorithm and the sophistication of key selection, DES is not universally accepted as absolutely secure. Critics point out the following deficiencies:

- 56-bit keys are inadequate. With the powerful and special hardware available now, they are breakable. Even such expensive hardware is within the reach of organized crime and hostile governments. However, 128-bit keys are expected to be unbreakable within the foreseeable future. A better technique known as PGP (pretty good privacy) uses 128-bit keys. Another possible remedy is double application of the algorithm at each step.
- Users must be given the key for decryption. Authorized users must receive the key through secure means. It is very difficult to maintain this secrecy. This is a major weakness.

Public Key Encryption

This technique overcomes some of the problems associated with the DES technique. In DES you have to keep the encryption key a secret, and this is not an easy thing to accomplish. Public key encryption addresses this problem. The public key as well as the encryption algorithm need not be kept secret. Is this like locking the door and making the key available to any potential intruder? Let us examine the concept.

The widely used public key encryption technique was proposed by Rivest, Shamir, and Adleman. It is known by the acronym RSA. The RSA model is based on the following concepts:

- Two encryption keys are used—one public and the other private.
- Each user has a public key and a private key.
- The public keys are all published and known openly.
- The encryption algorithm is also made freely available.

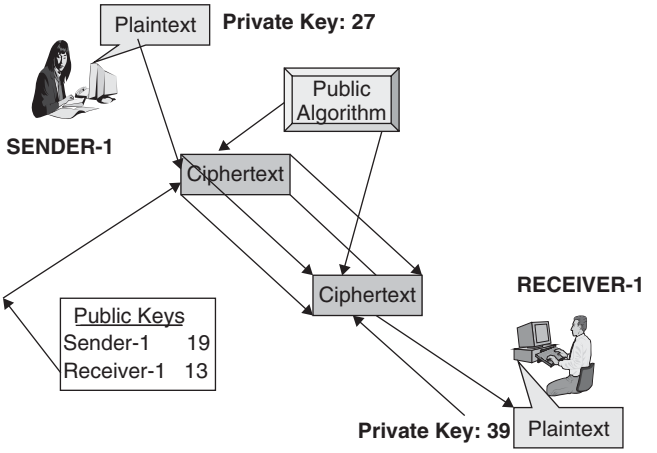

Figure 16-11 RSA: public key encryption.

- Only an individual user knows his or her private key.
- The encryption and decryption algorithms are inverses of each other.

Figure 16-11 illustrates this public key encryption technique.

How it Works Consider the following scenario:

Data provider U1 wants to share data with authorized user U2.
U2 has a public key P2 and a private key R2.
U1 uses the public key P2 of U2 to encrypt the data and transmits the data to U2.
U2 uses the private key R2 to decrypt the data.

If data have to be transmitted securely, it must be extremely difficult for any intruder to deduce the private key from the publicly known public key and the encrypting algorithm. The RSA model stands on the premise that it is virtually impossible to deduce the private keys. This premise rests on the following two facts:

- A known efficient algorithm exists for testing whether or not a given number, however large, is a prime number.
- No known efficient algorithm exists for determining the prime factors of given number, however large.

Incidentally, one of the authors of the RSA technique had estimated that testing whether a given 130-digit number is a prime or not would take about 7 minutes on a fast computer. On the other hand, if you have a large number obtained by multiplying two 63-digit prime numbers, then it would take about 40 quadrillion years on the same machine to determine the prime factors of the product!

The public key encryption technique treats data as a collection of integers. The public and private keys are reckoned for an authorized user as follows:

- Choose two large, random prime numbers n1 and n2.
- Compute the product of n1 and n2. This product is also known as the limit L, assumed to be larger than the largest integer ever needed to be encoded.
- Choose a prime number larger than both n1 and n2 as the public key P
- Choose the private key R in a special way based on n1, n2, and P

[If you are interested, R is calculated such that R * P = 1, modulo (n1–1) * (n2–1).]

The limit L and the public key P are made known publicly. Note that the private key R may be computed easily if the public key P and the prime numbers n1 and n2 are given. However, it is extremely difficult to compute the private key R if just the public key P and the limit L are known. This is because finding the prime factors of L is almost impossible if L is fairly large.

Data Exchange Example Let us consider the use of public key encryption in a banking application. Here are the assumptions:

- Online requests for fund transfers may be made to a bank called ABCD. The bank's customer known as Good places a request to transfer $1 million.
- The bank must be able to understand and acknowledge the request.
- The bank must be able to verify that the fund transfer request was in fact made by customer Good and not anyone else.
- Also, customer Good must not be able to allege that the request was made up by the bank to siphon funds from Good's account.

Figure 16-12 illustrates the use of public key encryption technique showing the banking transaction. Note how each transfer is coded and decoded.

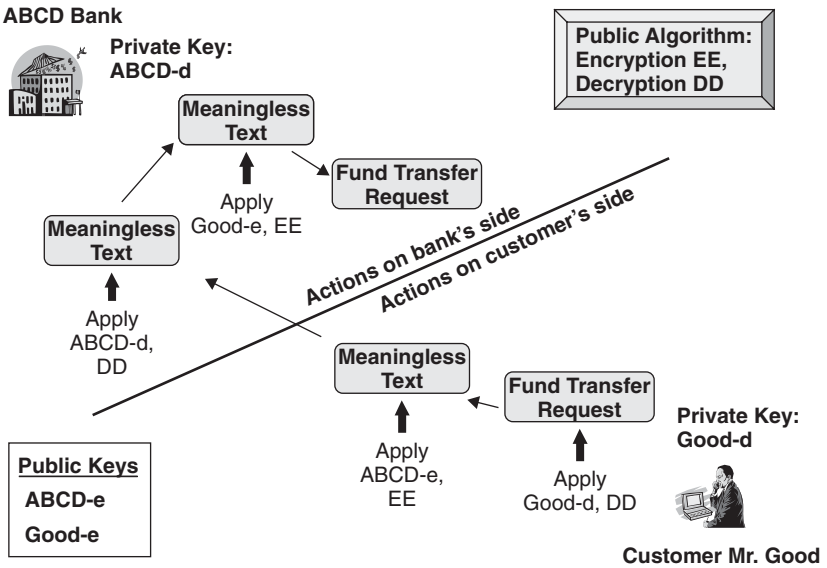

Figure 16-12 Public key encryption: data exchange.

CHAPTER SUMMARY

- An organization's database, being a vital asset, must be protected from unintentional and intentional threats by an effective security system.
- The security system must safeguard a database against theft, vandalism, and fraud; it must preserve the privacy of those about whom data is stored; it must also guarantee proper access privileges to authorized users.
- General security procedures include controls in the physical environment, prevention of malicious acts, and securing of equipment. A strong security policy must guide the procedures. Contingency plans are absolutely necessary.
- Computer-based security measures: protection through authorizing only those who have to be given database access, tailoring access provisions through user views, backup and recovery procedures, and safeguarding of sensitive data by encryption.
- Discretionary access control relates to granting access privileges to authorized users for accessing specific data items on prescribed access modes. Mandatory access control classifies database objects by levels of criticality, assigns security clearance levels to subjects such as users, and matches object levels to subject clearances for providing access.
- Statistical databases pose a special security challenge for protecting privacy. A number of solution options have been proposed.
- Encryption of data provides protection from intruders who attempt to bypass the normal access control mechanisms. The encryption method codes data to make them unintelligible to an intruder and then decodes the data back to original format for use by an authorized user.
- The encryption technique involves encryption and decryption keys along with encryption and decryption algorithms. Symmetric encryption uses the same key for both encryption and decryption; asymmetric encryption uses different keys.
- The Data Encryption Standard (DES) is a single-key encryption technique requiring that the encryption key be kept secret.
- In the public key encryption technique, a public key and a private key are used. The public key is openly available, whereas the private key is chosen and kept secret by the authorized user.

REVIEW QUESTIONS

1. List the major goals and objectives of a database security system. Which ones are important?
2. What are the types of general access control procedures in a database environment?
3. What is data privacy? What are some of the privacy issues to be addressed?
4. What is discretionary access control? What are the types of access privileges available to users?
5. Describe an authorization graph with an example.

6. Name the components of a mandatory access control model.
7. Distinguish between security authorization and authentication.
8. What purposes do statistical databases serve? Name any four types of options to preserve individual privacy in a statistical database.
9. What are the important characteristics of a good encryption technique?
10. Compare DES with the public key encryption technique and list the differences.

EXERCISES

1. Indicate whether true or false:
   A. Discretionary access control provides better protection than mandatory access control.
   B. Biometric verification is an authentication procedure.
   C. Statistical databases have potential data privacy problems.
   D. In the DES encryption technique, the encryption key may be made public.
   E. In the RSA encryption technique, two encryption keys are used.
   F. Simple security restriction relates to discretionary access control.
   G. An authorization graph shows the cycle of how access privileges are passed along.
   H. The DROP VIEW command destroys the data represented in the user view.
   I. The DBMS maintains a profile for each user.
2. As a data security consultant for a large department store, prepare a draft outline of a security policy document.
3. You are the DBA for an airline company. Group the users in your company by functional roles. Indicate four major database tables for the company. Prepare a sample authorization matrix showing user groups, database tables, and access privileges.
4. Explain how user views can be used as security tools. Provide three examples.
5. Choose one of the encryption techniques: DES or RSA. Describe, by means of a simple example, how data is exchanged by using the technique.