Week 3

ACCESS CONTROL

DATABASE AND DATA CENTER SECURITY

Chapter 4

Access Control

Access Control Defined

NISTIR 7298 defines access control as:

- “the process of granting or denying specific requests to: (1) obtain and use information and related information processing services; and (2) enter specific physical facilities”

RFC 4949 defines access control as:

- “a process by which use of system resources is regulated according to a security policy and is permitted only by authorized entities (users, programs, processes, or other systems) according to that policy”

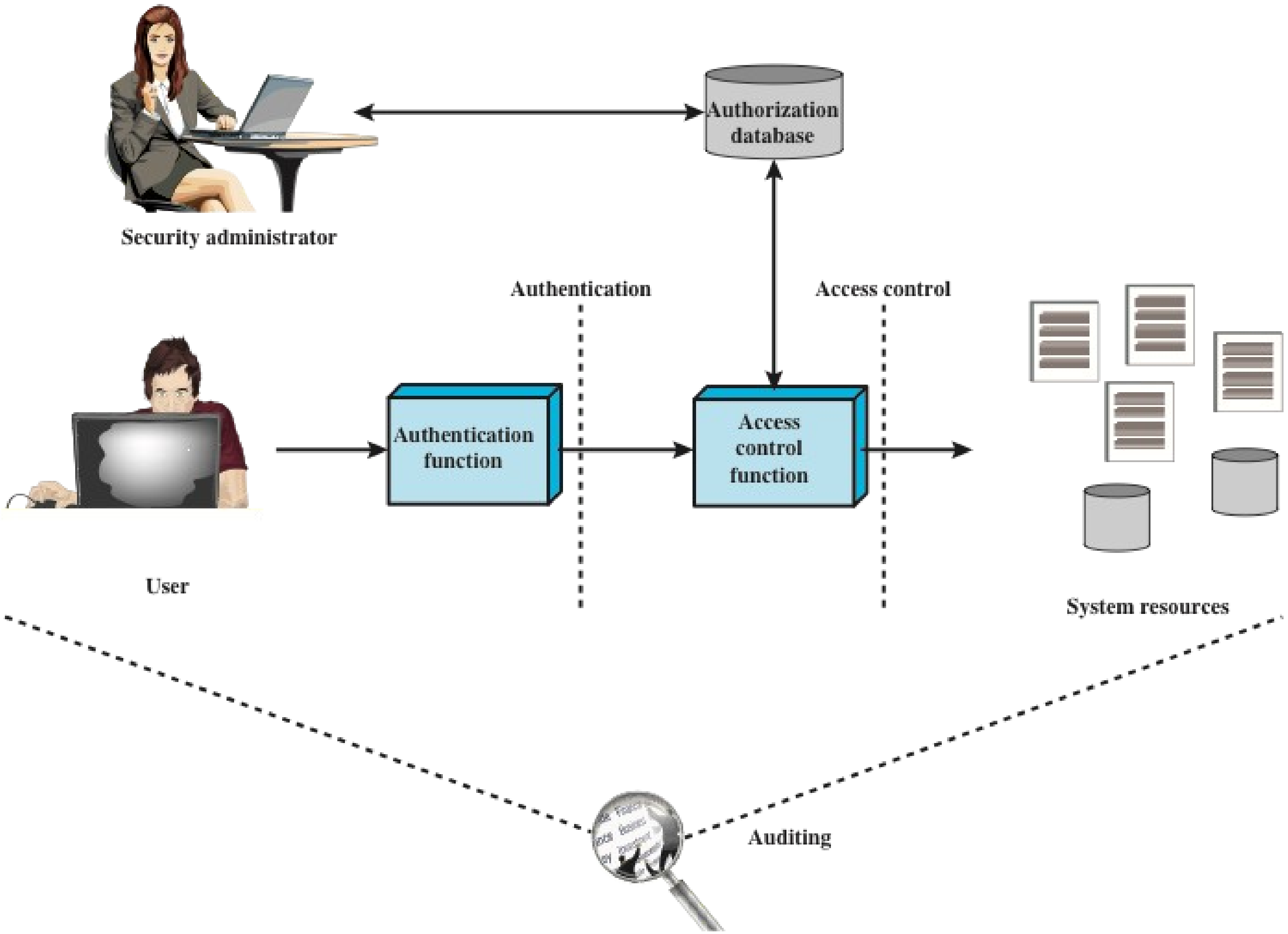


Figure 4.1 Relationship Among Access Control and Other Security Functions

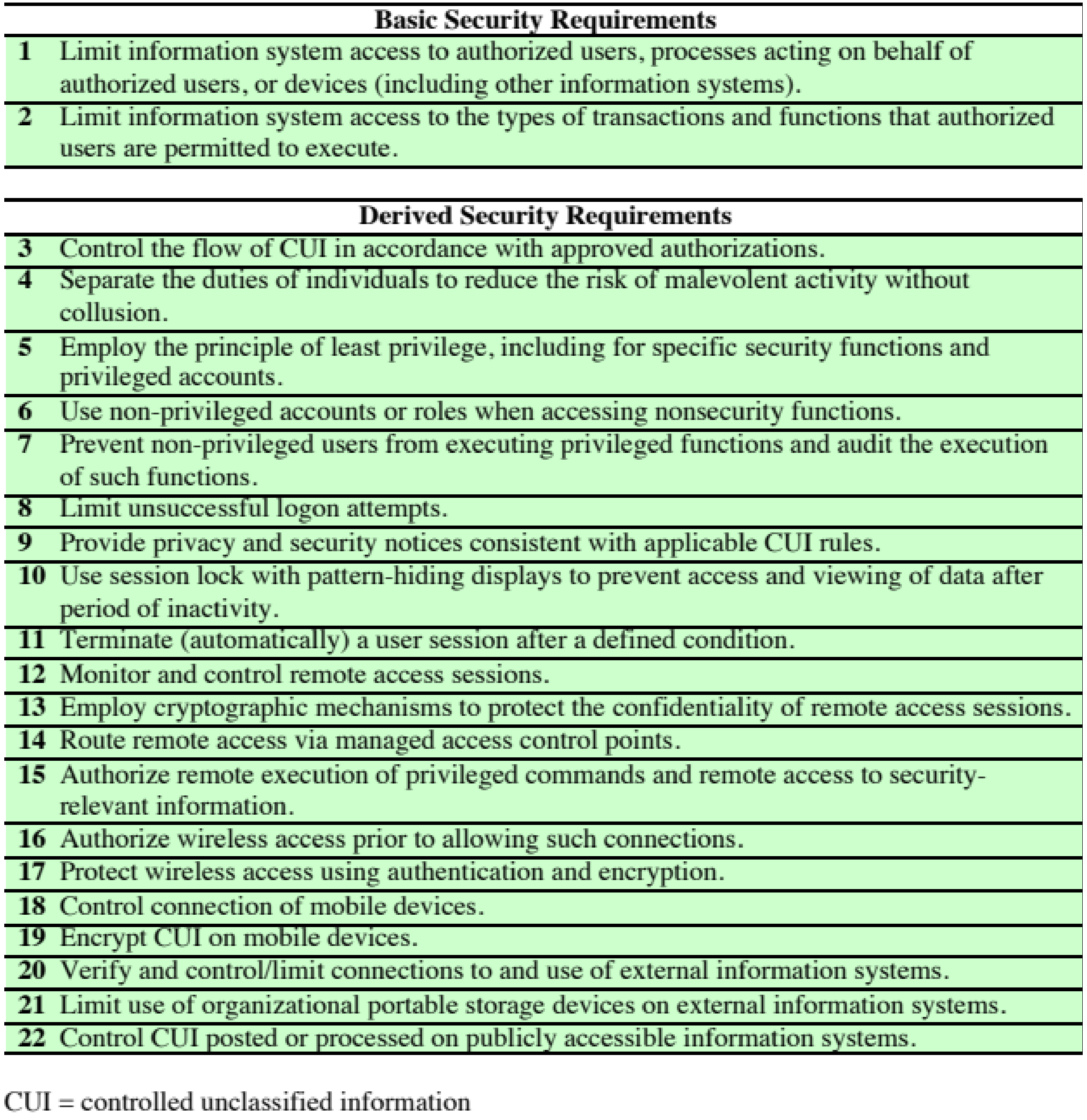
Authentication, Authorization, Audit

Authentication – Verification that the credentials of a user or other system entity are valid (e.g. username/password)

Authorization – Granting of a right to access a system resource (e.g. access to the database within an organization)

Audit – Independent review of a system to test adequacy of system controls and adherence to establish policies

Access Control Security Requirements



Basic Security Requirements	
1	Limit information system access to authorized users, processes acting on behalf of authorized users, or devices (including other information systems).
2	Limit information system access to the types of transactions and functions that authorized users are permitted to execute.

Derived Security Requirements	
3	Control the flow of CUI in accordance with approved authorizations.
4	Separate the duties of individuals to reduce the risk of malevolent activity without collusion.
5	Employ the principle of least privilege, including for specific security functions and privileged accounts.
6	Use non-privileged accounts or roles when accessing nonsecurity functions.
7	Prevent non-privileged users from executing privileged functions and audit the execution of such functions.
8	Limit unsuccessful logon attempts.
9	Provide privacy and security notices consistent with applicable CUI rules.
10	Use session lock with pattern-hiding displays to prevent access and viewing of data after period of inactivity.
11	Terminate (automatically) a user session after a defined condition.
12	Monitor and control remote access sessions.
13	Employ cryptographic mechanisms to protect the confidentiality of remote access sessions.
14	Route remote access via managed access control points.
15	Authorize remote execution of privileged commands and remote access to security-relevant information.
16	Authorize wireless access prior to allowing such connections.
17	Protect wireless access using authentication and encryption.
18	Control connection of mobile devices.
19	Encrypt CUI on mobile devices.
20	Verify and control/limit connections to and use of external information systems.
21	Limit use of organizational portable storage devices on external information systems.
22	Control CUI posted or processed on publicly accessible information systems.

CUI = controlled unclassified information

Access Control Policies

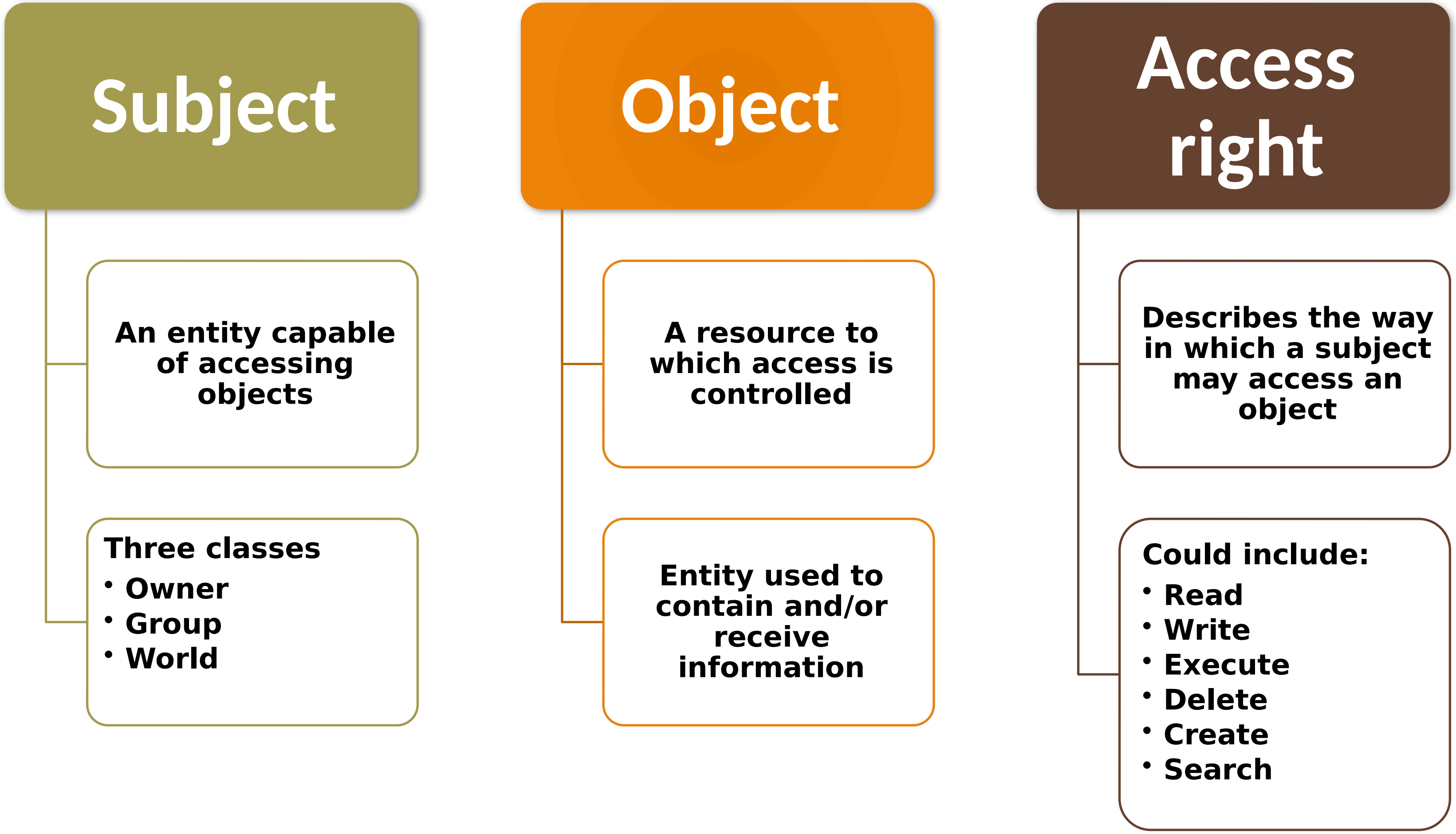
- Discretionary access control (DAC)
  - Controls access based on the identity of the requestor and on access rules (authorizations) stating what requestors are (or are not) allowed to do

- Mandatory access control (MAC)
  - Controls access based on comparing security labels with security clearances

- 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

Subjects, Objects, and Access Rights



Subject

- An entity capable of accessing objects
- Three classes
  - Owner
  - Group
  - World

Object

- A resource to which access is controlled
- Entity used to contain and/or receive information

Access right

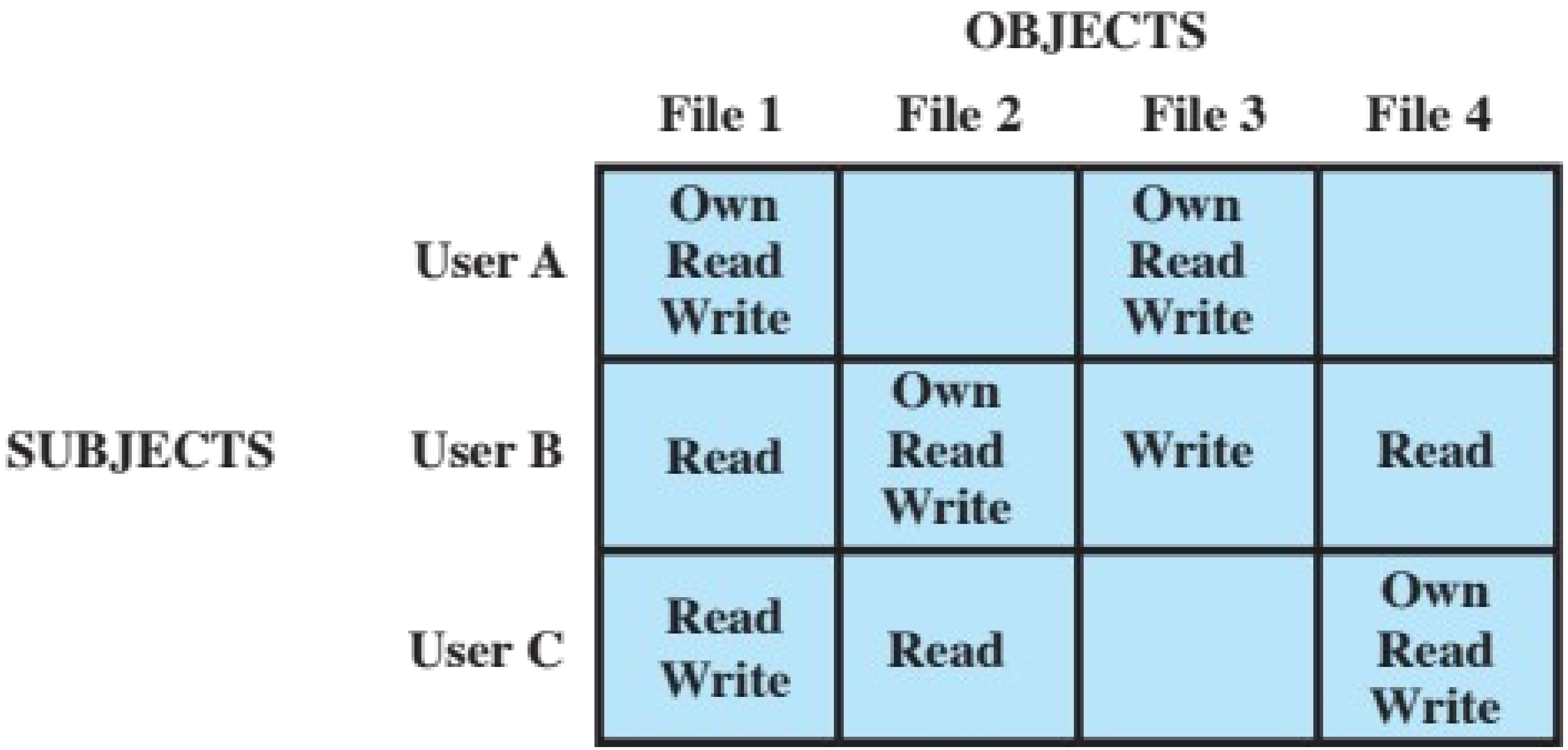
- Describes the way in which a subject may access an object
- Could include:
  - Read
  - Write
  - Execute
  - Delete
  - Create
  - Search

Discretionary Access Control

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

- Often provided using an access matrix
  - One dimension consists of identified subjects that may attempt data access to the resources
  - The other dimension lists the objects that may be accessed
- Each entry in the matrix indicates the access rights of a particular subject for a particular object

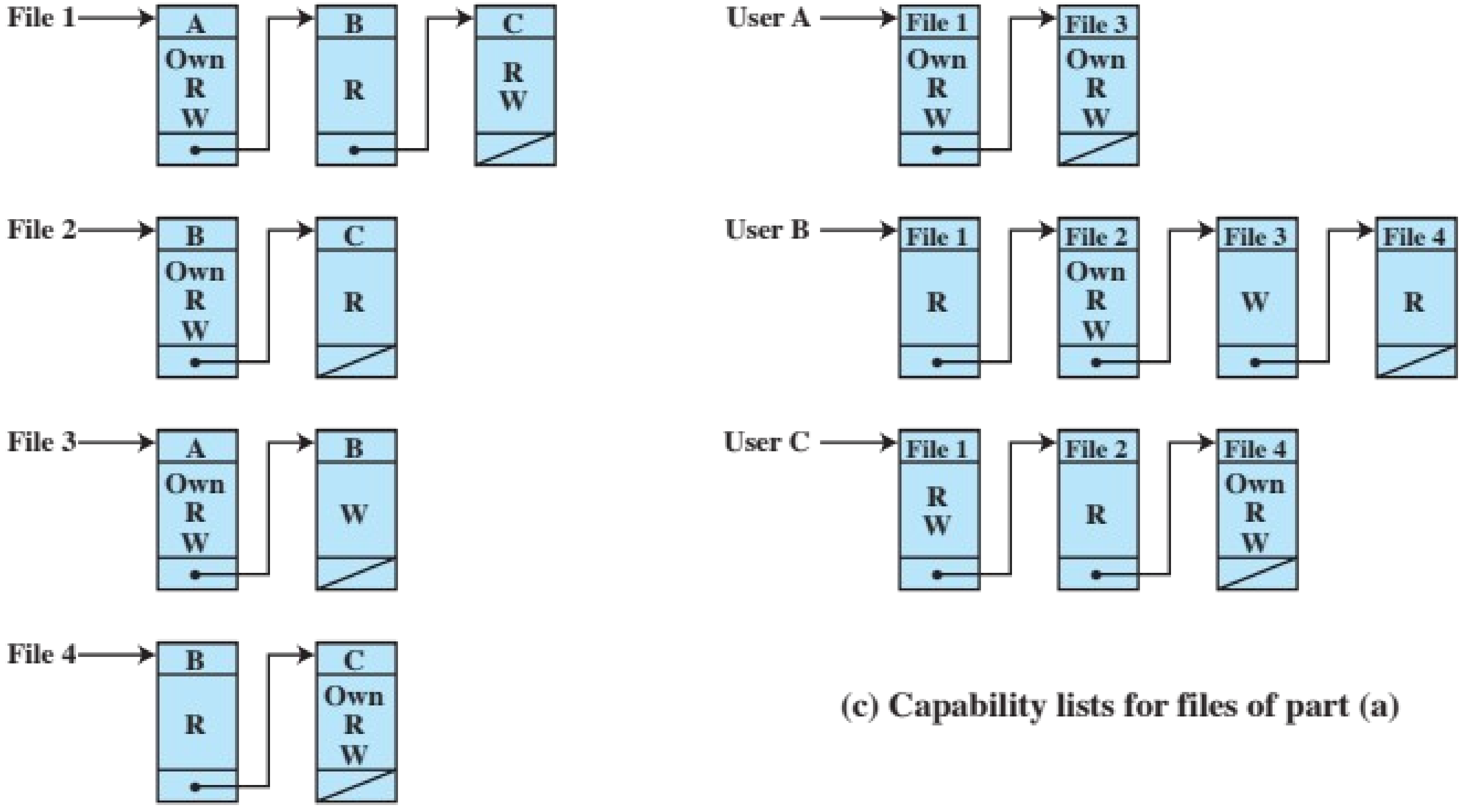
DAC Access Matrix



		OBJECTS			
		File 1	File 2	File 3	File 4
SUBJECTS	User A	Own Read Write		Own Read Write	
	User B	Read	Own Read Write	Write	Read
	User C	Read Write	Read		Own Read Write

(a) Access matrix

Access Control Structures



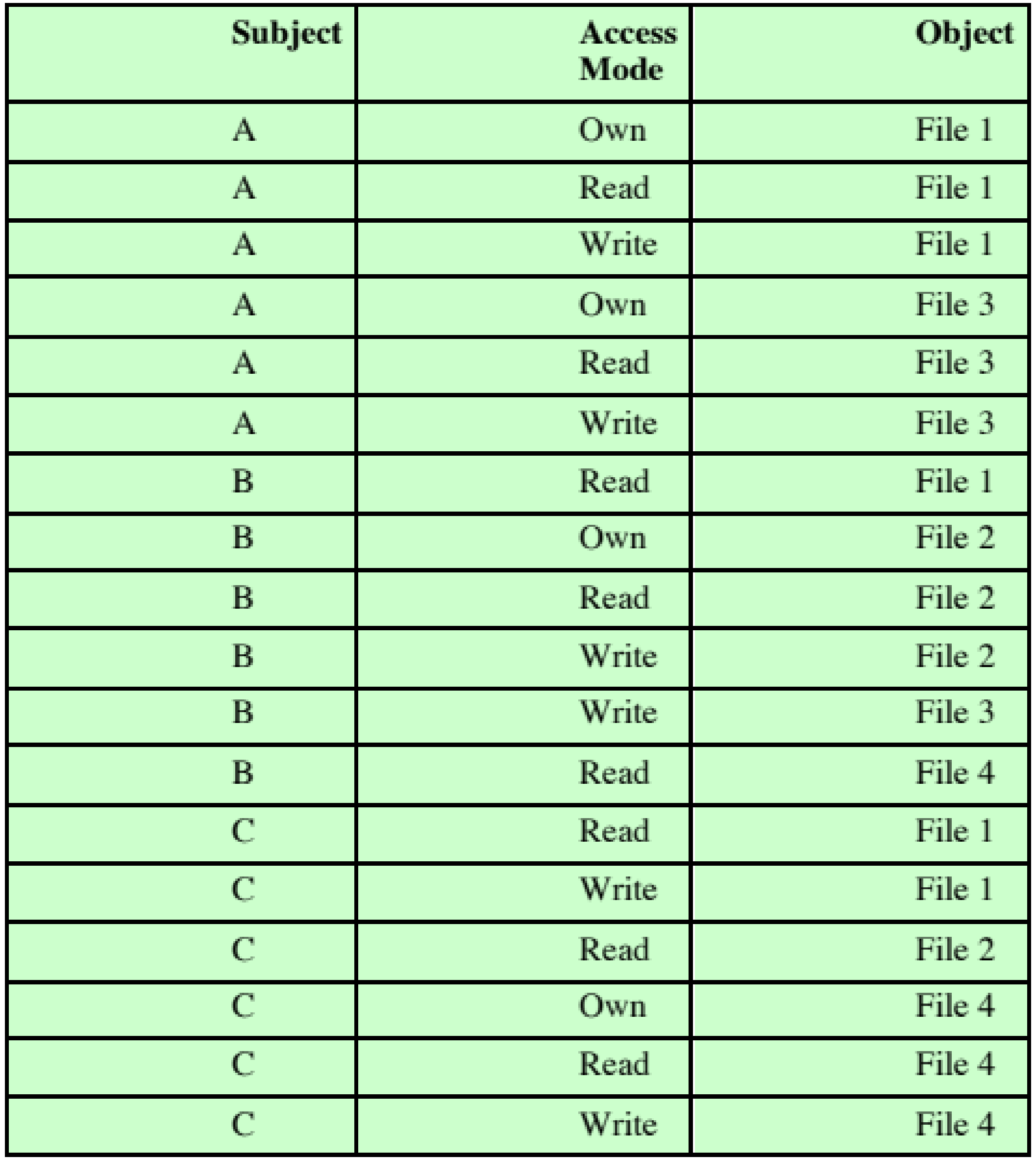
(c) Capability lists for files of part (a)

(b) Access control lists for files of part (a)

Figure 4.2 Example of Access Control Structures

Authorization Table

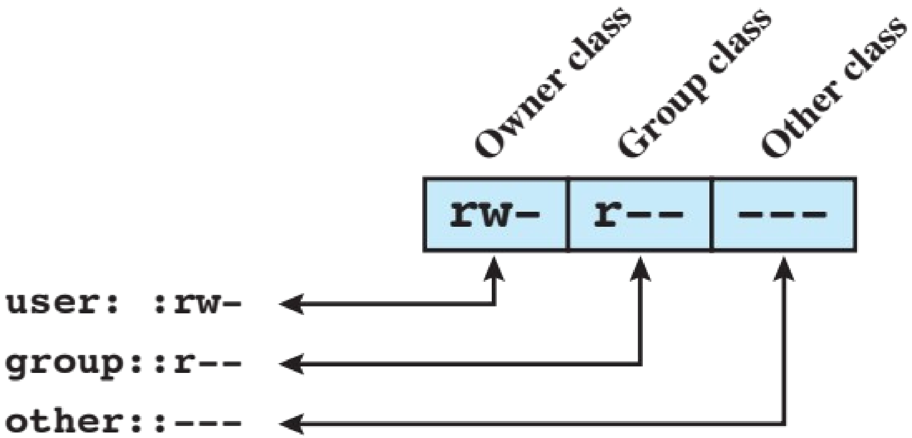
Proposes a data structure that is not sparse (access matrix) but more convenient than ACLs or capability lists



Subject	Access Mode	Object
A	Own	File 1
A	Read	File 1
A	Write	File 1
A	Own	File 3
A	Read	File 3
A	Write	File 3
B	Read	File 1
B	Own	File 2
B	Read	File 2
B	Write	File 2
B	Write	File 3
B	Read	File 4
C	Read	File 1
C	Write	File 1
C	Read	File 2
C	Own	File 4
C	Read	File 4
C	Write	File 4

Example: Unix File Access Control

- Unique user identification number (user ID)
- Member of a primary group identified by a group ID
- Belongs to a specific group
- 12 protection bits
  - Specify read, write, and execute permission for the owner of the file, members of the group and all other users
  - The owner ID, group ID, and protection bits are part of the file's inode



(a) Traditional UNIX approach (minimal access control list)

Users, Roles, and Resources

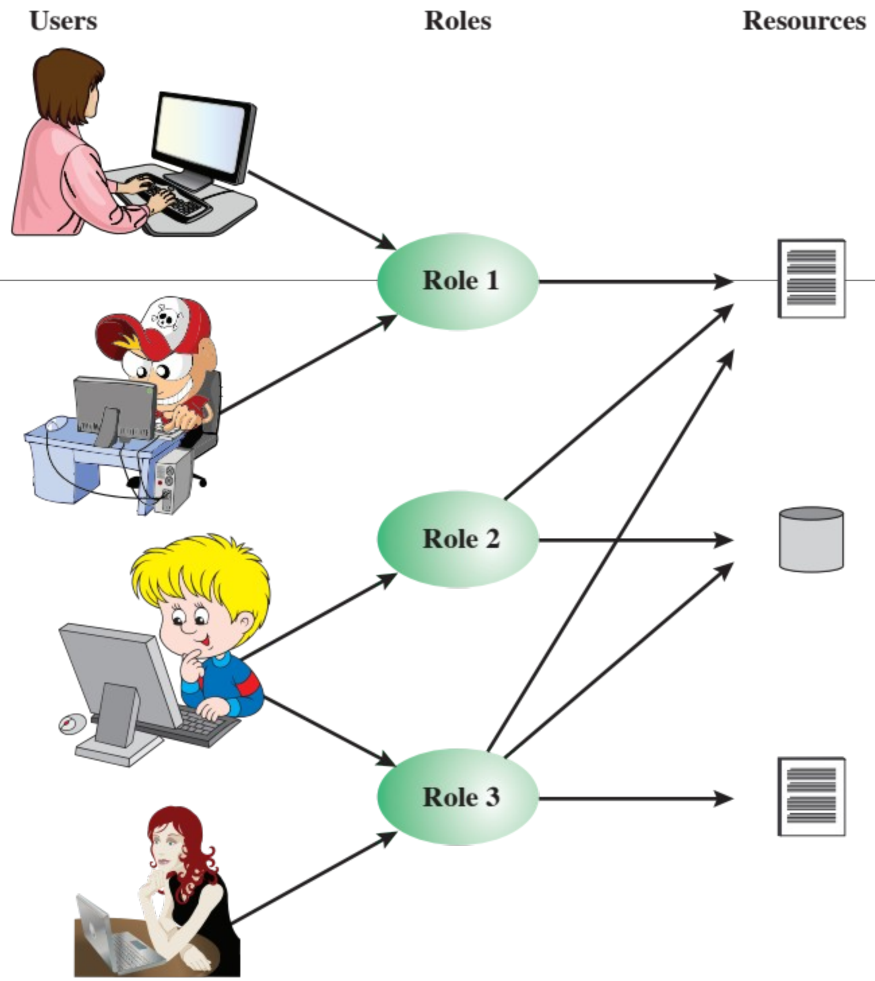
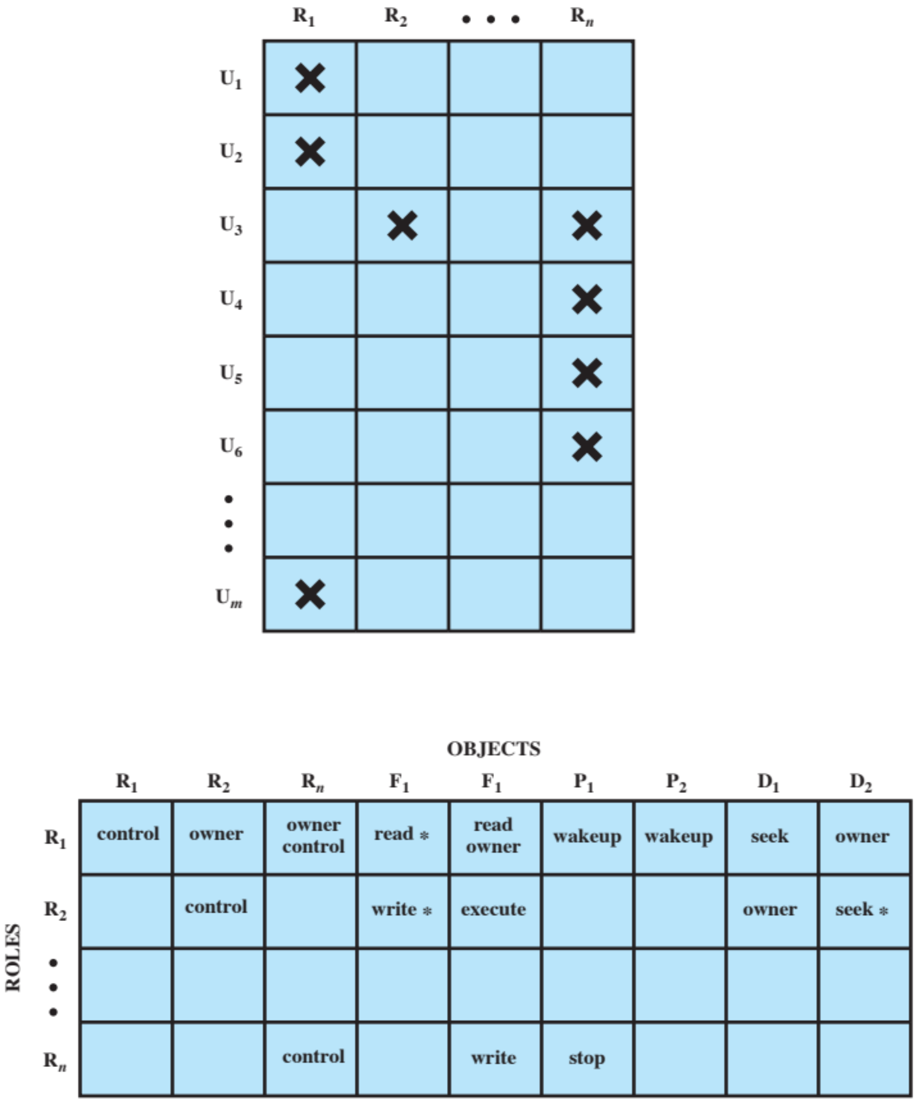


Figure 4.6 Users, Roles, and Resources

Matrix Representation of RBAC



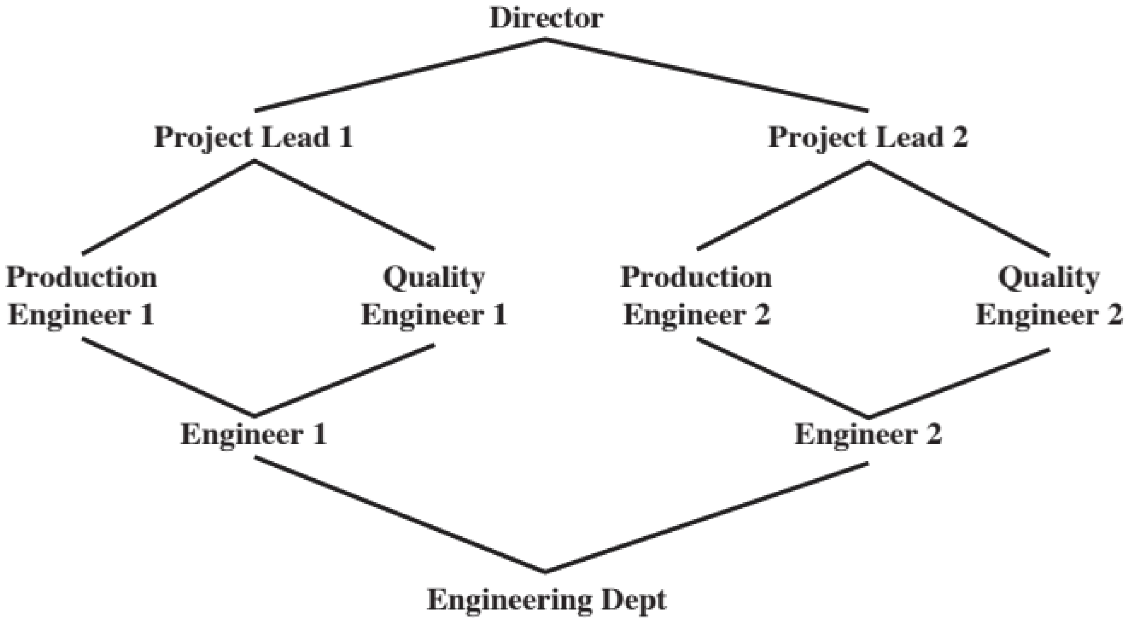
	R_1	R_2	• • •	R_n
U_1	✗			
U_2	✗			
U_3		✗		✗
U_4				✗
U_5				✗
U_6				✗
⋮				
U_m	✗			

OBJECTS

ROLES	R_1	R_2	R_n	F_1	F_1	P_1	P_2	D_1	D_2
R_1	control	owner	owner control	read *	read owner	wakeup	wakeup	seek	owner
R_2		control		write *	execute			owner	seek *
⋮									
R_n			control		write	stop			

Figure 4.7 Access Control Matrix Representation of RBAC

Role Hierarchy



Director

Project Lead 1

Project Lead 2

Production Engineer 1

Quality Engineer 1

Production Engineer 2

Quality Engineer 2

Engineer 1

Engineer 2

Engineering Dept

Figure 4.9 Example of Role Hierarchy

A role hierarchy definition can be utilized to define access control

Adapting RBAC to Policies

Mutually exclusive roles

- A user can only be assigned to one role in the set (either during a session or statically)
- Any permission (access right) can be granted to only one role in the set

Cardinality

- Setting a maximum number of users that can be assigned to a role

Prerequisite roles

- Dictates that a user can only be assigned to a particular role if it is already assigned to some other specified role

Attribute-Based Access Control (ABAC)

Can define authorizations that express conditions on properties of both the resource and the subject

Strength is its flexibility and expressive power

Main obstacle to its adoption in real systems has been concern about the performance impact of evaluating predicates on both resource and user properties for each access

Web services have been pioneering technologies through the introduction of the eXtensible Access Control Markup Language (XAMCL)

There is considerable interest in applying the model to cloud services

ABAC Attributes

Subject attributes

- A subject is an active entity that causes information to flow among objects or changes the system state
- Attributes define the identity and characteristics of the subject
- Example: User, Application

Object attributes

- An object (or resource) is a passive information system-related entity containing or receiving information
- Objects have attributes that can be leverages to make access control decisions
- Example: Devices, network, domains

Environment attributes

- Describe the operational, technical, and even situational environment or context in which the information access occurs
- These attributes have so far been largely ignored in most access control policies
- Example: Date, time

Benefits of ABAC

- Distinguishable because it controls access to objects by evaluating rules against the attributes of entities, operations, and the environment relevant to a request
- Relies upon the evaluation of attributes of the subject, attributes of the object, and a formal relationship or access control rule defining the allowable operations for subject-object attribute combinations in a given environment
- Systems are capable of enforcing DAC, RBAC, and MAC concepts
- Allows an unlimited number of attributes to be combined to satisfy any access control rule

ABAC - Depicted

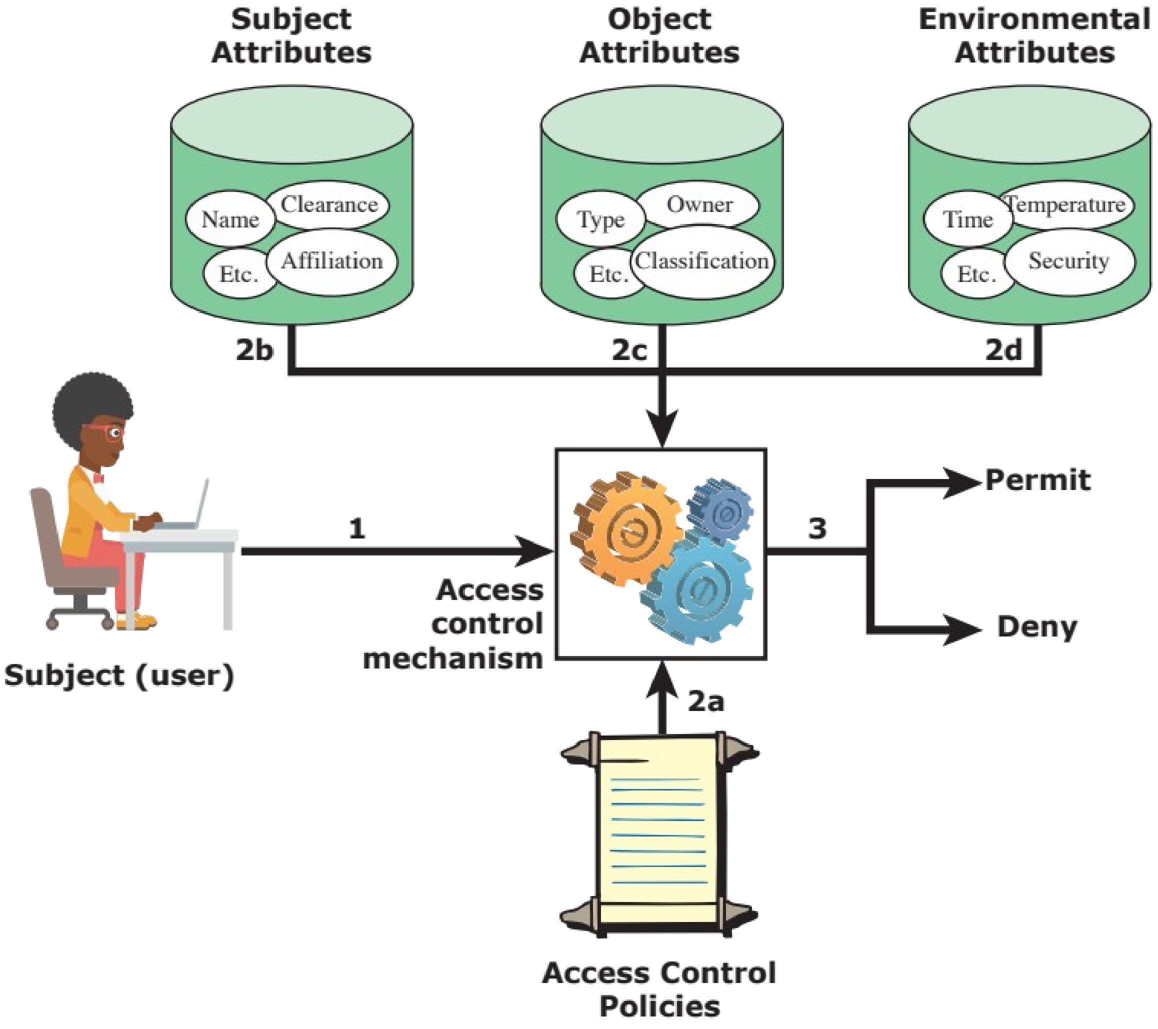
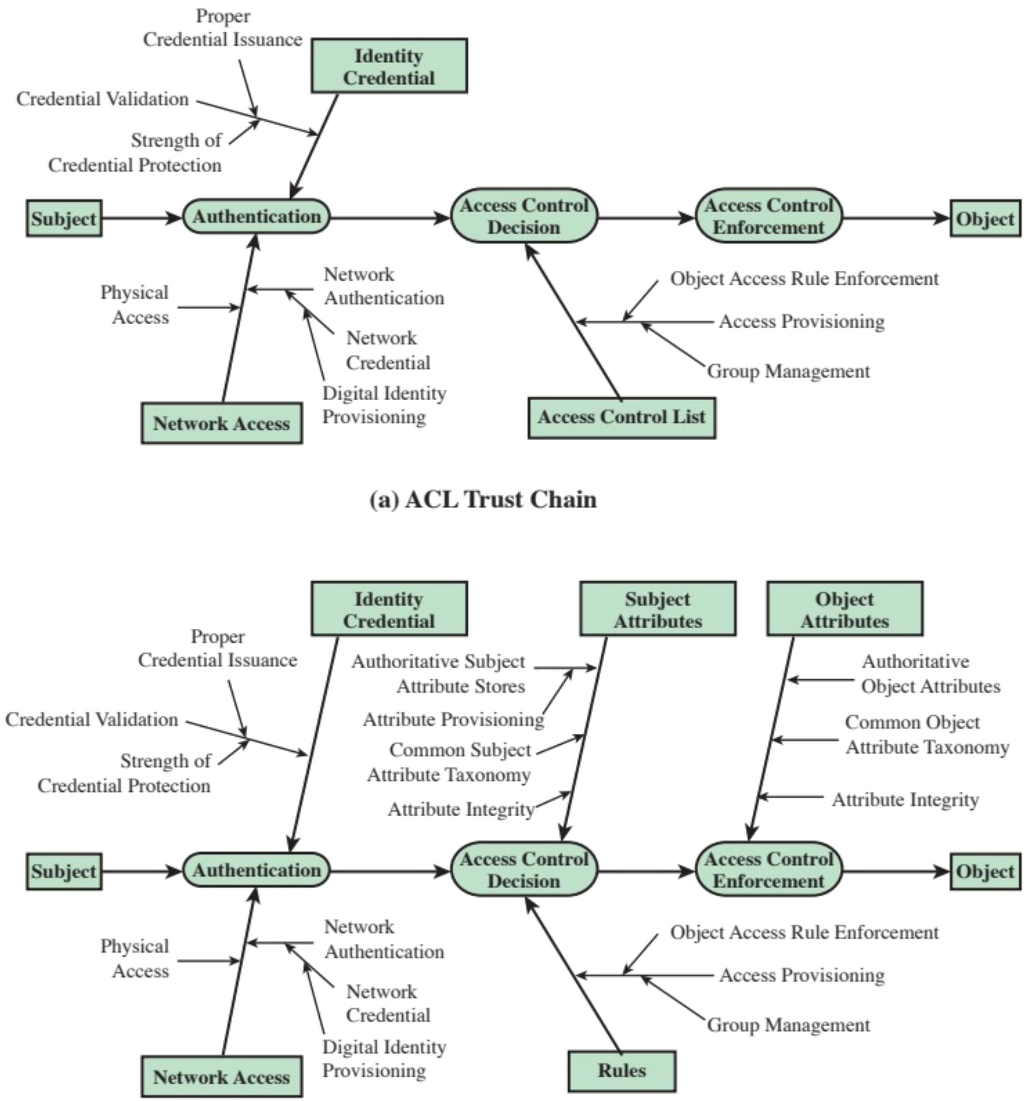


Figure 4.10 ABAC Scenario

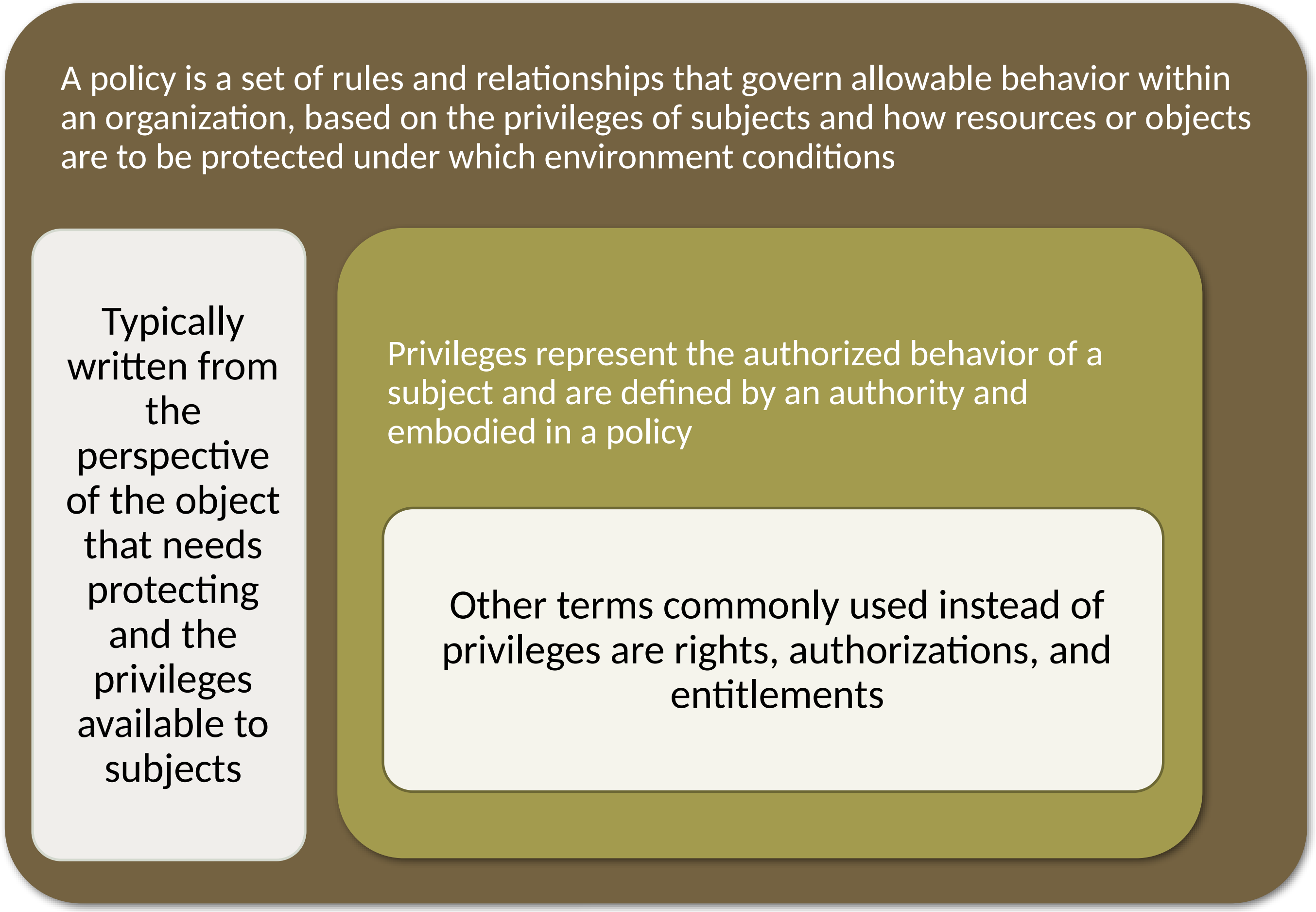
ACL and ABAC Relationship



(a) ACL Trust Chain

(b) ABAC Trust Chain

Example: ABAC Policy



A policy is a set of rules and relationships that govern allowable behavior within an organization, based on the privileges of subjects and how resources or objects are to be protected under which environment conditions

Typically written from the perspective of the object that needs protecting and the privileges available to subjects

Privileges represent the authorized behavior of a subject and are defined by an authority and embodied in a policy

Other terms commonly used instead of privileges are rights, authorizations, and entitlements

Consider an online video streaming site’s ABAC policy that governs access to R, PG-13, and G rated movies. That policy will look something like:

```
R1:can_access((u)ser, (m)ovie) ->

{

Age(u) >= 17 && Rating(m) == (R, PG-13, G)
||
Age(u) >= 13 && Age(u) < 17 && Rating(m) == (PG-13, G)
||
Age(u) < 13 && Rating(m) == (G)

}
```

Identity Credential and Access Management

- A comprehensive approach to managing and implementing digital identities, credentials, and access control
- Developed by the U.S. government
- Designed to:
  - Create trusted digital identity representations of individuals and nonperson entities (NPEs)
  - Bind those identities to credentials that may serve as a proxy for the individual of NPE in access transactions
    - A credential is an object or data structure that authoritatively binds an identity to a token possessed and controlled by a subscriber
  - Use the credentials to provide authorized access to an agency’s resources

ICAM (or more commonly called IAM)

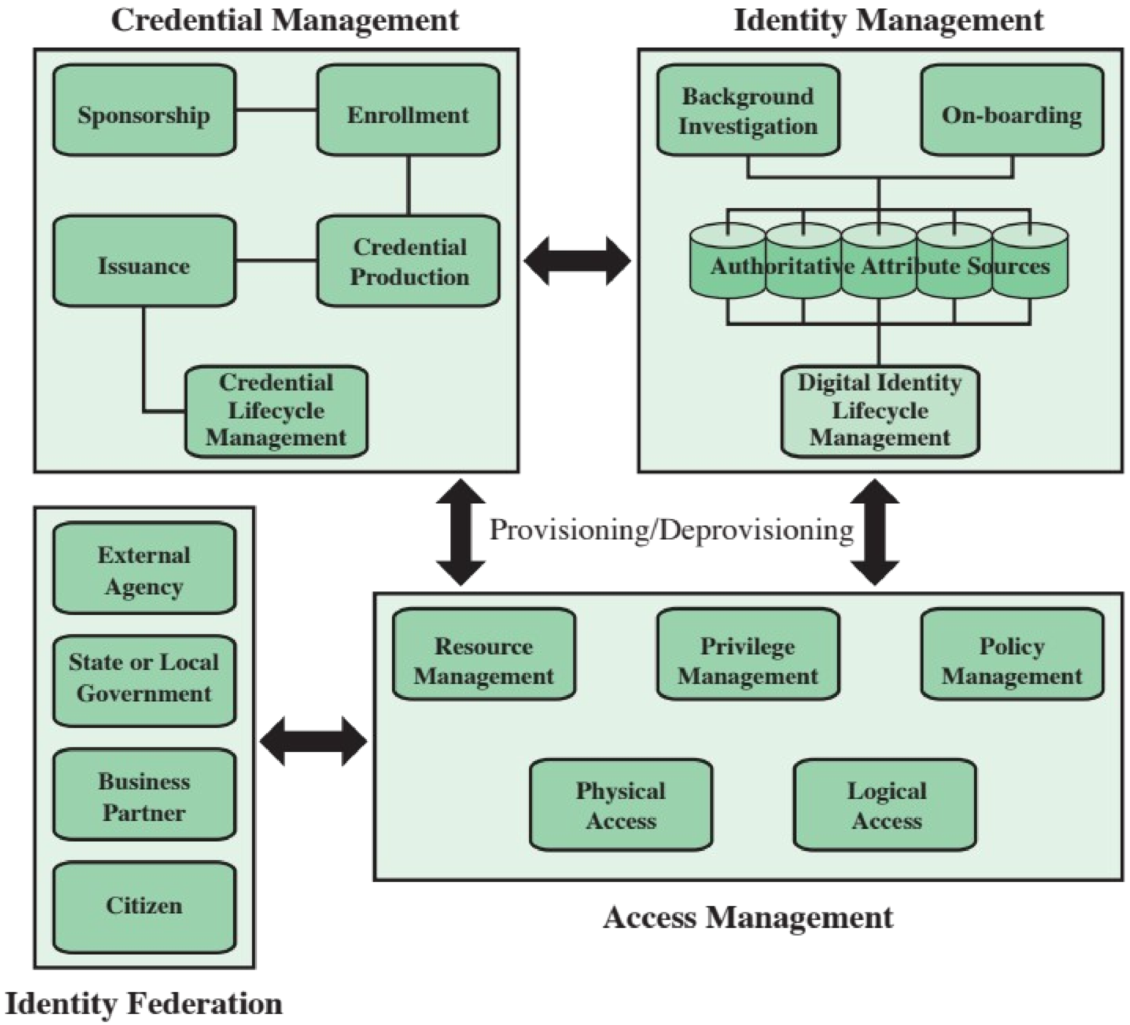


Figure 4.12 Identity, Credential, and Access Management (ICAM)

Identity Management

Concerned with assigning attributes to a digital identity and connecting that digital identity to an individual or NPE

Goal is to establish a trustworthy digital identity that is independent of a specific application or context

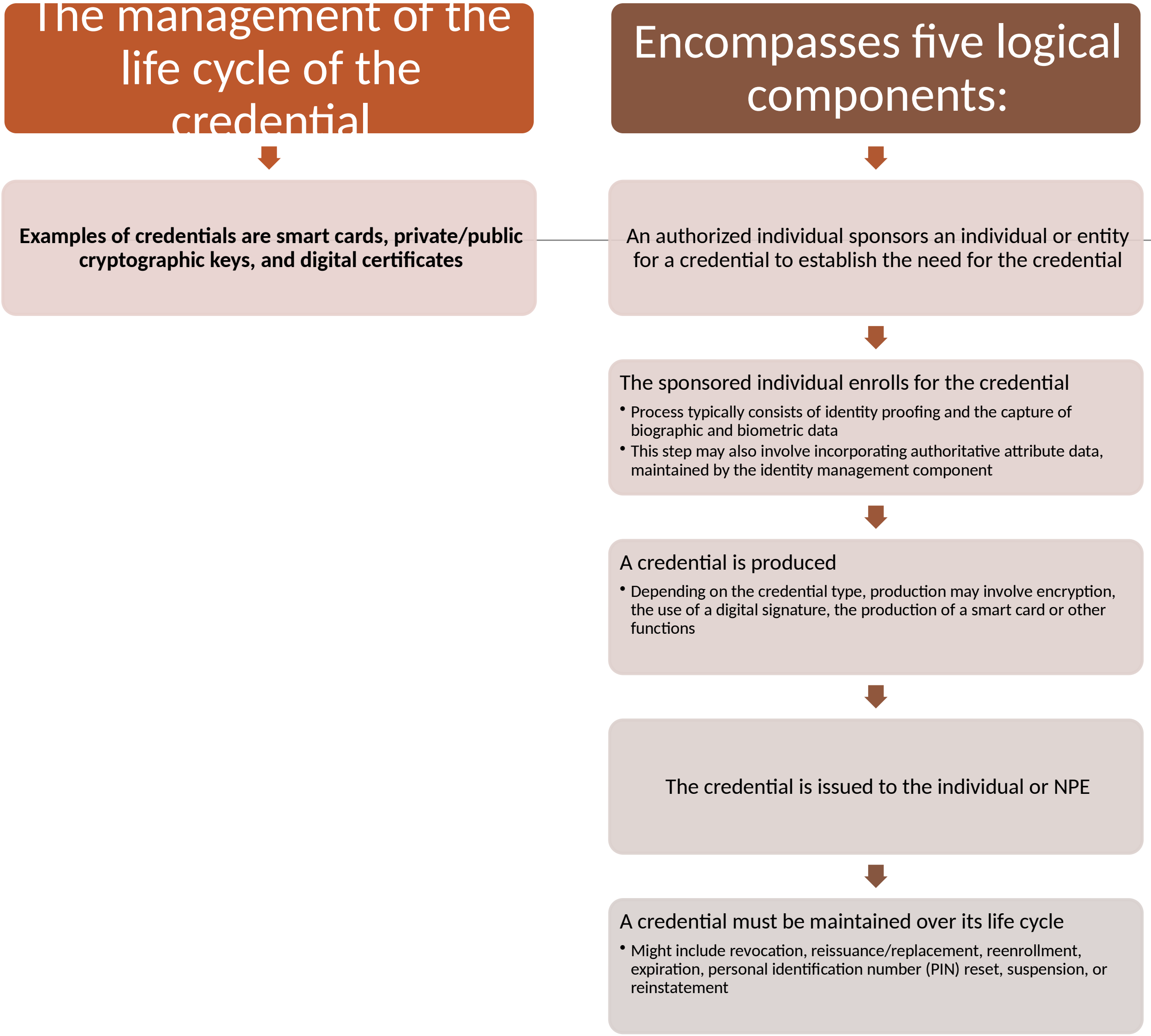
Most common approach to access control for applications and programs is to create a digital representation of an identity for the specific use of the application or program

Maintenance and protection of the identity itself is treated as secondary to the mission associated with the application

Techniques for sharing authoritative identity data with applications that need it

Revocation of an enterprise identity

Credential Management



The management of the life cycle of the credential

Examples of credentials are smart cards, private/public cryptographic keys, and digital certificates

Encompasses five logical components:

An authorized individual sponsors an individual or entity for a credential to establish the need for the credential

The sponsored individual enrolls for the credential

- Process typically consists of identity proofing and the capture of biographic and biometric data
- This step may also involve incorporating authoritative attribute data, maintained by the identity management component

A credential is produced

- Depending on the credential type, production may involve encryption, the use of a digital signature, the production of a smart card or other functions

The credential is issued to the individual or NPE

A credential must be maintained over its life cycle

- Might include revocation, reissuance/replacement, reenrollment, expiration, personal identification number (PIN) reset, suspension, or reinstatement

Access Management

Deals with the management and control of the ways entities are granted access to resources

Covers both logical and physical access

May be internal to a system or an external element

Purpose is to ensure that the proper identity verification is made when an individual attempts to access a security sensitive building, computer systems, or data

Three support elements are needed for an enterprise-wide access control facility:

- Resource management
- Privilege management
- Policy management

Requirements for an Enterprise-wide Access Control Facility

Resource management

- Concerned with defining rules for a resource that requires access control
- Rules would include credential requirements and what user attributes, resource attributes, and environmental conditions are required for access of a given resource for a given function

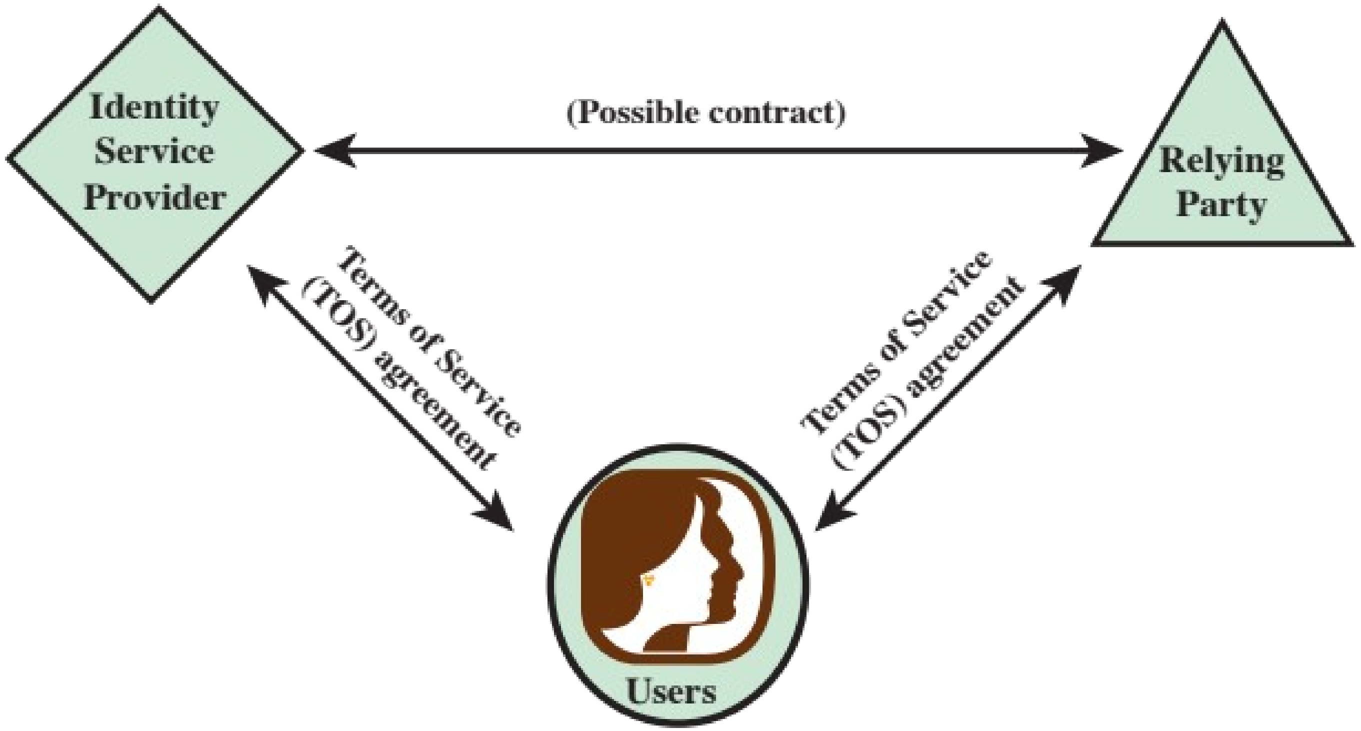
Privilege management

- Concerned with establishing and maintaining the entitlement or privilege attributes that comprise an individual’s access profile
- These attributes represent features of an individual that can be used as the basis for determining access decisions to both physical and logical resources
- Privileges are considered attributes that can be linked to a digital identity

Policy management

- Governs what is allowable and unallowable in an access transaction

Identity Informati on Exchange


Identity Service Provider
(Possible contract)
Relying Party
Terms of Service (TOS) agreement
Terms of Service (TOS) agreement
Users

(a) Traditional triangle of parties involved in an exchange of identity information

Some Open Identity Trust Frameworks

OpenID

- An open standard that allows users to be authenticated by certain cooperating sites using a third party service
- OAuth, OpenID Connect, SAML

OIDF

- OpenID Foundation is an international nonprofit organization of individuals and companies committed to enabling, promoting, and protecting OpenID technologies

ICF

- Information Card Foundation is a nonprofit community of companies and individuals working together to evolve the Information Card ecosystem

OITF

- Open Identity Trust Framework is a standardized, open specification of a trust framework for identity and attribute exchange, developed jointly by OIDF and ICF

OIX

- Open Identity Exchange Corporation is an independent, neutral, international provider of certification trust frameworks conforming to the OITF model

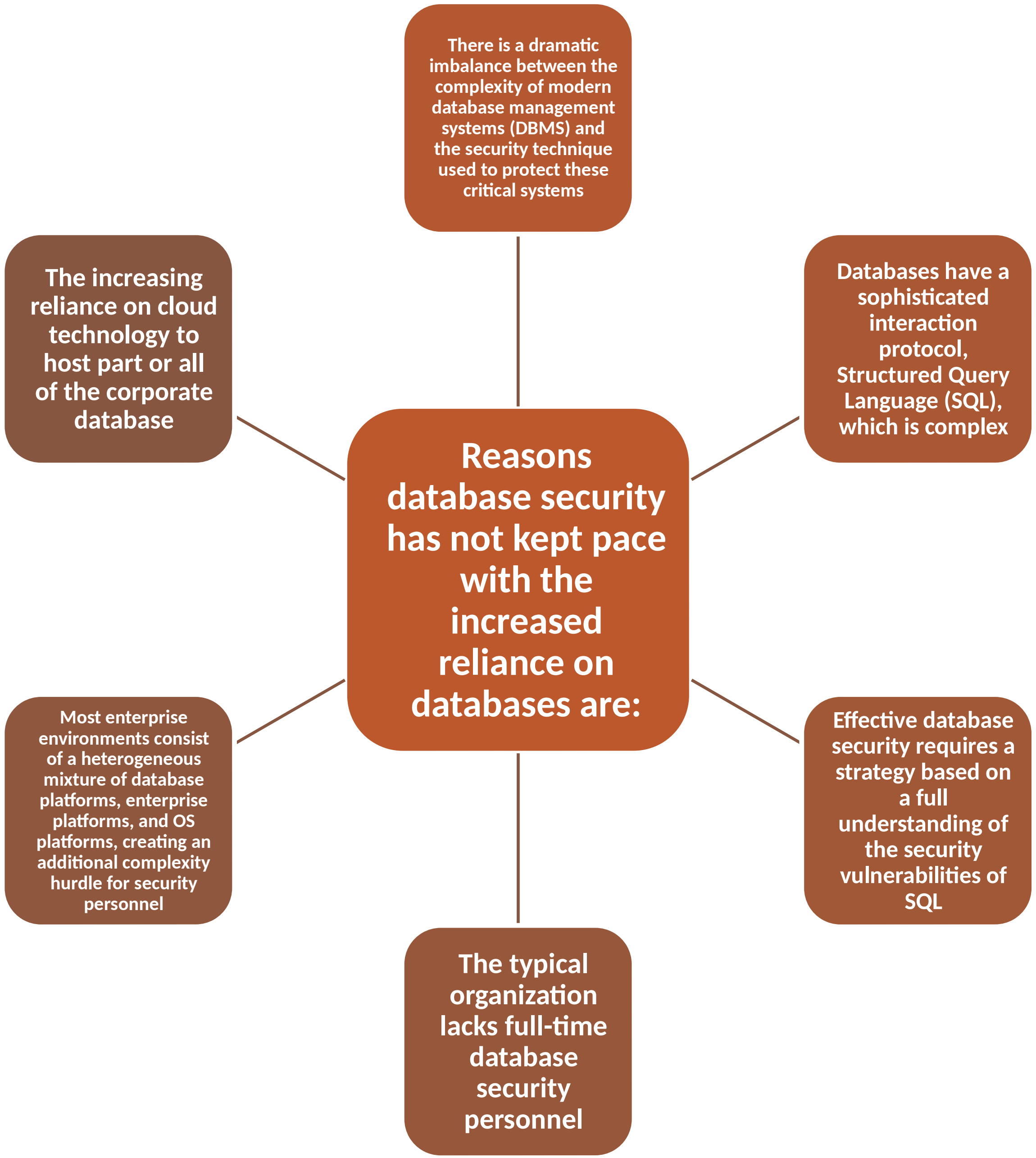
AXN

- Attribute Exchange Network is an online Internet-scale gateway for identity service providers and relying parties to efficiently access user asserted, permissioned, and verified online identity attributes in high volumes at affordable costs

Chapter 5

Database and Data Center Security

Database Security



Reasons database security has not kept pace with the increased reliance on databases are:

There is a dramatic imbalance between the complexity of modern database management systems (DBMS) and the security technique used to protect these critical systems

Databases have a sophisticated interaction protocol, Structured Query Language (SQL), which is complex

Effective database security requires a strategy based on a full understanding of the security vulnerabilities of SQL

The typical organization lacks full-time database security personnel

Most enterprise environments consist of a heterogeneous mixture of database platforms, enterprise platforms, and OS platforms, creating an additional complexity hurdle for security personnel

The increasing reliance on cloud technology to host part or all of the corporate database

Databases

- Structured collection of data stored for use by one or more applications
- Contains the relationships between data items and groups of data items
- Can sometimes contain sensitive data that needs to be secured
  - Query language
  - Provides a uniform interface to the database for users and applications

Database management system (DBMS)

- Suite of programs for constructing and maintaining the database
- Offers ad hoc query facilities to multiple users and applications

Relational Database Model

- Table of data consisting of rows and columns
  - Each column holds a particular type of data
  - Each row contains a specific value for each column
  - Ideally has one column where all values are unique, forming an identifier/key for that row
- Enables the creation of multiple tables linked together by a unique identifier that is present in all tables
- Use a relational query language to access the database
  - Allows the user to request data that fit a given set of criteria

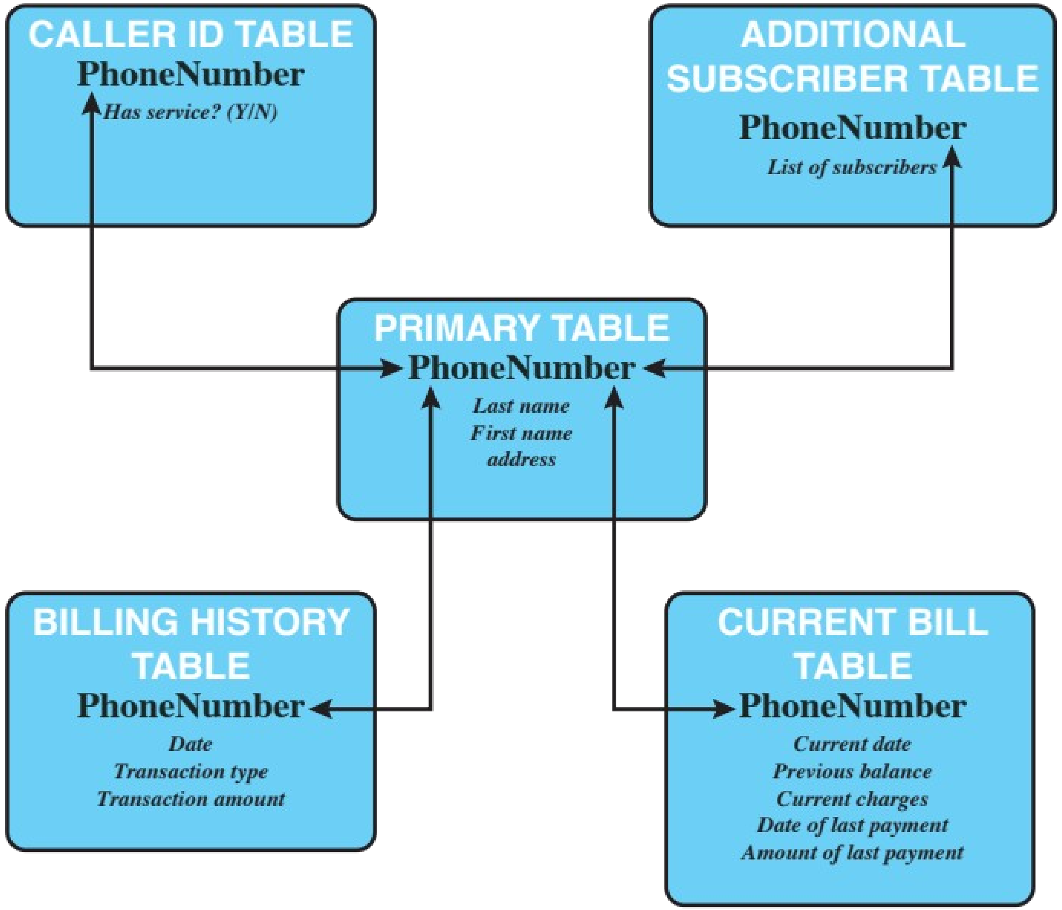


Figure 5.2 Example Relational Database Model. A relational database uses multiple tables related to one another by a designated key; in this case the key is the PhoneNumber field.

Relational Database Elements

- Relation
  - Table/file
- Tuple
  - Row/record
- Attribute
  - Column/field

Primary key

- Uniquely identifies a row
- Consists of one or more column names

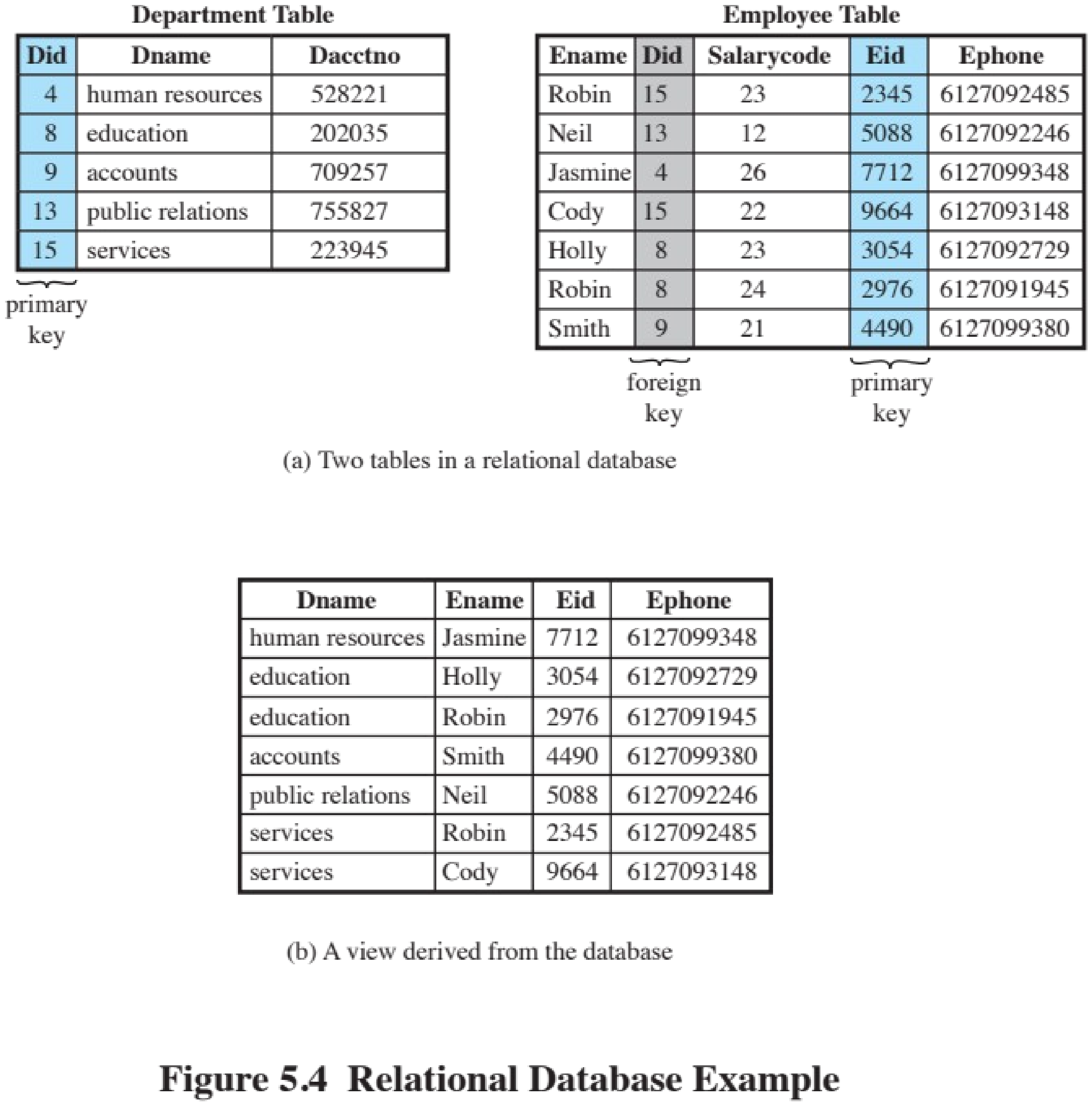
Foreign key

- Links one table to attributes in another

View/virtual table

- Result of a query that returns selected rows and columns from one or more tables
- Views are often used for security purposes

Database Example



Department Table

Did	Dname	Dacctno
4	human resources	528221
8	education	202035
9	accounts	709257
13	public relations	755827
15	services	223945

primary key

Employee Table

Ename	Did	Salarycode	Eid	Ephone
Robin	15	23	2345	6127092485
Neil	13	12	5088	6127092246
Jasmine	4	26	7712	6127099348
Cody	15	22	9664	6127093148
Holly	8	23	3054	6127092729
Robin	8	24	2976	6127091945
Smith	9	21	4490	6127099380

foreign key

primary key

(a) Two tables in a relational database

Dname	Ename	Eid	Ephone
human resources	Jasmine	7712	6127099348
education	Holly	3054	6127092729
education	Robin	2976	6127091945
accounts	Smith	4490	6127099380
public relations	Neil	5088	6127092246
services	Robin	2345	6127092485
services	Cody	9664	6127093148

(b) A view derived from the database

Figure 5.4 Relational Database Example

Structured Query Language (SQL)

- Standardized language to define schema, manipulate, and query data in a relational database
- Several similar versions of ANSI/ISO standard
- All follow the same basic syntax and semantics

SQL statements can be used to:

- Create tables
- Insert and delete data in tables
- Create views
- Retrieve data with query statements

SQL Injection Attacks (SQLi)

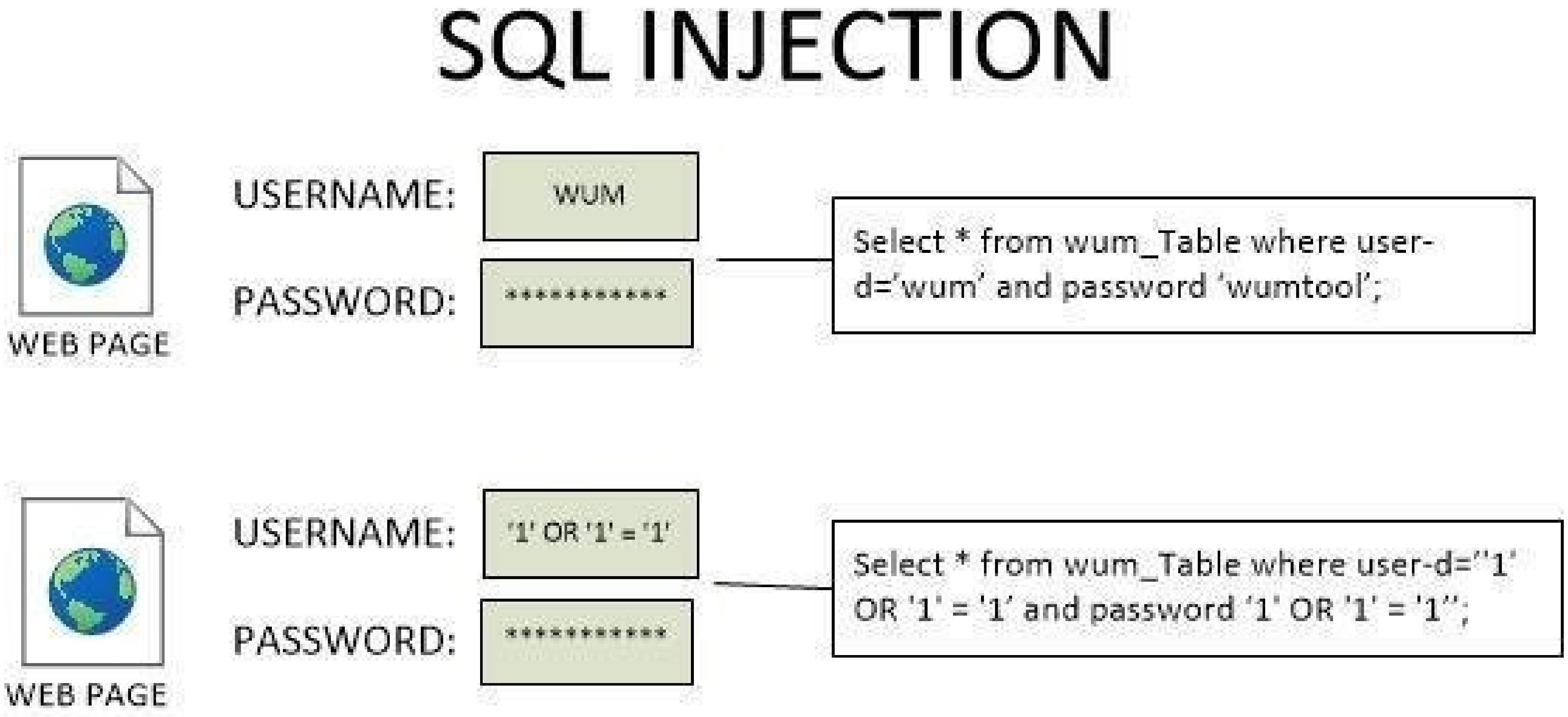
- One of the most prevalent and dangerous network-based security threats

- Designed to exploit the nature of Web application pages

- Sends malicious SQL commands to the database server

- Most common attack goal is bulk extraction of data

- Depending on the environment SQL injection can also be exploited to:
  - Modify or delete data
  - Execute arbitrary operating system commands
  - Launch denial-of-service (DoS) attacks



```
Select * from wum_Table where user-
d=’wum’ and password ‘wumtool’;
```

```
Select * from wum_Table where user-d=’‘1’
OR '1' = ‘1’ and password ‘1' OR '1' = '1’’;
```

SQLi Example

SQLi Defensive Coding

Consider 2 ways of providing a service that lets students sign up for a new account:

```
PreparedStatement stmt = conn.createStatement("INSERT INTO students VALUES('" +
user + "')");
stmt.execute();
```

```
PreparedStatement stmt = conn.prepareStatement("INSERT INTO student
VALUES(?)");
stmt.setString(1, user);
stmt.execute();
```

What happens if the ‘user’ came from user input and was:

```
Robert'); DROP TABLE students; --
```

SQLi Defensive Coding (cont.)

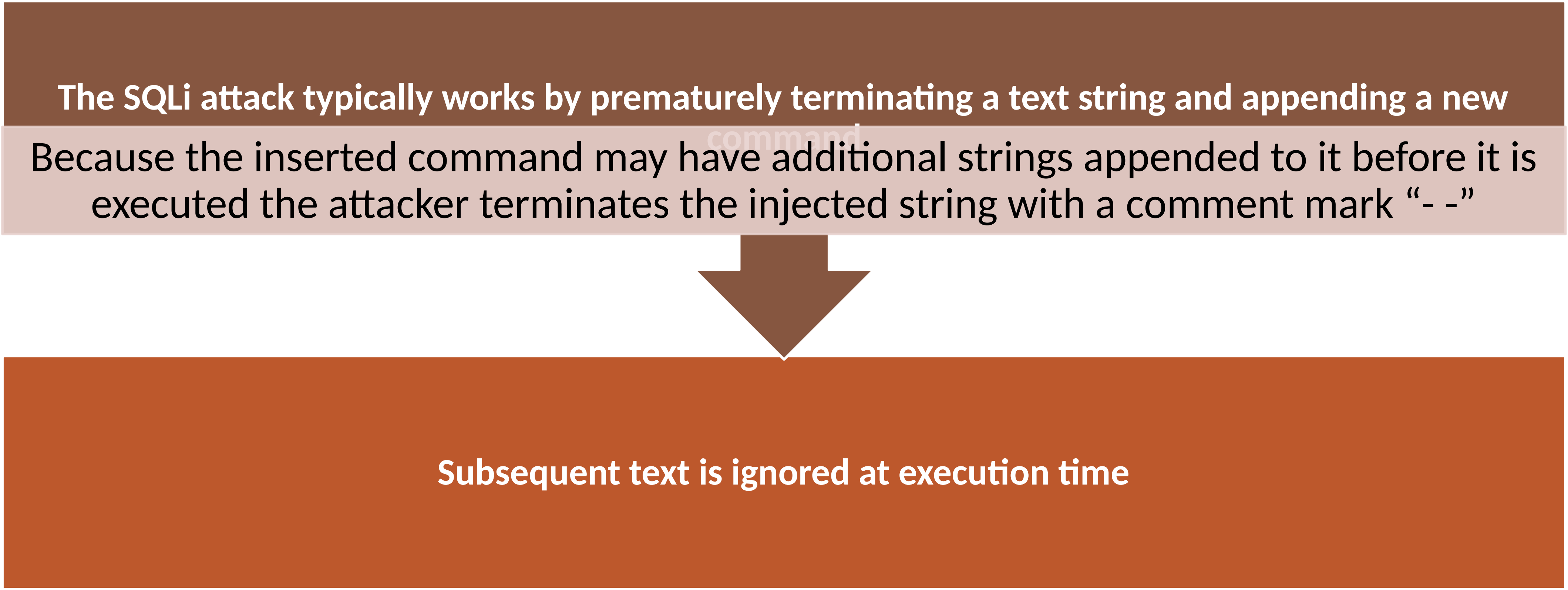
```
PreparedStatement stmt = conn.createStatement("INSERT INTO students
VALUES('Robert'); DROP TABLE students; --’”);
```

^ Your DB is hosed!

```
PreparedStatement stmt = conn.prepareStatement("INSERT INTO student
VALUES(?)");
stmt.setString(1, user);
☾
conn.prepareStatement("INSERT INTO student VALUES(Robert'); DROP TABLE
students; --)");
```

^ You insert a new student named “Robert’); DROP TABLE students; --” and you keep your job

SQLi Technique



The SQLi attack typically works by prematurely terminating a text string and appending a new command

Because the inserted command may have additional strings appended to it before it is executed the attacker terminates the injected string with a comment mark “- -”

Subsequent text is ignored at execution time

SQLi Attack Avenues

User input

Attackers inject SQL commands by providing suitable crafted user input

Server variables

Attackers can forge the values that are placed in HTTP and network headers and exploit this vulnerability by placing data directly into the headers

Second-order injection

A malicious user could rely on data already present in the system or database to trigger an SQL injection attack, so when the attack occurs, the input that modifies the query to cause an attack does not come from the user, but from within the system itself

Cookies

An attacker could alter cookies such that when the application server builds an SQL query based on the cookie’s content, the structure and function of the query is modified

Physical user input

Applying user input that constructs an attack outside the realm of web requests

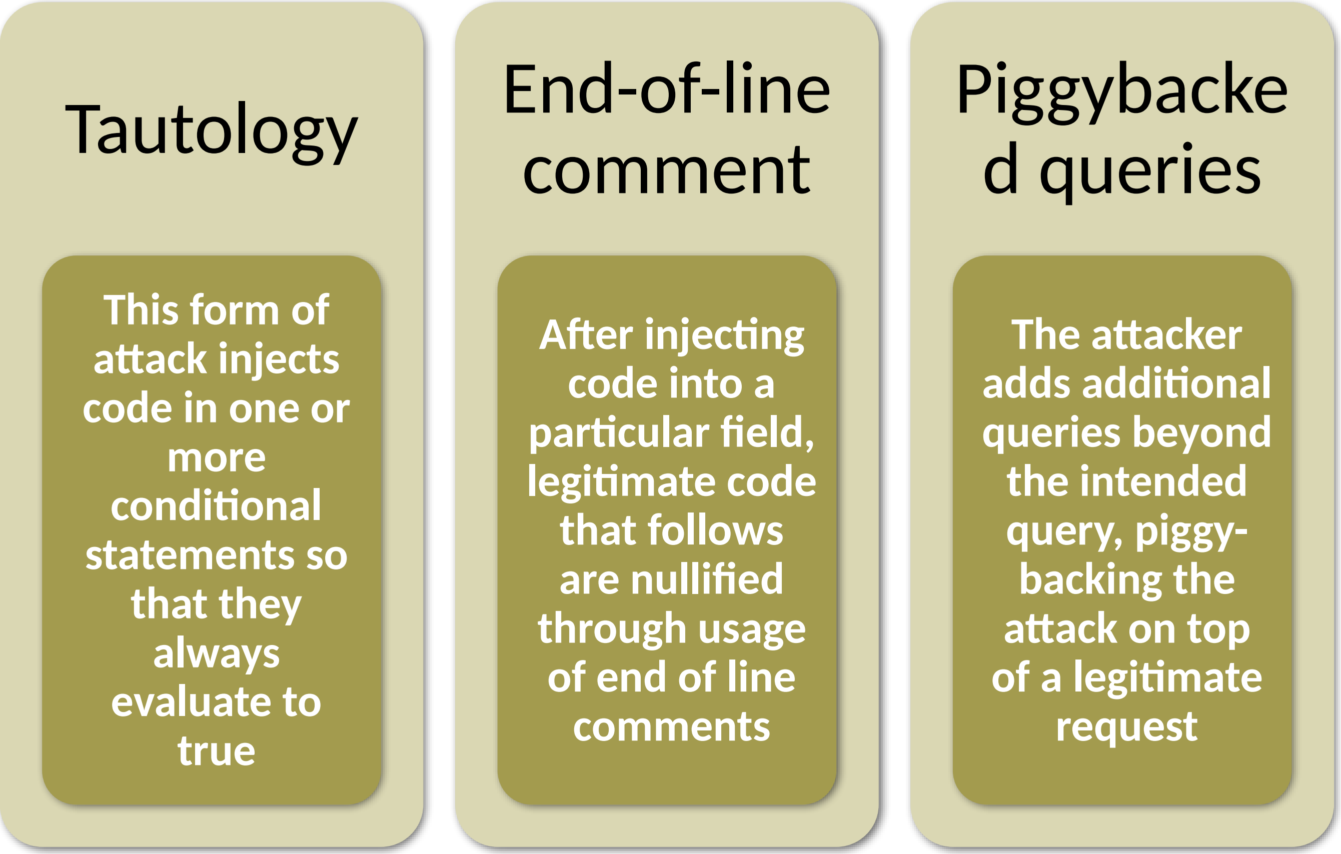
SQLi Interactive Tutorial

https://www.hacksplaining.com/exercises/sql-injection#/first-login-attempt

Inband Attack

Uses the same communication channel for injecting SQL code and retrieving results

- The retrieved data are presented directly in application Web page



Tautology

This form of attack injects code in one or more conditional statements so that they always evaluate to true

End-of-line comment

After injecting code into a particular field, legitimate code that follows are nullified through usage of end of line comments

Piggybacked queries

The attacker adds additional queries beyond the intended query, piggy-backing the attack on top of a legitimate request

Inferential Attack

There is no actual transfer of data, but the attacker is able to reconstruct the information by sending particular requests and observing the resulting behavior of the Website/database server

Include:

- Illegal/logically incorrect queries
  - This attack lets an attacker gather important information based on the server's error response about the type and structure of the backend database of a Web application
  - The attack is considered a preliminary, information-gathering step for other attacks
- Blind SQL injection
  - Allows attackers to infer the data present in a database system even when the system is sufficiently secure to not display any erroneous information back to the attacker

Inferential Attack – Blind SQLi Example

Given that a particular server behaves differently based on an SQL query’s returning of data, a blind SQLi attack can be made.

For example, if the server happens to return a message “Welcome” on a non-empty result of a query and no message on an empty result of a query, consider the next example.

For example, suppose there is a table called **Users** with the columns **Username** and **Password**, and a user called **Administrator**. We can systematically determine the password for this user by sending a series of inputs to test the password one character at a time.

To do this, we start with the following input into the **password field**:

xyz' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) > 'm

This returns the "Welcome back" message, indicating that the injected condition is true, and so the first character of the password is greater than m.

Next, we send the following input:

xyz' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) > 't

This does not return the "Welcome back" message, indicating that the injected condition is false, and so the first character of the password is not greater than t.

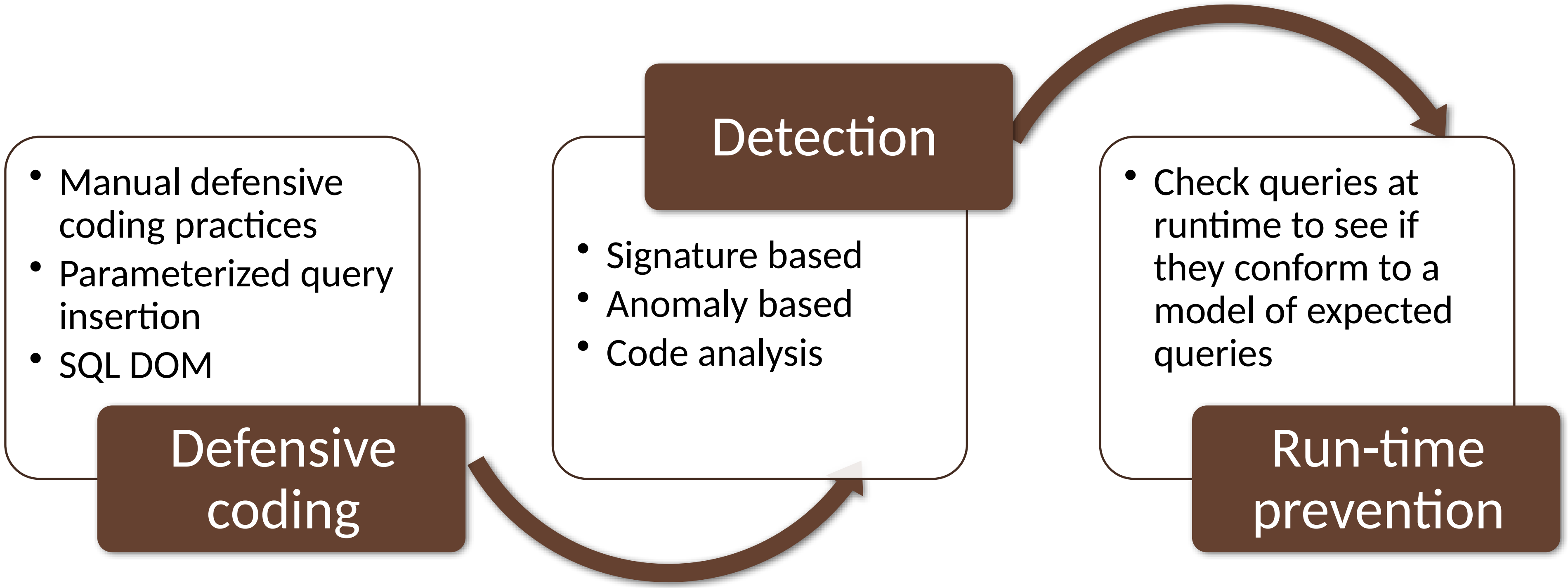
Eventually, we send the following input, which returns the "Welcome back" message, thereby confirming that the first character of the password is s:

xyz' AND SUBSTRING((SELECT Password FROM Users WHERE Username = 'Administrator'), 1, 1) = 's

Out-of-Band attack

- Data are retrieved using a different channel (not the same channel as SQLi)
- Not very common
- This can be used when there are limitations on information retrieval, but outbound connectivity from the database server is weak
- Example – Relies on database server’s ability to make outbound DNS or HTTP calls to the attacker’s server, delivering data (e.g. Oracle DB UTL_HTTP package, Microsoft SQL Server xp_dirtree)

SQLi Countermeasures



Defensive coding

- Manual defensive coding practices
- Parameterized query insertion
- SQL DOM

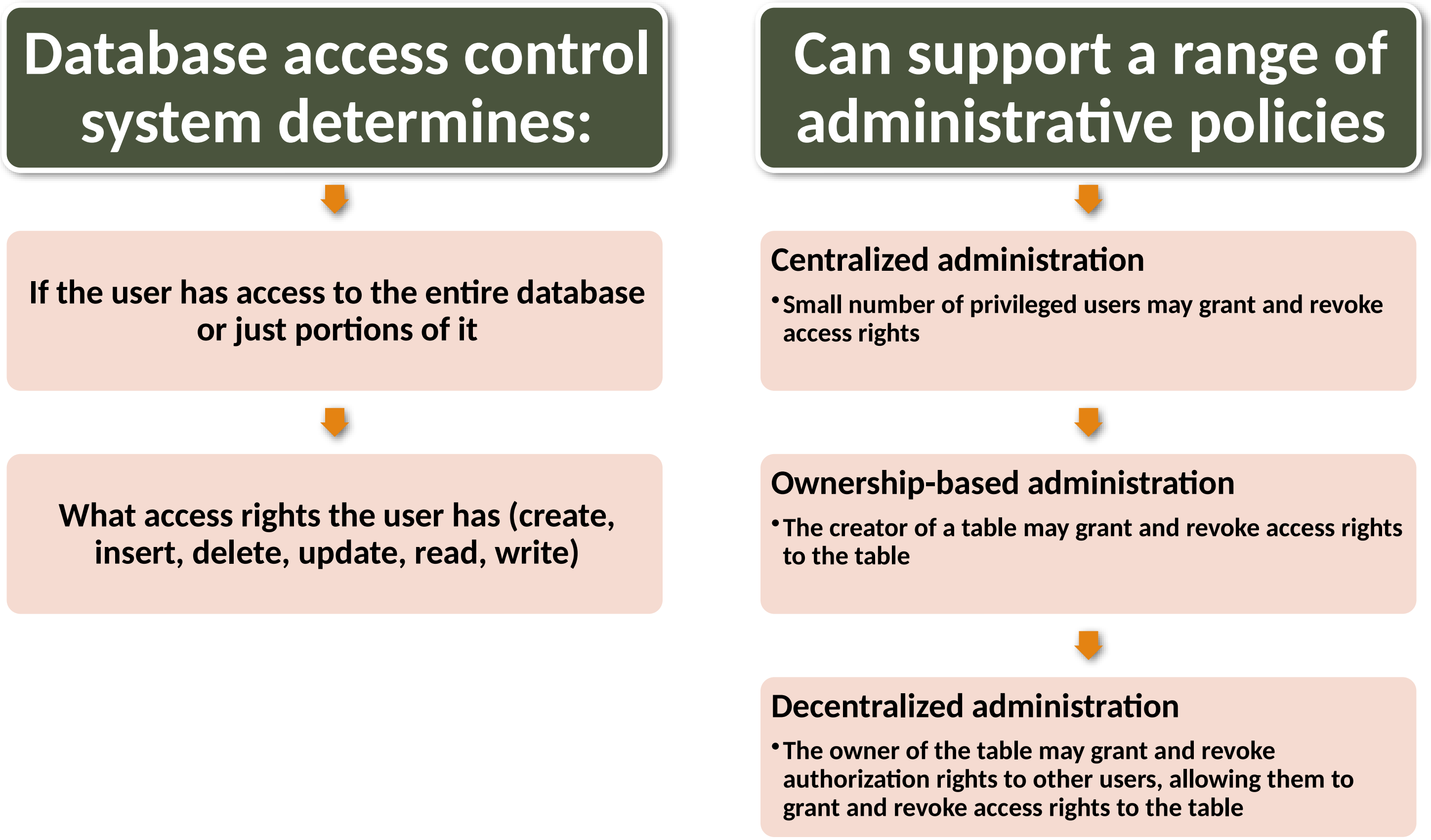
Detection

- Signature based
- Anomaly based
- Code analysis

Run-time prevention

- Check queries at runtime to see if they conform to a model of expected queries

Database Access Control



Database access control system determines:

If the user has access to the entire database or just portions of it

What access rights the user has (create, insert, delete, update, read, write)

Can support a range of administrative policies

Centralized administration

- Small number of privileged users may grant and revoke access rights

Ownership-based administration

- The creator of a table may grant and revoke access rights to the table

Decentralized administration

- The owner of the table may grant and revoke authorization rights to other users, allowing them to grant and revoke access rights to the table

SQL Access Controls

Two commands for managing access rights:

- Grant
  - Used to grant one or more access rights or can be used to assign a user to a role
- Revoke
  - Revokes the access rights

Typical access rights are:

- Select
- Insert
- Update
- Delete
- References

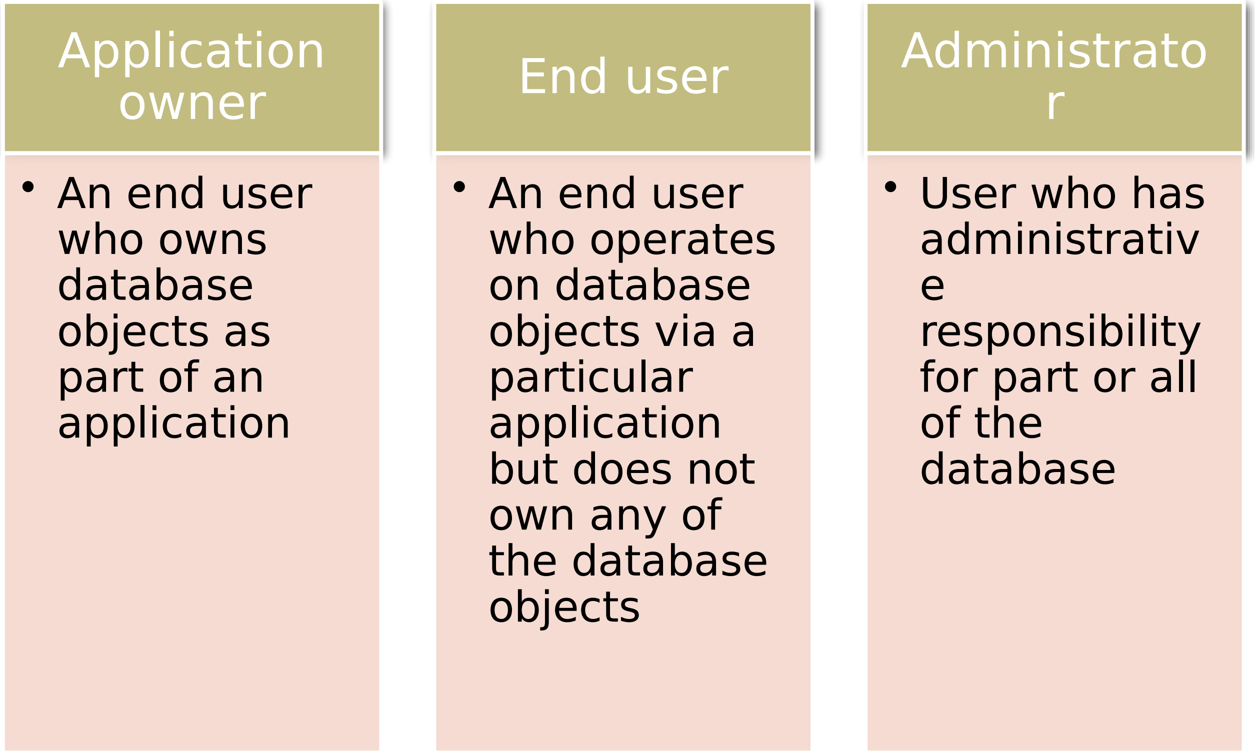
Role-based Access Control (RBAC)

Role-based access control eases administrative burden and improves security

A database RBAC needs to provide the following capabilities:

- Create and delete roles
- Define permissions for a role
- Assign and cancel assignment of users to roles

Categories of database users:



Application owner

- An end user who owns database objects as part of an application

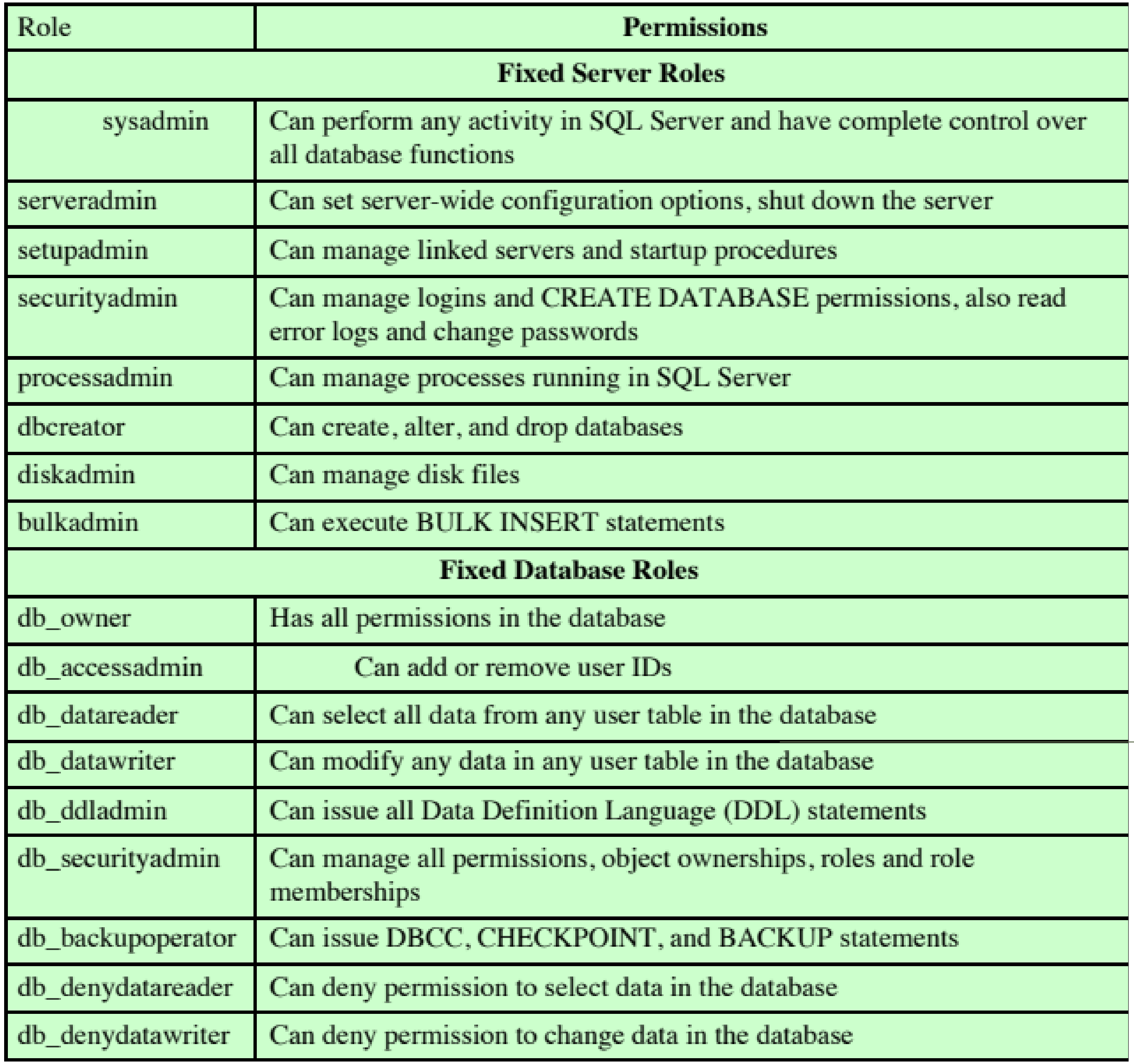
End user

- An end user who operates on database objects via a particular application but does not own any of the database objects

Administrator

- User who has administrative responsibility for part or all of the database

Fixed Roles in Microsoft SQL Server



Role	**Permissions**
Fixed Server Roles	
sysadmin	Can perform any activity in SQL Server and have complete control over all database functions
serveradmin	Can set server-wide configuration options, shut down the server
setupadmin	Can manage linked servers and startup procedures
securityadmin	Can manage logins and CREATE DATABASE permissions, also read error logs and change passwords
processadmin	Can manage processes running in SQL Server
dbcreator	Can create, alter, and drop databases
diskadmin	Can manage disk files
bulkadmin	Can execute BULK INSERT statements
Fixed Database Roles	
db_owner	Has all permissions in the database
db_accessadmin	Can add or remove user IDs
db_datareader	Can select all data from any user table in the database
db_datawriter	Can modify any data in any user table in the database
db_ddladmin	Can issue all Data Definition Language (DDL) statements
db_securityadmin	Can manage all permissions, object ownerships, roles and role memberships
db_backupoperator	Can issue DBCC, CHECKPOINT, and BACKUP statements
db_denydatareader	Can deny permission to select data in the database
db_denydatawriter	Can deny permission to change data in the database

Database Encryption

The database is typically the most valuable information resource for any organization

- Protected by multiple layers of security

Firewalls, authentication, general access control systems, DB access control systems, database encryption

Encryption becomes the last line of defense in database security

- Can be applied to the entire database, at the record level, the attribute level, or level of the individual field

Disadvantages to encryption:

- Key management

Authorized users must have access to the decryption key for the data for which they have access

- Inflexibility

When part or all of the database is encrypted it becomes more difficult to perform record searching

Database Encryption - Depicted

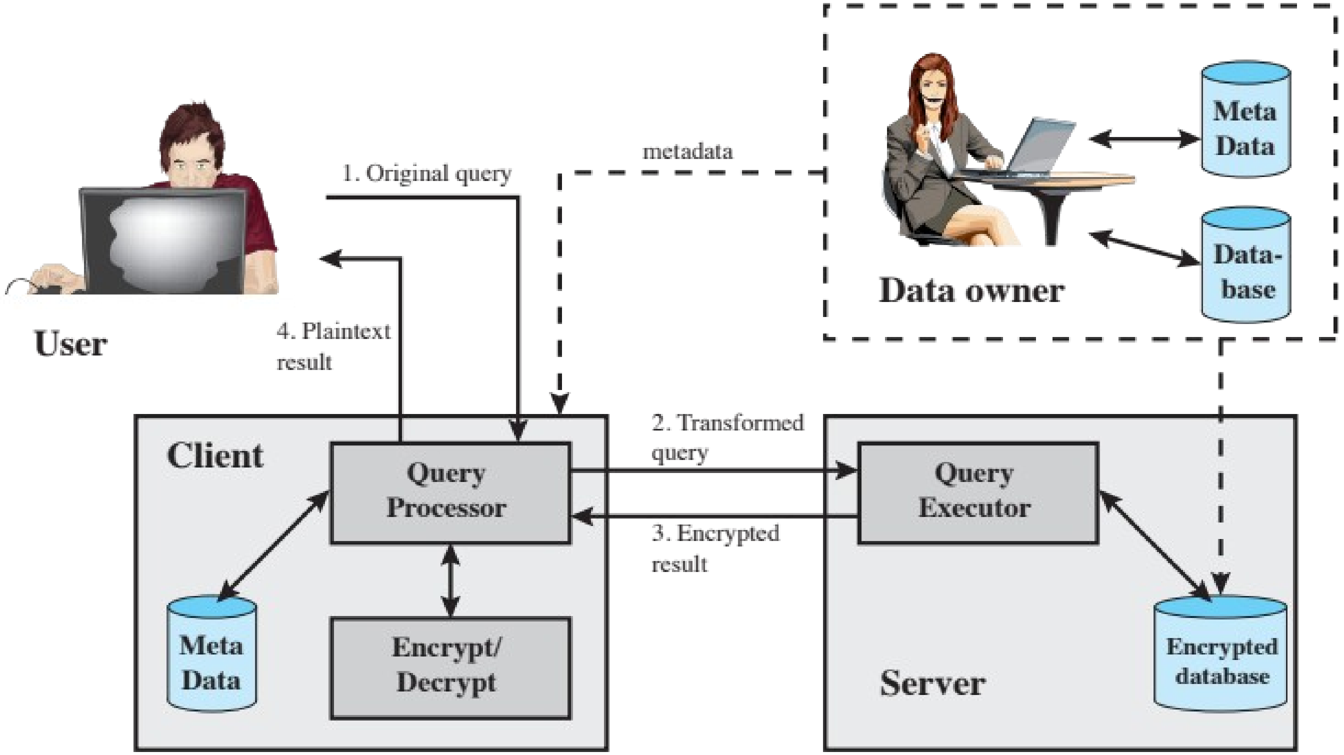


Figure 5.9 A Database Encryption Scheme

Data Center Security

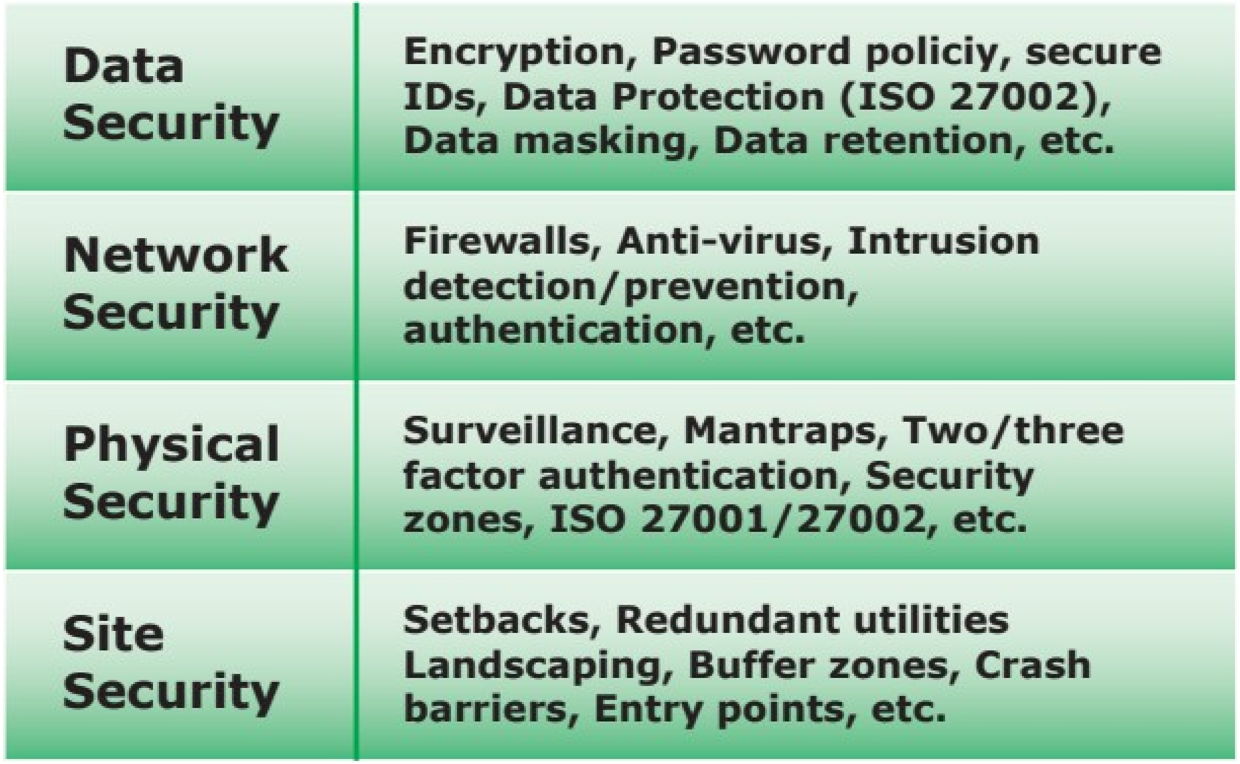
Data center:

- An enterprise facility that houses a large number of servers, storage devices, and network switches and equipment
- The number of servers and storage devices can run into the tens of thousands in one facility
- Generally includes redundant or backup power supplies, redundant network connections, environmental controls, and various security devices
- Can occupy one room of a building, one or more floors, or an entire building

Examples of uses include:

- Cloud service providers
- Search engines
- Large scientific research facilities
- IT facilities for large enterprises

Data Center Security



Data Security	Encryption, Password policiy, secure IDs, Data Protection (ISO 27002), Data masking, Data retention, etc.
Network Security	Firewalls, Anti-virus, Intrusion detection/prevention, authentication, etc.
Physical Security	Surveillance, Mantraps, Two/three factor authentication, Security zones, ISO 27001/27002, etc.
Site Security	Setbacks, Redundant utilities Landscaping, Buffer zones, Crash barriers, Entry points, etc.

Figure 5.12 Data Center Security Model