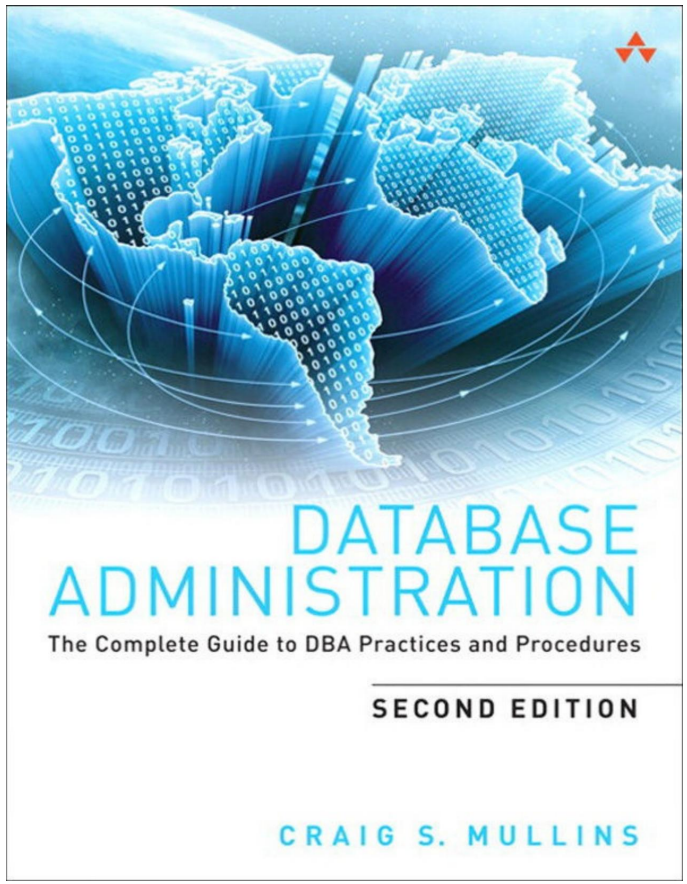
Database Administration

Lecture 08: Database Security.

Mullins, 2013

17 de marzo de 2025

Database Administration: Database Security.



Content has been extracted from *Database Administration: The Complete Guide to DBA Practices and Procedure (Chapter 14)*, by Craig Mullins, 2013. Visit https://www.oreilly.com/library/view/database-administration-the/9780133012743/.j

Overview

- Database security is crucial due to increasing data breaches and regulations.
- Data breaches are costly and prevalent.
- Databases are a prime target for hackers.
- Securing data requires a comprehensive plan.

Data Breaches

- Data breaches continue to rise, with significant impact.
- Over 544 million records were breached between 2005 and 2012.
- Average cost per lost record is between $90 and $305.
- Costs include legal fees, customer losses, and bad publicity.
- Database servers are the main source of breached data.

Database Security Basics

- DBMS controls all database resources.
- Users must be granted privileges to perform operations.
- DBA is typically responsible for database security.
- Key aspects of database security:
  - Authentication (Who is it?)
  - Authorization (Who can do it?)
  - Encryption (Who can see it?)
  - Audit (Who did it?)

Authentication

- Strong authentication is essential.
- Logins (user IDs) with passwords are required.
- Information needed for login creation:
  - Password
  - Default database
  - Default language
  - User's full name
  - Additional details (e.g., email, phone number)
- Passwords should be changed regularly.

Password Guidance

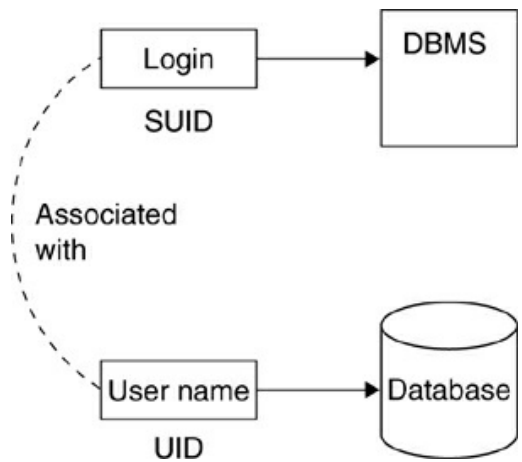
- Passwords should be difficult to guess.
- Guidelines:
  - Minimum 6 characters, longer if possible.
  - Use a combination of alphabetic and numeric characters.
  - Avoid complete words.
  - Do not embed personal information.
  - Consider concatenating unrelated words.
  - Use mnemonic devices, but not obvious ones.
  - Avoid common and weak password archetypes.
- Work with security administration to create and distribute guidelines.

Login Administration

- Drop logins when no longer needed.
- Limit database users who can create database objects to DBAs only.
- Lock logins that may need to be reactivated.
- Drop logins that will never need to be reactivated.

Database Users

- Some systems require an additional account to use specific databases.
- Login (SUID): Accesses the DBMS or database server.
- User name (Database UID): Associated with the login account, required to access each database.
- Guest usage can be permitted by configuring a special user name.



Granting and Revoking Authority

- DBA controls security using Data Control Language (DCL).
- DCL statements:
  - GRANT: Assigns permission to a database user.
  - REVOKE: Removes permission from a database user.
- GRANT statement requires a list of privileges and users.
- WITH GRANT OPTION allows a user to pass the authority to grant privileges.
- Avoid issuing GRANT and REVOKE statements from within an application program.
- Some DBMSs provide additional functionality (e.g., DENY in SQL Server).

Types of Privileges

- Basic types: access data, create objects, perform system functions.
- Common privilege types:
  - Table: Access and modify data within tables.
  - Database object: Create and drop database objects.
  - System: Perform system-wide activities.
  - Program: Create, modify, and use database programs.
  - Stored procedure: Execute specific functions and stored procedures.

Granting Table Privileges

- Control access to tables, views, and columns.
- Privileges:
  - SELECT: Enable user to select from table/view.
  - INSERT: Enable user to insert rows.
  - UPDATE: Enable user to update rows/columns.
  - DELETE: Enable user to delete rows.
  - ALL: Enable user to select, insert, update, and delete.
- Column-level privileges can be specified for SELECT and UPDATE.
- DBA typically grants table privileges in test environments.
- Production access should be controlled using program and stored procedure privileges.

Granting Database Object Privileges

- Control permissions to create database structures.
- Privileges depend on the DBMS and supported object types.
- Options to grant CREATE privileges on various objects (e.g., databases, tablespaces, tables, indexes).
- Ability to create objects is usually reserved for DBAs.
- Uncontrolled object creation can lead to difficulties in tracking and managing database objects.

Granting System Privileges

- Control which users can use DBMS features and commands.
- Privileges vary from DBMS to DBMS.
- Examples: archive logs, shut down/restart server, start traces, manage storage and caches.
- System privileges are granted system-wide, not at the database level.
- Should be granted with care, usually reserved for DBA and SA.

Granting Program and Procedure Privileges

- EXECUTE privilege allows users to execute programs or stored procedures.
- Granting privileges on programs/procedures is easier to control than on individual tables/columns.
- Procedural logic controls which tables and columns can be modified.
- DBA can better maintain data integrity by controlling changes programmatically.

Granting to PUBLIC

- Granting authorization to PUBLIC allows anyone who can log in to the DBMS to have that authority.
- Grants to PUBLIC cannot be given with the WITH GRANT OPTION.
- Using PUBLIC can appear easier but requires caution.
- DBA loses control over the object or resource when a privilege is granted to PUBLIC.
- Reserve PUBLIC usage for objects/resources that should be available to everyone.
- PUBLIC authority can be a shortcut if another security mechanism is in place (e.g., transaction processor security).

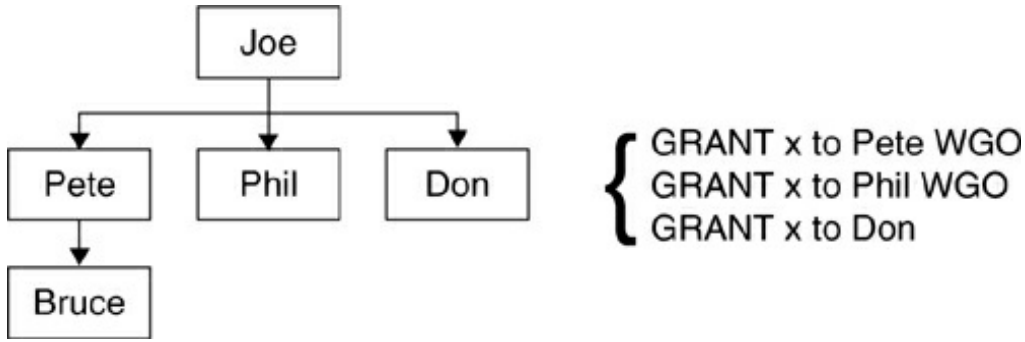
PUBLIC Authority and the System Catalog

- Dilemma: Whether to grant PUBLIC access to system catalog tables.
- Metadata in the system catalog is useful for DBAs, developers, and analysts.
- System catalog may contain sensitive information exploitable by hackers (e.g., authorization details, user IDs, login information).
- Exposes table names, making it easier for hackers to attempt unauthorized access or SQL injection.
- Refrain from granting blanket PUBLIC access to the system catalog tables.

Revoking Privileges

- REVOKE statement removes previously granted privileges.
- Syntax is the ”flip side.of the GRANT syntax.
- Privileges are automatically revoked when a database object is dropped.
- Revoking a PUBLIC privilege does not remove it from users granted it in a separate GRANT statement.

Cascading REVOKEs



- When a revoke causes the DBMS to revoke additional related privileges.
- Example: Joe grants privilege X to Pete and Phil (with GRANT option), Pete grants X to Bruce, Joe grants X to Don (without GRANT option).
- If X is revoked from Joe, it will also be revoked from Pete, Phil, Don, and Bruce (cascading effect).
- To minimize impact, avoid granting privileges using the WITH GRANT OPTION.
- Some DBMS products allow revoking without cascading.

Chronology and Revokes

- Timing of GRANT or REVOKE can affect impact.
- Example: Grant DELETE on titles to public; COMMIT; REVOKE DELETE on titles from userx;
- In some DBMSs (e.g., Microsoft SQL Server), this sequence bars userx from deleting, even though PUBLIC was granted DELETE.
- Some DBMSs (e.g., DB2) do not permit such exclusions; PUBLIC overrides revokes.
- DBA must understand how GRANTs and REVOKEs work for each DBMS.

Label-Based Access Control (LBAC)

- Need for low-level access control is critical as systems become more sophisticated.
- LBAC secures each piece of data, ensuring only authorized users can perform authorized functions.
- Supports applications needing a more granular security scheme.
- Can specify who can read and modify data in individual rows and/or columns.

LBAC Details

- Administrator configures LBAC by creating security label components.
- Security policy, composed of label components, describes criteria for data access.
- Security administrator defines policy by creating security labels.
- Security label can be associated with columns and rows to protect data.
- Users are granted access by granting them security labels.

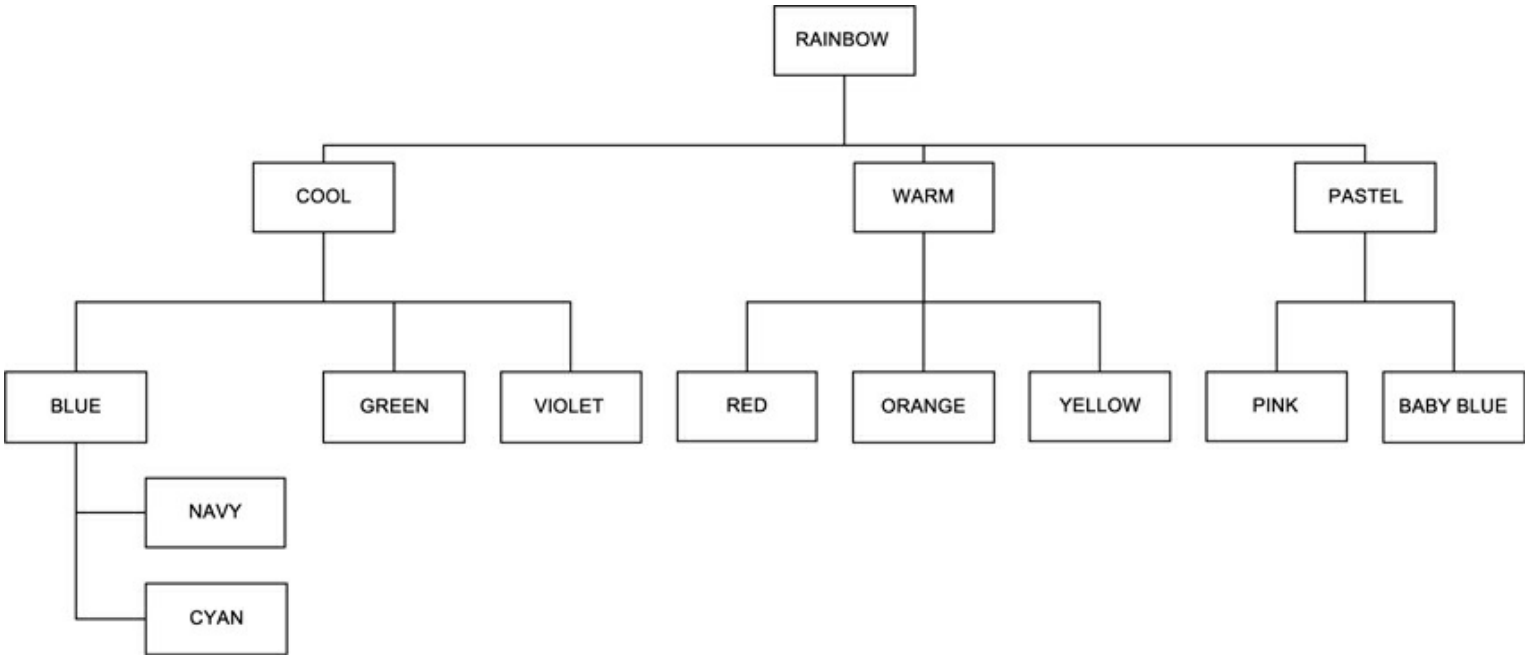
LBAC Process

- When a user tries to access protected data, their security label is compared to the label protecting the data.
- Protecting label blocks some security labels but not others.
- Users can hold labels for multiple policies but at most one label for read and one for write access per policy.
- Exemptions can be granted to users, allowing access to data their labels might otherwise prevent.
- Security labels and exemptions are called LBAC credentials.

LBAC Implementation

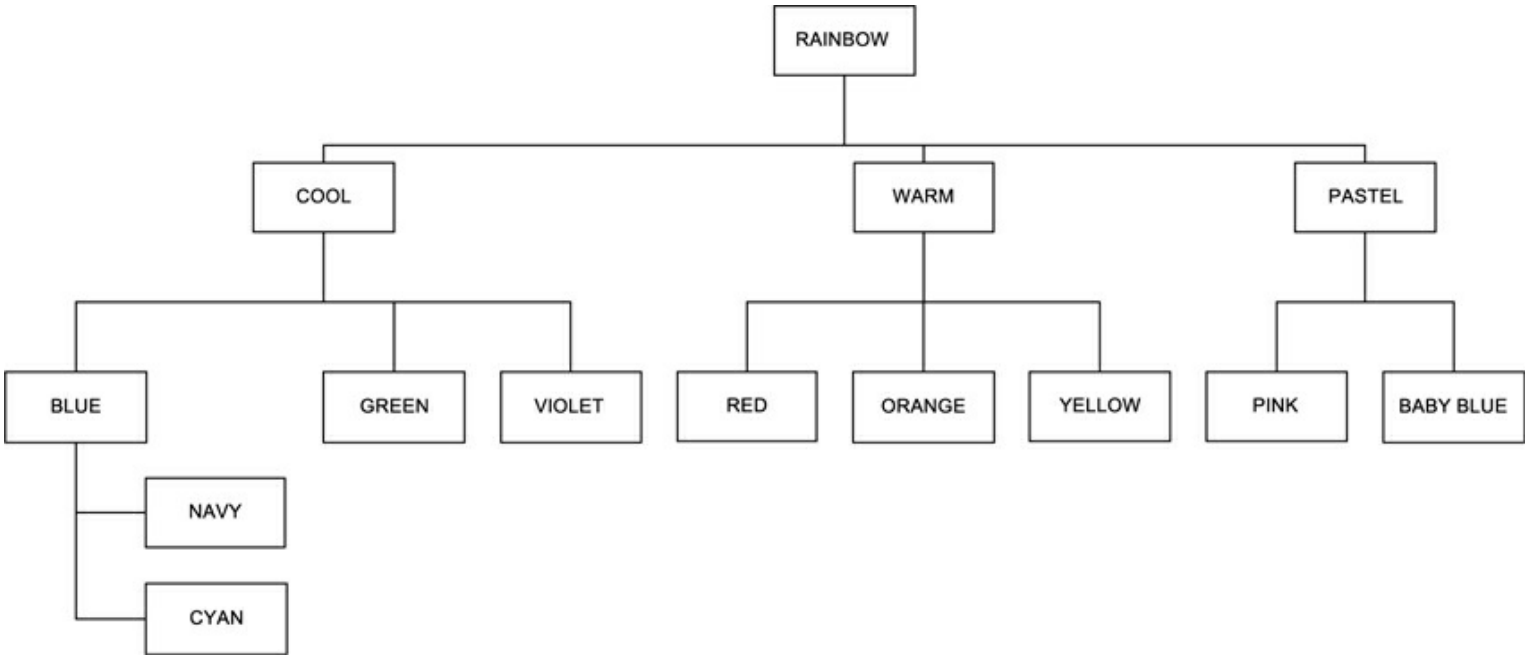
- Attempted access to a protected column fails if LBAC credentials do not permit it.
- If users try to read protected rows not allowed by their credentials, the DBMS acts as if those rows do not exist.
- Aggregate functions (e.g., COUNT(*)) ignore rows when LBAC credentials do not allow read access.
- Requires adding a specially named column to act as the security label.
- Security label column is matched with the multilevel security hierarchy set up by the administrator.

LBAC Hierarchy Example



- Hierarchy example uses colors (RAINBOW, COOL, WARM, PASTEL, BLUE, GREEN, VIOLET, NAVY, CYAN).
- Hierarchy need not be complex (e.g., TOP SECRET, SECRET, UNCLASSIFIED).

LBAC Hierarchy Example



- Hierarchy is established in security software (e.g., RACF with DB2 for z/OS).
- RAINBOW label includes everything.
- Middle levels (e.g., COOL, WARM, PASTEL) represent additional groupings.

LBAC Flexibility and Control

- Hierarchy provides flexibility for assigning various security levels to data.
- Assigning a security label column and populating it controls whether a user can access a particular row.
- RAINBOW makes data accessible to anyone.
- NAVY is more restrictive.
- BLUE data can access BLUE rows and subordinate blue shades (NAVY and CYAN).

Security Reporting

- DBA needs to monitor and report on user privileges.
- Database security is maintained in the system catalog.
- SQL can be used to retrieve information from system catalog tables.
- Some DBMSs provide views and system-stored procedures for easier retrieval.
- Protect the security of the system catalog, especially in production.

Security Reviews

- User security requirements evolve over time.
- Database security needs to change as new applications are added and business requirements change.
- Regular security reviews are necessary to ensure implemented security matches current requirements.
- Reports from system catalog tables can provide input for reviews.

Authorization Roles and Groups

- DBMS may allow assigning users specific privileges to a role or built-in groups of privileges.
- Terminology varies among DBMSs (roles vs. groups).
- DBA needs to understand how each DBMS implements roles and groups to simplify security administration.

Roles

- Role: Collection of privileges.
- DBA creates a role and assigns privileges to it.
- Role can be assigned to one or more users, simplifying security administration.
- Example:

```
CREATE role MANAGER; COMMIT;
GRANT select, insert, update, delete on employee to MANAGER;
GRANT select, insert, update, delete on job_title to MANAGER;
GRANT execute on payroll to MANAGER; COMMIT;
GRANT MANAGER to user1; COMMIT;
```

- Additional users can be assigned the `MANAGER` role without needing to issue individual `GRANT` statements.

Groups

- Group-level authority is similar to roles.
- DBMS provides built-in groups that cannot be changed.
- Each DBMS implements group-level security differently with different group names and privileges.
- Similarities exist across DBMSs.

Common Groups

- System administrator (SA or SYSADM): Most powerful, can execute all commands and access all objects.
- Database administrator (DBADM or DBA): All privileges over a specific database, can access but not modify data.
- Database maintenance (DBMAINT): Privileges for maintaining database objects (e.g., utilities and commands).
- Security administrator: Privileges for granting/revoking security, login/password administration, auditing, security configuration.
- Operations control (OPER or SYSOPR): Authority to perform operational tasks like backup/recovery and terminating tasks.

Group Management

- Limit the number of users with SA role or group-level authority.
- SA capabilities are very powerful; only corporate DBAs and systems programmers should have this authority.
- Group-Level Security and Cascading REVOKEs:
  - Some group-level privileges allow users to grant privileges to others.
  - Revoking group-level authority from such a user also revokes any privileges they granted.
  - Similar to cascading REVOKEs with the WITH GRANT option.

Using Views for Security

- Views can simplify some aspects of database security.
- Example: Employee table with sensitive information (salary, etc.).
- Granting SELECT privilege on the table can cause a security problem.
- A view can be created to omit sensitive information.
- Users can be granted SELECT on the view to access appropriate information.

Vertical Restriction with Views

- Example View:

```
CREATE VIEW emp_all AS
SELECT
    first_name, last_name, middle_initial,
    street_address, state, zip_code
FROM
    employee;
```

- Only information specified in the view can be retrieved.
- Eliminating columns from a base table to create a view is vertical restriction.
- Definition of sensitive data varies by organization.

Plan

Stored Procedures for Security

- Stored procedures enhance security by restricting direct access to base tables.
- Execution privilege must be explicitly granted or revoked.
- Ensures controlled data access through procedural logic.

Logic-Oriented Security

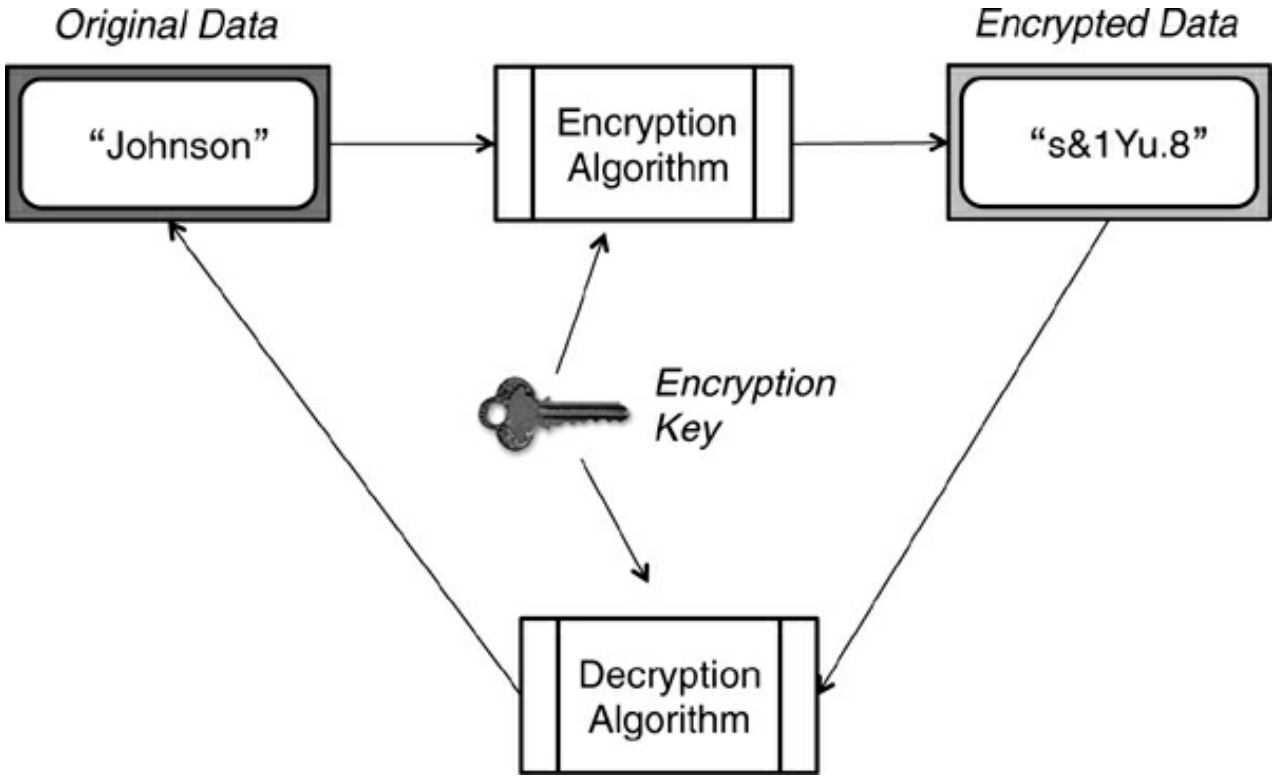
- Implement security based on conditions such as user identity and time of access.
- Prevent unauthorized data manipulation.
- Example: Restrict access to specific hours.

Plan

Encryption Overview

- Protects data by transforming it into unreadable formats without decryption keys.
- Two primary types: **Data at Rest** and **Data in Transit** encryption.
- Used to comply with security regulations and prevent unauthorized access.

Encryption and Decryption



Types of Encryption

- Data at Rest: Protects stored data from unauthorized access.
- Data in Transit: Ensures secure data transfer over networks.
- Common encryption techniques: AES, DES, and SHA hashing.

Plan

SQL Injection Attacks

- A technique used by attackers to execute unauthorized SQL commands.
- Exploits improper input validation in web applications.
- Can lead to data leaks, unauthorized modifications, or even database destruction.

Preventing SQL Injection

- Validate all user inputs.
- Use prepared statements and parameterized queries.
- Restrict database permissions to limit exposure.
- Avoid concatenating user input in SQL statements.

Examples of SQL Injection Attacks

- In 2006, Russian hackers stole credit card data from a Rhode Island government website.
- In 2007, a hacker defaced Microsoft's UK website using SQL injection.
- In 2008, tens of thousands of PCs were infected via an automated SQL injection attack targeting Microsoft SQL Server.

More SQL Injection Cases

- In 2010, a Swedish voter attempted an SQL injection attack in a write-in vote.
- In 2011, the Public Broadcasting System (PBS) was hacked using SQL injection.
- Other victims: Barracuda Networks, MySQL.com, Lady Gaga's website, Nokia, and Canon.

Plan

Database Auditing

- Tracks database activities to detect unauthorized access.
- Helps ensure compliance with security policies.
- Logs who accessed data, what was changed, and when.

Threats to Security

- Internal threats from disgruntled employees.
- External cyber-attacks targeting sensitive information.
- Auditing helps in identifying and mitigating such threats.

Plan

External Security Mechanisms

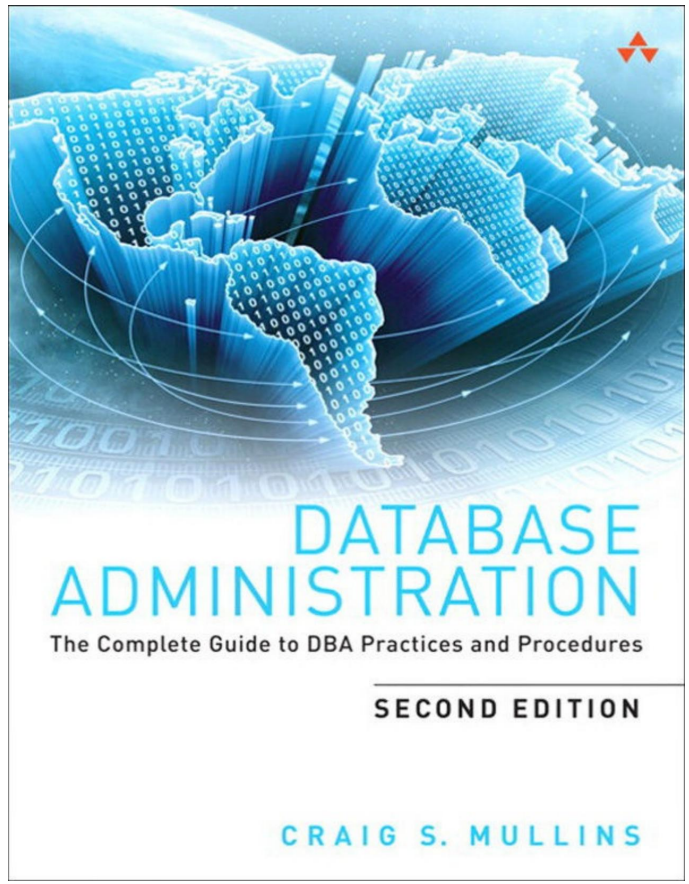
- Protect database files at the operating system level.
- Use firewalls, access controls, and encryption.
- Prevent unauthorized access outside DBMS control.

Plan

Summary

- Database security is critical to protect sensitive data.
- Use stored procedures, encryption, and auditing to enhance security.
- Prevent SQL injection and secure external resources.

Database Administration: Database Security.



Content has been extracted from *Database Administration: The Complete Guide to DBA Practices and Procedure (Chapter 14)*, by Craig Mullins, 2013. Visit https://www.oreilly.com/library/view/database-administration-the/9780133012743/.