
Database Security

Database Security Issues

- Types of Security
  - Legal and ethical issues
  - Policy issues
    - Governmental, institutional, corporate levels
  - System-related issues
    - Physical hardware, OS or DBMS levels
  - Need to identify multiple security levels

Database Security Issues

- Threats to databases
  - Loss of **integrity**
    - Improper modification of data
  - Loss of **availability**
    - Legitimate users cannot access data
  - Loss of **confidentiality**
    - Unauthorized disclosure of confidential data

Database Security Issues

- To protect databases against these types of threats four kinds of countermeasures can be implemented:
  - **Access control**
    - Create user accounts and passwords
  - **Inference control**
    - Ensure information about individuals cannot be accessed
  - **Flow control**
    - Prevent information from flowing to unauthorized users
  - **Encryption**
    - Protect sensitive data at rest and during transmission

Database Security and the DBA

- Database administrator (DBA)
  - Central authority for administering database system
  - Superuser or system account
- DBA-privileged commands
  - Account creation
  - Privilege granting
  - Privilege revocation
  - Security level assignment

Access Control, User Accounts, and Database Audits

- User must log in using assigned username and password
- Login session
  - Sequence of database operations by a certain user
  - Recorded in system log
- Database audit
  - Reviewing log to examine all accesses and operations applied during a certain time period

Sensitive Data

- Sensitivity of data
  - Inherently sensitive
  - From a sensitive source
  - Declared sensitive
  - A sensitive attribute or sensitive record
  - Sensitivity in relation to previously disclosed data

Sensitive Data

- Factors in deciding whether it is safe to reveal the data
  - Data availability
    - Not available when being updated
  - Access acceptability
    - Authorized users
  - Authenticity assurance
    - External characteristics of the user
    - Example: access only allowed during working hours

Sensitive Data

- A tradeoff between precision and security
- Precision
  - Protect all sensitive data while making available as much nonsensitive data as possible
- Security
  - Ensuring data kept safe from corruption and access suitably controlled

Information Security and Privacy

- Concept of privacy goes beyond security
  - Ability of individuals to control the terms under which their personal information is acquired and used
  - Security a required building block for privacy
- Preventing storage of personal information
- Ensuring appropriate use of personal information
- Trust relates to both security and privacy

Database Security Issues

- Discretionary security mechanisms
  - Used to grant privileges to users
- Mandatory security mechanisms
  - Classify data and users into various security classes
  - Implement security policy
- Role-based security

Discretionary Access Control

- Two levels for assigning privileges to use a database system
  - Account level
    - Example: CREATE SCHEMA or CREATE TABLE privilege
  - Relation / Table level

Discretionary Access Control

- Relation / Table level
  - Each relation R assigned an owner account
  - Owner of a relation given all privileges on that relation
  - Owner can grant privileges to other users on any owned relation
    - Retrieval/Read (SELECT) privilege on R
    - Modification privilege on R
    - References privilege on R

Privileges through use of Views

- Consider owner A of relation R and another user B
  - A can create view V of R that includes only attributes A wants B to access
  - Define V as a SELECT query that only shows tuples which B needs access to
  - Grant read-only access on V to B

Mandatory Access Control

- Additional security policy that classifies data and users based on security classes
- Typical security classes
  - Top secret
  - Secret
  - Confidential
  - Unclassified

Discretionary vs. Mandatory

- DAC policies have a high degree of flexibility
  - Do not impose control on how information is propagated
- Mandatory policies ensure high degree of protection
  - Rigid
  - Prevent illegal information flow

Role-Based Access Control (RBAC)

- Permissions associated with organizational roles
  - Users are assigned to appropriate roles
- Can be used with traditional discretionary and mandatory access control
- Identity management
  - To effectively authenticate people and manage their access to confidential information
- Temporal constraints on roles

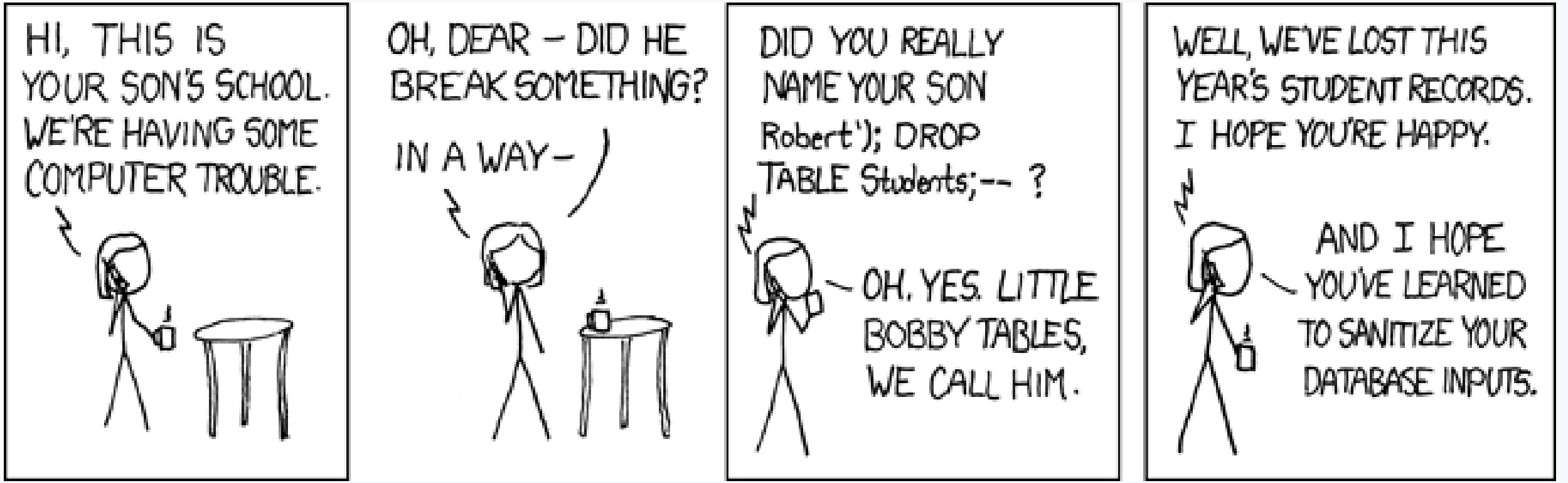
Row-Level Access Control

- Sophisticated access control rules implemented by considering the data row by row
- Each row given a label
  - Used to prevent unauthorized users from viewing or altering certain data
- Label security policy
- Provides finer granularity of data security

Web and Mobile Applications

- E-commerce environments require elaborate access control policies
  - Go beyond traditional DBMSs
- Legal and financial consequences for unauthorized data breach
- Content-based access control
  - Policies take the protection object content into account


SQL Injection


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR - DID HE BREAK SOMETHING?
IN A WAY-
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

What is SQL injection?

- Attacker injects a string input through the web application
- Changes/manipulates the application generated SQL statement to the attacker’s advantage

SQL Manipulation

- Change the SQL command in the application
  - Adding conditions to the WHERE clause of a query

```
SELECT * FROM users WHERE username = 'jake' and PASSWORD =
    'jakespasswd'.
```

```
SELECT * FROM users WHERE username = 'jake' and (PASSWORD =
    'jakespasswd' or 'x' = 'x')
```

Code Injection

- Add additional SQL statements or commands to existing SQL statement by exploiting bugs
  - Patch systems in a timely manner

Function Call Injection

- A database or OS function call inserted into a vulnerable SQL statement

```
SELECT TRANSLATE ('user input', 'from_string', 'to_string') FROM dual;
```

```
SELECT TRANSLATE (" || UTL_HTTP.REQUEST ('http://129.107.2.1/') || ",
    '98765432', '9876') FROM dual;
```

Risks associated with SQL injection

- Database finger printing
- Denial of Service
- Bypass authentication
- Identify injectable parameters
- Execute remote commands
- Privilege escalation

Protection techniques

- Bind variables
  - parameterize statements

```
PreparedStatement stmt = conn.prepareStatement( "SELECT * FROM
    EMPLOYEE WHERE EMPLOYEE_ID=? AND PASSWORD=?");
stmt.setString(1, employee_id);
stmt.setString(2, password);
```

Protection techniques

- Input validation / Sanitize the input
  - By filtering input, remove escape characters from input strings with *Replace* function
  - Single quote delimiter ‘ replaced by ‘’
  - Define good data for input
    - Strip out bad stuff – quotes, semicolons, escapes
  - Control type of file uploads

Protection techniques

- Function security
  - Restrict access for both standard and custom functions

Protection techniques

- Limit database permissions and segregate users
  - Web application must use a database connection with very limited rights
  - Only logged-in users have required rights to work with the database

Protection techniques

- Isolate the webserver
  - Keep the database server on a different host from the webserver
  - Keep them on separate subnets/network locations

Summary

- Major Database Security Issues
  - Privilege abuse
  - Weak authentication
  - Backup data exposure
  - SQL injection
  - DB platform vulnerabilities

Summary

- Fixes
  - Encryption
  - Levels of access control (Query, Content)
  - Strong Authentication
  - Firewall/IDS
  - Patch management