Database Security

COMP620

Database Value

• Database
  – Representing an essential corporate resource
  – Should be properly secured using appropriate controls

• Database Security
  – The mechanisms that protect the database against intentional or accidental threats
  – Protecting the database from
    • Unauthorized access
    • Modification
    • Destruction

Database Security: Before, Now, and Future

• A couple decades ago, databases were
  – Physically secure
  – Housed in central data centers – not distributed
  – External access mediated through customer service reps, purchasing managers, etc.
  – Security issues rarely reported

• Now increasingly DB’s externally accessible
  – Suppliers directly connected
  – Customers directly connected
  – Customers and partners directly sharing

• Data is the most valuable resource in applications
• DB security a growing problem

Privacy and Database Design

• Privacy
  – The right of individuals to have some control over information about themselves

• Laws
  – Many countries have laws designed to protect privacy
  – Every organization that collects and stores information about individuals is legally obliged to adopt policies that conform to local privacy legislation

• Database designers have a responsibility to protect the privacy of individuals about whose data is kept
### Legal Perspective

- “Database and Collections of the Information Misappropriation Act of 2003”
  - Limitation of Copyright laws
  - Protect Privacy
- “Sweat of the brow”
  - The effort expended in labor, and the value created thereby
  - Protecting large database from competitors

### Accidental Security Threats

- The user may unintentionally request an object or an operation for which he should not be authorized
- A system error might connect a user to a session that belongs to another user with different access privileges
- The operating system might not erase files that should be destroyed

### Deliberate Security Threats

- Impersonating an authorized user, or a user with greater access, by using his or her login and password
- Writing system programs with illegal code to bypass the database management system and its authorization mechanism
- Writing application programs to perform unauthorized operations
- Bribing, blackmailing, or influencing authorized users to obtain information from the database

### Database Application Questions

- Can the database be attacked through a web page?
- Can the data be tampered with?
- Are illegal operations traceable?
- Do error messages leak too much info?
- Can an attacker send script into your database?
Security Context

- A process or thread always executes in the context of an identity
  - May be the identity of the process itself
  - Or may be derived from user or client identity: impersonation (next slide)
  - And may flow between application or tiers: Delegation (next slide)

Impersonation and Delegation

Access Control

- Goal: make sure that data are accessed only in authorized ways
- Access Control Matrix
  - Specifying types of access permitted on a resource
- Responsibility of the DBA
  - Have a comprehensive view of users, roles, and resources
  - Design the access control matrix
  - Implement the access control matrix using the authorization language

Access Control Matrix

- A set of subjects $S$
- A set of objects $O$
- A set of privileges $R$
- One row for each subject
- One column for each object
- Elements are privileges of subject on an object
Example Access Control Matrix

<table>
<thead>
<tr>
<th>Subject (user)</th>
<th>Student (table)</th>
<th>StudentView (view)</th>
<th>Faculty (table)</th>
<th>Enrol (table)</th>
<th>Wrapup (procedure)</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>read, update</td>
<td>read</td>
<td>Read</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tom</td>
<td>read</td>
<td>read</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Faculty (role)</td>
<td>read</td>
<td>read</td>
<td>Read, insert, update, delete</td>
<td>execute</td>
<td></td>
</tr>
</tbody>
</table>

Privilege

- **Privilege**
  - A privilege is an action that a user is permitted to perform on database objects
    - Read
    - Update
    - Delete
    - Execute
    - Etc.

- **Granting privileges**
  - In standard SQL, the creator of an object (table, view, role, procedure, module, etc) is given all privileges in it
  - The creator can also pass these privileges on to others

SQL User Creation

- Create a user
  create user *username* identified by *password*
  default tablespace *tablespacename*
  temporary tablespace *temptablespacename*
- Example
  create user tom identified by tom
  default tablespace comp620project
  temporary tablespace temp;

Granting SQL Privileges

- **Granting Privileges Statement**
  GRANT (ALL_PRIVILEGES | privilege-list)
  ON (table-name | view-name)
  TO (PUBLIC | userlist | rolelist) [WITH GRANT OPTION];
- **Examples**
  - GRANT UPDATE ON branch TO tom;
    User Tom can update table branch.
  - GRANT UPDATE (assets) ON branch TO tom;
    User Tom can only update the asset column in the branch table.
  - GRANT UPDATE ON branch TO tom WITH GRANT OPTION;
    User Tom can update table branch and also grant it to other users
Revoking SQL Privileges

• Remove Privileges
  REVOKE {ALL PRIVILEGES | privilege-list}
  ON object-list FROM {PUBLIC | user-list | role-list}
  [CASCADE];

• CASCADE
  – If an individual has the grant option for a certain
  privilege and the privilege or the grant option on it is
  later revoked, all users who have received the
  privilege from that individual have their privilege
  revoked as well.

• Example
  REVOKE insert ON student FROM tom

Using Views for Access Control

• View
  – A widely used method for implementing access control
    • Hiding structures and data that the user should not see

• Example
  GRANT SELECT on Student to Tom;
  – Tom is a student and he can read other student records
  – Protect the Privacy of the student table
  – Create a view for Tom for his record
  CREATE VIEW TomView AS
    SELECT * FROM student WHERE name='Tom';
  Grant SELECT on TomView to Tom;

Role-Based Access Control

<table>
<thead>
<tr>
<th>Individuals</th>
<th>Roles</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Role 1</td>
<td>Table 1</td>
</tr>
<tr>
<td></td>
<td>Role 2</td>
<td>View 2</td>
</tr>
<tr>
<td></td>
<td>Role 3</td>
<td>Table 3</td>
</tr>
</tbody>
</table>

User’s change frequently, Roles don’t

SQL Roles

• Role
  CREATE ROLE rolename;
  – Example
    CREATE ROLE AdvisorRole;
    CREATE ROLE FacultyRole;

• Grant Privileges to Role
  GRANT SELECT ON Student TO AdvisorRole
  GRANT UPDATE ON Enroll TO AdvisorRole
  GRANT SELECT ON Enroll TO FacultyRole

• Assign a Role to a User
  GRANT AdvisorRole to Jack
Excessive Privilege Abuse

- When users (or applications) are granted database access privileges that exceed the requirements of their job
- Query-Level access control can be used to limit what a user can access
  - limit the SQL operations (SELECT, UPDATE, etc.)
  - limit access to tables, rows and columns

Platform Vulnerabilities

- Unless properly secured, it may be possible for a user to access the database files without using the database system
- Files can be copied, inspected or changed
- A separate database can be created from copies of the original database

Audit Trails

- Databases provide the ability to keep track of who did what
  - Regulatory Requirement - Sarbanes-Oxley (SOX) and the Healthcare Information Portability and Accountability Act (HIPAA) have audit requirements
- Deterrence
- Problem detection
- Recovery

Audit Weaknesses

- Lack of User Accountability – some audit trails do not record the exact user
- Performance Degradation – maintaining the audit trail consumes resources
- Limited Granularity – the audit trail might not record enough information
- Separation of Duties – Can a privileged user turn off the audit trail?
Denial of Service

• A user may be able to make many resource intensive requests of the database which can significantly reduce performance
• Users should be given only the capabilities necessary for their job

Backup File Exposure

• Organizations need to backup their databases on a regular schedule
• The backup data needs to be secure
• All backups should be encrypted