Basics of Database Security
(Spring Computer Science School)
INI, June 17-19 2006, Algeria

Nacer Boudjlida
LORIA, UHP Nancy 1 (F)
Nacer.Boudjlida@loria.fr
‘’Basics’’ of Database Security

- Technical and Organizational point of view:
  1. Protecting data against
     1. Unauthorized access
     2. Unauthorized operations
     3. Watching the database activities (Auditing)
  2. Operational Security: tend toward service continuity:
     1. Replication
     2. Recovery
“’Basics’” of Database Security: Outline

I. Preamble:
   i. DBMS
   ii. DB applications in C/S Architectures

II. Protecting Database Access

III. Operational Security and Recovery

IV. Operational Security Thanks to Replication

V. Database Auditing

VI. Concluding Remarks
Database Management Systems Features

1. Data Description ("objects", attributes, relationships, constraints)
2. (High Level, Set-Oriented) Data Manipulation
3. Consistency and Privacy
4. Concurrency Control
5. Security (and Recovery from failure)
DBMSs in C/S Architectures

- Accessing Data/Object Servers
  - SQL Queries
    - SQL Query
    - Query results
    - SQL Server
    - Data Base
  - Remote (compiled) procedures (Preferred)
    - Procedure call
    - [Results]
    - SQL Server
    - Stored Procedures
    - DB
Web Servers and Data/Object Servers

HTTP "over" TCP/IP

HTML and Documents

java

Groupware

Applications

Data Server

TP Monitor

HTML documents
Middleware: the Client-Server “glue”

General Middleware includes:

- Communication stacks
- Authentication services
- Remote procedure calls
- Message queues
- Distributed naming services
- etc.
DBMSs in C/S Architectures: Middleware

- **Specific Middleware:**
  - Database: ODBC, JDBC, DRDA, EDA/SQL
  - Transaction: TxRPC, Transactional RPC
  - Distributed Objects: CORBA, OLE/DCOM
  - Internet: HTTP, S(ecure)-HTTP
‘’Basics’’ of Database Security: Outline

1. Preamble:
   1. DBMS
   2. DB applications in C/S Architectures

2. Protecting Database Access

3. Operational Security and Recovery

4. Operational Security Thanks to Replication

5. Database Auditing

6. Concluding Remarks
Data Integrity and Privacy

- **Privacy:**
  - Encryption, Delusions
  - Login, Password,
  - Views
  - Privileges, Roles and Profiles

- **Identification Levels**
  1. Operating/Networking System
  2. DBMS
  3. [Database(s)]
‘’Basics’’ of Database Security: Outline

1. Preamble:
   1. DBMS
   2. DB applications in C/S Architectures

2. Protecting Database Access

3. Operational Security and Recovery

4. Operational Security Thanks to Replication

5. Database Auditing

6. Concluding Remarks
Operational Security and Recovery

- Ensure fast recovery from
  - Program failure
  - DBMS failure
  - Operating System failure
  - etc.

- Tightly Coupled with Concurrency Control

- Technical as well as Organizational Problem
Concurrency Control & Transaction Management

- Transaction: Atomic sequence of actions
- ACID Transaction Properties:
  - Atomicity
  - Consistency
  - Isolation
  - Durability  (Skip details ?)

- Recovery: Transaction logs (Transaction history)
  - *Undo*: “Play” the transaction backward (Rollback)
  - *Redo*: “Play” the transaction forward
Atomicity: Money transfer from C1 to C2
- Withdraw from C1 (C1 ← C1 - 100)
- Deposit into C2 (C2 ← C2 + 100)
- Problem after Withdrawal and before Deposit

<table>
<thead>
<tr>
<th>Transaction</th>
<th>Account C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read(C1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C1 ← C1 - 100</td>
<td>3100</td>
<td>5000</td>
</tr>
<tr>
<td>Write C1</td>
<td></td>
<td>3000</td>
</tr>
<tr>
<td>Read(C2)</td>
<td></td>
<td>5100</td>
</tr>
<tr>
<td>C2 ← C2 + 100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write(C2)</td>
<td>3000</td>
<td></td>
</tr>
</tbody>
</table>

Transaction Management (cont’d): Isolation Levels
Transactions (cont’d): Isolation Levels

- **Consistency**: Constraint \((x = y)\)

```
T1
X ← Read(x)
X ← X * 2
x ← Write (x)
Y ← Read(y)
Y ← Y * 2
y ← Write(Y)
```

```
T2
X ← Read(x)
X ← X + 20
x ← Write (X)
Y ← Read(y)
Y ← Y + 20
y ← Write(Y)
```

\[x \neq y\]
Transactions (cont’d): Isolation Levels

- **Isolation**: Withdrawal and Deposit (same account)

- T1: \( \{ X \leftarrow \text{Read}(C); X \leftarrow X - 200; C \leftarrow \text{Write}(X) \} \)

- T2: \( \{ Y \leftarrow \text{Read}(C); Y \leftarrow Y + 600; C \leftarrow \text{Write}(Y) \} \)

- Concurrent execution of T1 and T2 (Dirty Reads)
  - \( \times \) T1: \( X \leftarrow \text{Read}(C): \quad X \leftarrow 5000 \)
  - \( \times \) T1: \( X \leftarrow X - 200: \quad X \leftarrow 4800 \)
  - \( \times \) T2: \( Y \leftarrow \text{Read}(C): \quad Y \leftarrow 5000 \)
  - \( \times \) T2: \( Y \leftarrow Y + 600: \quad Y \leftarrow 5600 \)
  - \( \times \) T2: \( C \leftarrow \text{Write}(Y): \quad C \leftarrow 5600 \)
  - \( \times \) T1: \( C \leftarrow \text{Write}(X): \quad C \leftarrow 4800 \quad \text{(instead of 5400)} \)
Transaction (cont’d): Isolation Levels

- **Durability**: Permanent database updates

1. (Non) Repeatable Reads
   - $C$ in Database = 5000
     - $T_1$: Read($C$) $\rightarrow$ $X$: $X \leftarrow 5000$
     - $T_2$: Read($C$) $\rightarrow$ $Y$: $Y \leftarrow 5000$
     - $T_1$: $X \leftarrow X + 5000$: $X \leftarrow 10 000$
     - $T_1$: Write($X$) $\rightarrow$ $C$: $C \leftarrow 10 000$
     - $T_2$: Read($C$) $\rightarrow$ $Y$: $Y \leftarrow 10 000$

- **Different values !!!**
Transaction (cont’d): Isolation Levels

- Durability: Permanent database updates (cont’d)

2. (Non) Shadow tuples

- T1: Read(X) such that X = 500
- T2: Read(X) such that X = 500
- T1: Delete(X)
- T2: Read(X) such that X = 500

- No X, such that X = 500, exists anymore
Transaction (cont’d): Isolation Levels (end)

- Introduced in SQL’92
- **Level 0**: Dirty reads permitted
  - While a data item D is being updated by T1
  - Any other transaction can read D but it CANNOT modify it
- **Level 1**: Dirty reads not allowed
- **Level 2**: Reads repeatability not allowed before commit
- **Level 3**: Inhibits shadow tuples
Transaction Logging and Recovery

- Write Ahead Logging Protocol (WAL)
- Rollback: ‘Execute’ Transaction Log backward

Diagram:

<table>
<thead>
<tr>
<th>Transaction</th>
<th>DBMS</th>
<th>LOG</th>
<th>Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin T</td>
<td></td>
<td>&lt; T, Begin &gt;</td>
<td></td>
</tr>
<tr>
<td>X ← Read(C)</td>
<td></td>
<td>&lt; T, Read, C &gt;</td>
<td>Read(C)</td>
</tr>
<tr>
<td>X ← Val(C)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>X ← X + 500</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C ← Write(X)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commit T</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; T, Write, C, Val(C), X &gt;</td>
<td>C ← X</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt; T, Committed &gt;</td>
<td></td>
</tr>
</tbody>
</table>
Concurrency Control and Serialisability:
Concurrent execution is equivalent to sequential (serial) execution

Foundations: partial ordering of actions

Techniques

- Time stamping (not usually implemented)
- Two-Phase Locking (ensures Serialisability)
  - Phase1: Acquire Locks
  - Phase2: Free Locks
  - Does not prevent from deadlocks or livelocks
Most Usual Types of locks:
- Shared: multiple reads and no update
- EXclusive: single update
- Intention locks: Read, Write, Update

<table>
<thead>
<tr>
<th>Requested Lock Mode</th>
<th>S</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>X</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Lock mode</th>
<th>S</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>U</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>W</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

cf. Select … for update/at isolation level serializable/holdlock
Distributed DB:

DB = « Union » of « Fragments »
Note: Distributed Transaction Management

**Context:**
- Multiple sites cooperate in (sub-)transactions
- Commit/Rollback on every site

**Two-Phase Commit Protocol**
- Lampson [1976], Gray [1978]
- Most commonly implemented
- Ensures transaction **Atomicity** [Baer & al., ICSE81]
- Requires a coordination site
Distributed Transaction Management (cont’d)

- Coordination site:
  - Decomposes a transaction into sub-transactions depending on data location (cf. Query processing)
  - Sends the sub-transactions

![Diagram of Distributed Transaction Management]

- Validation Point
- Sites 1 and 2
- 1st phase
- 2nd phase
- Time

**Steps:**
- Prepare
- Ready
- Commit
- Committed
Transaction failure:
- One or more sites cannot commit
- One or more answers to the prepare message are missing...
Distributed Transaction Management (cont’d)

○ Transaction Logs: (Usually) WAL Protocol

Diagram showing the sequence of events in a distributed transaction management system, including log entries, subtransactions, and coordination logs.
Transaction Management (end)

- **Serialisability**: Serial execution equivalent to concurrent execution
- **ACID properties**
- **Mechanisms**:
  - Concurrency Manager
  - Integrity Manager
  - Recovery Manager
  - Programmers
  - Security Officer [+ DB Administrator]
Log Management and Recovery

- Log management:
  - Configuration
    7 Database creation
    7 During database « life »
  - Periodic Save
  - [Restore, when needed]
Log Management and Recovery

○ **Commit vs Checkpoint**
  - Data Server configuration *(checkpoint frequency)*

![Diagram of transaction, buffers, data, logs, base, and log checkpoint process]
Log Management and Recovery in Oracle

- **Checkpoint: configuration parameters**
  - \( \text{Log-checkpoint\_interval} \) (number of blocks)
  - \( \text{Log\_checkpoint\_timeout} \) (in seconds)

- **Loging**:
  1. Redo logs
  2. Rollback segments/Undo Tablespaces

- **Automatic Archive Logs**
Log Management and Recovery in Oracle

- Back-Up and Recovery
  - Total/Partial

- Recovery
  - Rollforward (cache recovery)
  - Rollbackward (transaction recovery)
Log Management and Recovery in Oracle

- Rollforward
- Rollbackward
- Redo Logs
- Rollback Segments
1. Preamble:
   1. DBMS
   2. DB applications in C/S Architectures

2. Protecting Database Access

3. Operational Security and Recovery

4. **Operational Security Thanks to Replication**

5. Database Auditing

6. Concluding Remarks
Security thanks to Replication

- Synchronize the content of a storage unit with its replicate
- Switch in case of failure
- Replicate
  - At least the logs
  - And/or sensitive data
- Requires more space
- Activity overhead (may conflict performance issues)
Security thanks to Replication

- Example: Oracle Multiplexing

![Diagram showing Oracle Multiplexing]

Groupe 1
- Log11.log
- Log12.log

LGWR
- 1
- 1'

Groupe 2
- Log21.log
- Log22.log

- 2
- 2'
Replication and Distribution

- Case of Replication Servers with primary copy
Replication and Distribution

- Case of Replication Servers with primary copy

Diagram:

- DB
- Log
- Replicated Sites
- Primary Site
- Log
- DB

Process:
- Begin T
- Update DB
- ... Commit T
- Replicated Sites
- Committed
- Committed

DB-Security-Basics, June 2006, Algiers, DZ

©Nacer.Boudjlida@loria.fr
Replication and Distribution

- Case of Replication Servers with primary copy
  - Case of primary site failure
    - 7 Elect another one
  - Case of secondary site failure
    - 7 Exclude it during the failure
    - 7 Re-synchronization (Logs)
‘’Basics’’ of Database Security: Outline

1. Preamble:
   1. DBMS
   2. DB applications in C/S Architectures
2. Protecting Database Access
3. Operational Security and Recovery
4. Operational Security Thanks to Replication
5. Database Auditing
6. Concluding Remarks
Auditing Services

- Record events (*audit options*) about
  - Database(s)
  - User(s)
  - Data Server(s)

- Process:
  1. Install or Configure the Audit Mechanism
  2. Define the audit options
  3. Manage the audit trail
Auditing Services in Oracle

1. **Install or Configure the Audit Mechanism**
   - `audit_trail = true` (init file)
   - `sys.aud$` exists

2. **Define the audit options**
   1. Per session/per access
   2. Whenever successful/not successful
   3. Per type of SQL statements
   4. Per type of system privileged commands
   5. Per type of action on database objects
Auditing Services in Oracle: Examples

- `audit` `select` on `U1.MaTable` whenever not successfull
- `audit` `select` table, `update` table by `U1, U2`
- `audit` role whenever successfull
- `audit` all privileges by `U1, U2`
- `audit` `insert`, `delete`, `update` on `sys.aud$` per access
- `noaudit` `select` table by `U1, U2`
Auditing Services in Oracle

3. **Manage the audit trail**
   
   a) Fix the right size (create/alter table)
   
   b) Explore
      
      - Dictionary tables and views
      - `dba_stmt_opts, all_def_audit_opts`, etc.
   
   c) Save or purge
Concluding Remarks

- DB security:
  - Tools and mechanisms exist
  - But, it’s not only a matter of DB technology
    - Platform
    - Network
    - Global enterprise security: policy and rules

- New challenges:
  - Ubiquitous/mobile computing
  - Replicated Mobile Data
  - Web-based computing
The (Very) Basics of Database Security

That’s all Folks!

Thank you for paying attention.

Any questions?