Sensitive Data and Database Inference

Tom Kelliher, CS 325

Apr. 14, 2010

1 Administrivia

Announcements

Collect assignment.

Assignment

Read 7.1.

From Last Time

Database security and reliability.

Outline

1. Sensitive data.
2. Sensitive data inference.
3. Availability and recovery issues.

Coming Up

Introduction to networks.

2 Sensitive Data

1. Sensitive data: Data within a database that should not be public.
2. What makes data sensitive?
   (a) Inherently sensitive: An individual's salary.

   (b) From a sensitive source: An informer whose identity must be kept secret.

   (c) Declared sensitive: An anonymous donor; Tom's ice cream consumption.

   (d) A sensitive *attribute* or sensitive *record*.

   Some data within a table might be sensitive — the salary field of a personnel database or the "Top Secret Flavor" row in Moxley's ice cream database.

   (e) Sensitive in relation to previously disclosed data — a partial recipe for Coca Cola.
3. Dilemma: Provide as much access as possible without compromising sensitive data.

   Another dilemma: security vs. precision.
4. Factors entering into access decisions:
   (a) Use — rows may be locked during a transactions, preventing access by other users.

   (b) Acceptability — a user may attempt to access sensitive data.

   What about access non-sensitive fields of rows in which other fields are sensitive? Generating a non-sensitive statistic from sensitive data?

(c) Role — a user may only be permitted access during working hours. The system may track previous queries, to ensure that a combination of queries doesn't reveal sensitive data.

(This doesn't address possible conspiracies.)

5. Types of disclosures:

   (a) Exact data. Tom earned $20.13 last year.

   (b) Bounds. Example: Professors earn between $100 and $1,000,000.

   (c) Negative result. Person X does *not* have 0 felony convictions.

   (d) Existence. The fact that a certain piece of data even exists can be sensitive. Example: The Math Department has an ice cream budget.

   (e) Probable value. Using a series of queries to establish a likely value for a sensitive piece of data.

3 Sensitive Data Inference

Deriving sensitive data from non-sensitive data.

1. Direct attack: Going directly for a sensitive data item.

   Querying a database for salary data.

   Possible to obscure a query using bogus conditions:

   ```
   SELECT salary
   FROM payroll
   WHERE (lname = 'Smith') OR (sex <> 'M' AND sex <> 'F');
   ```

2. Indirect attack: Derive sensitive data from non-sensitive statistics.

```
-- Assume there is only one record in payroll that has 'Segedy' in the
-- lname field.
SELECT sum(salary)
FROM payroll
WHERE lname <> 'Segedy';

SELECT sum(salary)
FROM payroll;
```

If this is still too overt, one can build a linear system of equations to produce the result using as many queries as necessary to fool the system.

3. Controlling the release of sensitive data.

   (a) Limited response suppression.

   "n-item k-response rule:" If a query returns n result rows and these rows represent k percent or more of the entire result, suppress those items from the entire result.

   This may not be enough.

   (b) Combined results: report various statistics.

   As we have seen, it can be possible to circumvent this.

   (c) Random sample. Construct a random sample of the database and run the query on this subset.

   Reduces precision.

   (d) Random data perturbation. "Tweak" the results.

   Maintenance of statistical properties?

   (e) Query analysis. Track the user's query history, using it to determine if sensitive data can be derived from the entire query set.

   Multiple difficulties.

   (f) Treat the database as if it were a class object and precisely define the queries that can run via your business logic.

4 Availability and Recovery Issues in Databases

1. A DBA's worst nightmare is a database crash or anything else that results in a corrupted database.

2. Some recovery techniques:

   (a) Checkpoint the database at regular intervals and maintain update logs.

   Take the database back to the last checkpoint and replay the updates.

   (b) Take regular backups.

   Restore from a backup.

3. Backup issues. Many databases need 24 × 7 availability.

   (a) Traditional backup software works at the filesystem level.

   Database must be quiescent for this to work.

   (b) Run the database's backup tool and then archive the script file it generates.

   This typically guarantees a consistent view of the database.

   (c) If the database is on a RAID 1 device (mirrored), idle the database momentarily, break the mirror, perform a traditional backup from the backup disk, and, finally, re-establish the mirror.