Querying Databases Privately

Johann-Christoph Freytag
(Oliver Berthold)
freytag@dbis.informatik.hu-berlin.de
http://www.dbis.informatik.hu-berlin.de

Talk @ Berkeley, CA, September 2005

Content of this talk

- Introduction & Motivation
- Algorithms & trade-offs
- summary
Motivation

Introduction

- Problem to solve:
  - User/Client: no one should know the contents of the query nor the result (not even the server)

- Observation:
  - Encrypting the communication between client and server might not be sufficient (Adversary might access decrypted query if he can get "inside" the database system and if he can observe disk access)

- Naïve solution:
  - Client downloads the entire DB & executes queries locally – unrealistic solution (size & ownership of data)
Private Database Access (PDA)

- Problem: Privately query data from a database
- Formalize:
  - Database is an array of \( N \) records.
  - (Private) query: "return record \( x \) from Database"
  - Nobody, even the database server, learns the value of \( x \).
- Using a "Secure Coprocessor" (SC)
- Simple Solution: Read entire Database per query \( \Rightarrow O(N) \)

![Database Server Diagram]

First improvement: Using a shuffled and encrypted database

- Private shuffling
  - \( O(N^2) \) if SC only can store one record.
  - Optimized: \( O(N \cdot \sqrt{N}) \) if SC store \( 1/\sqrt{N} \) part of \( \sqrt{N} \) records in each cycle

![Shuffled Database Diagram]
PDA – Perfect Privacy

- To answer first query by only accessing requested record ⇒ $O(1)$
- To answer query $i$ by accessing requested record plus records of query $i-1$ - otherwise adversary could exclude record of query $i-1$
- Perfect privacy: $x$ could be any record with the same probability
- After $\sqrt{N}$ queries reshuffling of database with $O(N^{1/2})$ needed
- Complexity per query: $O(N)$

PDA – Probabilistic Privacy (Asonov)

Security Parameters:
- $a$ ... # of (sequential) requests to shuffled & encrypted database; reshuffling after $N/b$ queries necessary
- $b$ ... # of random requests to original database (... includes queried record)

- Probability distribution
  - Each record of original database: $P = (1-a/N)/b$
  - Others: $P = (a/N)/(N-b)$
- No one record can be completely excluded from query
PDA – Probabilistic Privacy

- **Basic Idea**
  1. Request randomly \( c \) records per query including requested record
  2. Buffering of requested records
  3. reshuffle requested records

- **Result**
  - Requested record is stored
    - Either at one of the \( c \) positions with the same probability or
    - is still in the buffer with probability \( P = \frac{\text{poolsize}}{\text{poolsize} + 1} \)

PDA – Probabilistic Privacy (2)

- If an adversary performs his own queries he can only learn a probability distribution for records queried from the database
- Probability that record is still in the buffer for query \( j \):

\[
P_{R_{\text{44}}} = \left( \frac{\text{poolsize}}{\text{poolsize} + 1} \right)^{j} \quad P_{R_{\text{8}}} = \left( \frac{\text{poolsize}}{\text{poolsize} + 1} \right)^{j+1}
\]
• Often reshuffling/requesting of blocks achieves an equal probability distribution
• History-List: Subset of the \(c\) records (requested per query) randomly selected from records of previous queries
• Goal: similar probability for (randomly choosing) records requested in the past compared to probability of record being found in the buffer (at this position)

\[ p_{\text{history}} \leq \frac{1}{\text{poolsize} + 1} \]

\(\Rightarrow\) adversary doesn’t learn anything by observing such a query

Simulation: Adversary performs first query to record \(A\) followed by \(i\) user queries to records \(\neq A\).

\(\Rightarrow\) No significant probability differences between a query for record \(A\) or for any other record …

---

• First significant probability differences, if observed record outside of History (\(i > 383\))
• Probability difference – “success of adversary” - depends on settings for \(N\), \(c\), and pool size
Privacy Goals

- Higher \( c \) or Pool size lowers success of adversary
- Comparing Privacy – Complexity
- Using entropy to evaluate privacy:
  - \( P_i \) … Probability of query for record \( i \)
  - \( E \) … uncertainty of adversary’s observation

\[
E = - \sum_{i=1}^{N} P_i \cdot \log P_i
\]

Summary

- Private access to database
  - With different algorithms varying in degree of privacy
    - Complete privacy
    - Probabilistic privacy
  - Measure of privacy by Entropy
- Not mentioned
  - Implementation (prototypes) exist
  - Simulation model for probabilistic approach exists