Ensuring Correctness over Untrusted Private Databases

Sarvjeet Singh, Sunil Prabhakar
Department of Computer Sciences, Purdue University, West Lafayette, Indiana, USA
sarvjeet@cs.purdue.edu, sunil@cs.purdue.edu

Motivation

- Consider food supply chain made of multiple entities: farms, distribution centers, retailers
- Each entity would like to prevent others from learning details of its operations
- What happens if a packet of beef sold at a given store is found to be contaminated?
- Current solutions to this problem are manual involving paperwork (i.e. slow)
- Automated solution is preferred
  - How to guarantee that results are correct?

- Common problem with virtually any situation where mutually distrusting entities need to exchange some data while preserving the privacy of the rest of the database
- Recent regulations (e.g. Sarbanes Oxley) also impose constraints on handling of private data
- Solutions that can provide guarantees of correctness of query results without exposing the entire database are desirable

Trusted Third Party

- Expensive solution in terms of volume of traffic and storage
- Third party is potential weakness and is now liable for privacy of data
- Such solutions will be resisted by privacy advocates due to too much of a “Big Brother” flavor

Model

- Both entities do not trust each other
- Bob is not willing to reveal anything other than the results of certain queries
- Alice wants a proof that query results are correct with respect to the committed database state
- No restriction on how the query results can be modified
- We do not trust Bob to follow the protocol honestly for proof generation

Issues

- Efficiency: Size of proof, Cost of proof generation, Size of verification object, Cost of verifying, Data exposure
- Attributes with small domain
- Granularity of hashing: Tradeoff between degree of exposure and generation cost

Overview

Querier (Alice)
Commit Database
Proof of Integrity
An event which causes Bob to become malicious
Query
Results
Verification Data

Database Owner (Bob)

Correctness

- α-correctness refers to correctness of result values (i.e. data is not tampered)
- β-correctness refers to correctness of query execution (i.e. query was executed correctly over the database)

Correctness

- Verification:
  - Incomplete result
  - Incorrect Selection
  - Incorrect Join

α-correctness:
- Bob sends Φ(A_i) for all i,j and S_1,j if applicable) for all attributes involved in selection to Alice

β-correctness:
- Selections and joins are correctly performed and all resulting tuples are returned

Merkle Tree

- A Merkle tree is a binary tree with labeled nodes:
  - Φ(parent) = h(Φ(left) || Φ(right))
  - Φ(leaf) = h(data content)

- Authenticity of any leaf node is ensured by:
  - Publishing the root’s label
  - Providing an authentication path for the leaf node

- For selections of the form A_i = a:
  - If Φ(A_i) = h(a_i || S_1,j) then tuple i should be present in the result

- For joins: if Φ(A^{(i)}) = Φ(A^{(j)}) then tuples i and j should be present in the result

Overhead of Proof Generation

Granularity of Hashing