Abstract

Privacy concerns in outsourced cloud databases have become more and more important recently and many efficient and scalable query processing methods over encrypted data have been proposed. However, there is limited work on how to securely process top-k ranking queries over encrypted data in the cloud. In this paper, we propose the first efficient and provably secure top-k query processing construction that achieves strong adaptively IND-CQA security. We develop an encrypted data structure called EHL and describe several secure sub-protocols under our security model to answer top-k queries. Furthermore, we optimize our query algorithms for both space and time efficiency. Finally, in the experiments, we empirically analyze our protocol using real world datasets and demonstrate that our construction is efficient and practical.

Top-k Query Processing

Finding k objects that have the highest overall grades.

Example:

<table>
<thead>
<tr>
<th>patient name</th>
<th>age</th>
<th>id</th>
<th>trestbps</th>
<th>chol</th>
<th>thalach</th>
</tr>
</thead>
<tbody>
<tr>
<td>E(Celeine)</td>
<td>E(43)</td>
<td>E(222)</td>
<td>E(120)</td>
<td>E(201)</td>
<td>E(160)</td>
</tr>
<tr>
<td>E(David)</td>
<td>E(60)</td>
<td>E(285)</td>
<td>E(100)</td>
<td>E(245)</td>
<td>E(142)</td>
</tr>
<tr>
<td>E(Flora)</td>
<td>E(43)</td>
<td>E(756)</td>
<td>E(100)</td>
<td>E(223)</td>
<td>E(127)</td>
</tr>
</tbody>
</table>

Table 1: Encrypted patients Heart-Disease Data

An authorized doctor, Alice, wants to get the top-k results based on some ranking criteria from the encrypted electronic health record database patients (see Table 1). The encrypted patients database may contain several attributes; here we only list a few in Table 1: patient name, age, id number, trestbps, chol, thalach.

Query: SELECT * FROM chol+thalach STOP AFTER k

Result: Enc(David), Enc(Emma)

No Random Access Algorithm (NRA)

1. Pre-computed lists over multiple attributes.
2. Combine scores by some monotonic aggregation function.

!!! Accesses modes: sorted access !!!

Secure Querying Algorithm

Algorithm 2 Top-k Query Processing: SecQuery

1. Sj receives Token from the client. Parse the Token and let Lj = Enc(Kj) for j ∈ M.
2. for each depth d at each list Lj do
3. for each E(Td)=(E(Encψ(E(Td))),E(Encψ(E(Td)))) ∈ Lj do
4. Compute E(W[Td]) = SecRank(E(Lj), E(Td)), where L = Enc(Lj).
5. Compute E(U[Td]) = SecBest(E(Lj), E(Td), μ).
6. Run Lj = SecUpdate([E(U[Td]]) with Sj and get the local encrypted list Lj.
7. Run Td = SecUpdate([E(U[Td]]) with Sj and get Td.
8. If |Td| ≤ k elements, go to the next depth. Otherwise, run EncSort(Td) by setting on E(W[Td]), get first k items as Td.
9. Let the (k+1)th item be E(U[Td]) and E(U[Td]), Sj then runs f = EncCompare(E(U[Td]), E(U[Td])) with St, where E(U[Td]) is the best score for E(U[Td]), and E(U[Td]) is the best score for E(U[Td]) in Td.
10. If f = 0 then
11. Halt and return the encrypted first k item in Td

Performance

(a) varying k
(b) varying m
(c) varying k
(d) varying m

Qry_F performance
Qry_E performance

(a) varying k
(b) varying m
(c) varying p

Communication Bandwidth

DataSet | Qry_F | Qry_E | Time per depth (milliseconds)
---|---|---|---
insurance | 2.3 | 4.8 | 29.4
Diabetes | 3.1 | 7.1 | 46.5
Pima | 3.3 | 7.1 | 54.1
Synthetic | 6.5 | 15.3 | 98.5

SecJoin (multiple tables, e.g t1, t2):
t2 = SELECT * FROM t1, t2 WHERE t1.l1 = t2.r1 ORDERED BY t1.l3 = t2.r1 + t2.r4 STOP AFTER k

Sec-Dedup (two tables, e.g t1, t2):
t2 = SELECT * FROM t1, t2 WHERE t1.id = t2.id | t2.id = Enc(t2.id) STOP AFTER k

Strong Security Guarantees

Xianrui Meng, Haohan Zhu, George Kollios
Boston University

Three Parties

Data Owner

Outsource

Cloud Server

Clients

Copyright © 2023 Boston University

Secure Access Algorithms (NRA)

1. Set-up phase: Enc(R), token + L
2. Query phase: Enc(R)
3. Leakage

"No efficient adversary can learn any partial information about the data or the queries, beyond what is explicitly allowed by the leakage functions."

"... even for queries that are adversarially-influenced and generated adaptively."

Theorem: The secure top-k query processing scheme SecTopK = (Enc, Token, SecQuery) we proposed above is IND-CQA secure if the key used in EHL is a pseudo-random function and the Paillier encryption is Chosen-Plaintext-Attack secure.