Supporting Complex Queries and Access Policies for Multi-User Encrypted Databases

Muhammad **Rizwan** Asghar, Giovanni Russello, Bruno Crispo, Mihaela Ion

The 5th ACM Workshop on Cloud Computing Security Workshop (CCSW) in conjunction with The 20th ACM Conference on Computer and Communications Security (CCS), Berlin, Germany November 8, 2013
Why Cloud Storage

- Cost saving
- Scalability
- Efficiency
- Availability
Problem

End-to-end confidentiality/privacy
The Proposed Approach

Honest but Curious

Add

Database Admin

Trusted

Insert

Query

Result

Authorised

Database User

Authorised

Database User

Authorised
Supported Queries

**Keyword**

SELECT *name* FROM *Personnel* WHERE *position*=manager

**Conjunction, Disjunction of Keywords**

SELECT *name* FROM *Personnel* WHERE *position*=manager AND *sex*=female OR *level*=2

**Complex Queries (e.g., range queries)**

SELECT *name* FROM *Personnel* WHERE *position*=manager OR *age*>30 AND *age*<40
State-of-the-Art Scheme Types

<table>
<thead>
<tr>
<th>Single User</th>
<th>Semi-Fledged Multi-User</th>
<th>Full-Fledged Multi-User</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
State-of-the-Art

|--------|--------------------|------------|-------------------------------|--------------------------|----------------------|------------------------|------------------|------------------|------------------|------------------|

<table>
<thead>
<tr>
<th>Single User</th>
<th>Semi-Fledged Multi-User</th>
<th>Full-Fledged Multi-User</th>
</tr>
</thead>
</table>

No Solution
Our Proposed Solution

- We propose full-fledged multi-user scheme that supports complex queries
- Our proposed mechanism protects queries and data stored in the cloud
- We enable enforcement of access control policies
- Users do not share any keys and each user has her own key
- A user can be removed without requiring re-distribution of keys or re-encryption of stored data
Solution Overview

- For data insertion and retrieval, we employ **Proxy Encryption (PE)**
  - Insert data \(\rightarrow\) PE
  - Retrieve data \(\rightarrow\) PE

- For making data *searchable*, we use **Keyword Encryption (KE)**
  - Insert data \(\rightarrow\) KE
  - Make query \(\rightarrow\) Trapdoor (TD)
  - Perform matching \(\rightarrow\) Match

- We protect access control policies using KE
Solution Details

Cloud Server

If (Match(KE(D), TD(Q) = true)

PE(D)

PE(D) = PEServer Re Enc(PE_i^*, K_{S_i})

TD(Q) = ServerTD(TD_j^*, K_{S_j})

User i

KE_i^* = KEUserEnc( D, K_{u_i})

PE_i^* = PEUserEnc( D, K_{u_i})

K_{S_i} = (i, x_{i2})

K_{S_j} = (j, x_{j2})

Insert

User j

TD_j^* = ClientTD(Q, K_{u_j})

D = PEDec(PE_j^*, K_{u_j})

Read

Key Management Authority

Master Secret Key = (x,)

Public Params = (...)

Random x_{i1}

x_{i2} = x - x_{i1}
Representation of Tables and Queries

SELECT name
FROM Personnel
WHERE position=manager AND level<3
Performance Evaluation: Insert

A prototype implemented in Java

Tested on Intel 2.67 GHz with 4 GB RAM, Microsoft Windows 7

Complexity: $\Theta(n + ms)$

$n$ is number of strings, $m$ is number of integers, each of size $s$ bits
Performance Evaluation: Select (Query)

Complexity: $\Theta(n + ms)$

$n$ is number of strings, $m$ is number of integers, each of size $s$ bits
Performance Evaluation: Select (Server)

Complexity: $O(r^*(n + ms^2))$

- $r$ is number of records
- $n$ is number of strings
- $m$ is number of integers
- Each of size $s$ bits
Performance Evaluation: Select (Decryption)

Complexity: $O(r^*(n + m))$

$r$ is number of records, $n$ is number of strings, $m$ is number of integers
Summary

- We presented the first multi-user scheme that supports complex queries in the cloud
- Each user has her own key
- A compromised user is revoked without requiring re-distribution of keys or re-encryption of stored data
- In future, we would like to
  - make performance improvements
  - build indexes
  - apply the technique on very large database
Thank You!

asghar@create-net.org

http://disi.unitn.it/~asghar/
## References


References (2)


References (3)


