Efficient Similarity Search over Encrypted Data

Mehmet Kuzu, Saiful Islam, Murat Kantarcioglu
Introduction

Client

Similarity Search over Encrypted Data

Selected Encrypted Items

Untrusted Server

Requires: Efficient and Secure
Similarity Searchable Encryption Protocols
Problem Formulation

- **BuildIndex(K, D):** Extract feature set for each data item in D and form secure index I with key K.
- **Trapdoor (K, f):** Generate a trapdoor for a specific feature f with key K and output T.
- **Search(I,T):** Perform search on I with trapdoor of feature f (T) and output encrypted collection C:

\[ C_j \in C \text{ if } \exists (f_i \in F_j) \ [dist(f_i, f) \leq \alpha] \]
\[ C_j \notin C \text{ if } \forall (f_i \in F_j) \ [dist(f_i, f) \geq \beta] \]
Locality Sensitive Hashing

- Family of functions is said to be $(r_1, r_2, p_1, p_2)$-sensitive if for any $x, y \in F$ and for any $h \in H$.
  
  - if $\text{dist}(x, y) \leq r_1$, then $\Pr[h(x) = h(y)] \geq p_1$
  
  - if $\text{dist}(x, y) \geq r_2$, then $\Pr[h(x) = h(y)] \leq p_2$

- A composite function $g: (g_1, \ldots, g_\lambda)$ can be formed to push $p_1$ closer to 1 and $p_2$ closer to 0 by adjusting the LSH parameters $(k, \lambda)$. 
Security Goals

- **Access Pattern** ($A_p$): Identifiers of data items that are in the result set of a specific query.
- **Similarity Pattern** ($S_p$): Relative similarity among distinct queries.

$$A_p(Q_1) = \{D_1, D_2, D_3\}, \quad A_p(Q_2) = \{D_2, D_3\}$$

$$S_p(Q_1, Q_2)$$

$$\begin{array}{c|ccc}
 Q_1 & & & \\
\hline
 1 & 0 & 0 \\
 0 & 0 & 0 \\
 0 & 1 & 0 \\
\end{array}$$
Secure LSH Index

- Content of any bucket $B_k$ is a bit vector ($V_{B_k}$):

$$V_{B_k}[id(D_z)] = 1 \quad \text{if } g_i(f_j) = B_k \text{ for } g_i \in g, f_j \in D_z$$
$$V_{B_k}[id(D_z)] = 0 \quad \text{otherwise}$$

- $[\text{Enc}_{id}(B_k), \text{Enc}_{payload}(V_{B_k})] \in I.$

\[\begin{array}{c|c|c}
D_1 & D_2 \\
B_1 & 1 & 1 \\
B_2 & 1 & 0 \\
B_3 & 1 & 1 \\
B_4 & 1 & 0 \\
B_5 & 1 & 0 \\
B_6 & 1 & 0 \\
B_7 & 0 & 1 \\
B_8 & 0 & 1 \\
B_9 & 0 & 1 \\
B_{10} & 0 & 1 \\
\end{array}\]

$$\text{Enc}_{id}(B_1) \quad 11\ldots$$
$$\text{Enc}_{id}(B_5) \quad 10\ldots$$
$$\text{Enc}_{id}(B_{10}) \quad 01\ldots$$
Secure Search Scheme

Shared Information

- $K_{\text{coll}}$: Secret key of data collection encryption
- $K_{\text{id}}, K_{\text{payload}}$: Secret keys of index construction
- $\rho$: Metric space translation function
- $g$: Locality sensitive function
Secure Search Scheme

- Trapdoor Construction for feature $f_i$:

$$T_{f_i} = \{Enc_{id}(g_1(\rho(f_i))), \ldots, Enc_{id}(g_\lambda(\rho(f_i)))\}$$

<table>
<thead>
<tr>
<th></th>
<th>$D_1$</th>
<th>$D_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B_1$</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$B_2$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$B_{87}$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>score</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Multi-Server Setting

- Basic search scheme reveals similarity and access patterns.
- It is desirable to separate leaked information to mitigate potential attacks.
- Multi-server setting enables lighter clients.
One Round Search Scheme

- This scheme is built on Paillier encryption that is semantically secure and additive homomorphic.

\[
\begin{align*}
    &\text{if } (\pi_S, \sigma_{V_S}) \in I, \text{ then } (\pi_S, [e_{S_1}, ..., e_{S_\ell}]) \in I' \\
    &e_{S_k} = E_{K_{pub}}(1) \text{ if } V_s[id(D_j)] = 1 \\
    &e_{S_k} = E_{K_{pub}}(0) \text{ otherwise}
\end{align*}
\]
One Round Search Scheme

- Bob performs homomorphic addition on the payloads of trapdoor components.

\[ \omega_{\text{score}}(i) = e_{t_1}(i) \odot \ldots \odot e_{t_\chi}(i) \]

\((i, \omega_{\text{score}}(i))\) pairs are sent to Charlie.
Error Aware Keyword Search

• Typographical errors are common both in the queries and data sources.
• In this context, data items be the documents, features be the words in the document and query feature be a keyword.
• Bloom filter encoding enables efficient space translation for approximate string matching.
Error Aware Keyword Search

• Elegant locality sensitive family has been designed for Jaccard distance (MinHash) that is \([r_1, r_2, 1-r_1, 1-r_2]\) sensitive.

\[
A = \rho(\text{john}) = \{1, 2, 4, 5, 7, 10\} \\
B = \rho(\text{john}) = \{1, 2, 4, 5, 6, 7, 10\}
\]

\[
J_d(A, B) = 1 - \frac{|A \cap B|}{|A \cup B|} \\
J_d(A, B) = 1 - \frac{6}{7} = 0.14
\]
Experimental Setup

- A sample corpus of 5000 emails is constructed from publicly available Enron e-mail dataset.
- Words in e-mails are embedded into 500 bit Bloom filter with 15 hash functions.
- (0.45, 0.8, 0.85, 0.01)-sensitive family is formed from MinHash to tolerate typos. Common typos are introduced into the queries 25% of the time.
- Default Parameters: (Number of documents: 5000, Number of features: 5000, $k$: 5, $\lambda$: 37).
• Ranking limits retrieval of irrelevant items.
Performance Evaluation (Single Server)

- Increase in $k$ and decrease in $\lambda$ have similar effects. Decrease in $\lambda$ leads smaller trapdoors.
• With increasing $n_d$, matching documents and the size of transferred bit vectors becomes larger.
Performance Evaluation (Multi-Server)

- Transfer of homomorphic addition results between servers is the main bottleneck.
Conclusion

• We proposed LSH based secure index and search scheme to enable fast similarity search over encrypted data.
• We provided a rigorous security definition and proved the security of the scheme to ensure confidentiality of the sensitive data.
• Efficiency of the proposed scheme is verified with empirical analysis.
THANKS …!

QUESTIONS?