Efficient Verifiable Dynamic Searchable Symmetric Encryption with a Tree-based Index
Kyohei Ozawa*, Ryuji Miyoshi**, and Hiroaki Yamamoto*
*Shinshu University, **Mitsubishi Electric Co., Japan

1. Introduction

Background and purpose: A searchable symmetric encryption (SSE) scheme is a method that searches encrypted data without decrypting it, and has been intensively studied. The purpose of the research is to develop an efficient and secure verifiable dynamic SSE (VDSSE) that works under a malicious server.

Contribution: We propose an efficient verifiable DSSE scheme that has the following properties: (1) the index consists of a tree-based structure forward/backward privacy, and is verifiable. That is, the proposed scheme works under a malicious server.

2. Comparison with Related Works

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Search Time</th>
<th>Index Size</th>
<th>FP</th>
<th>BP</th>
<th>VF</th>
</tr>
</thead>
<tbody>
<tr>
<td>MITRA¹</td>
<td>O(a&lt;sub&gt;op&lt;/sub&gt;)</td>
<td>O(N)</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kurosawa²</td>
<td>O(⌈D⌉)</td>
<td>O(1)</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Our Scheme</td>
<td>O(log&lt;sub&gt;p&lt;/sub&gt;+n&lt;sub&gt;L&lt;/sub&gt; + n&lt;sub&gt;L&lt;/sub&gt;</td>
<td>O(max{</td>
<td>EF&lt;sub&gt;0&lt;/sub&gt;</td>
<td>n&lt;sub&gt;L&lt;/sub&gt;, t&lt;sub&gt;0&lt;/sub&gt; + n&lt;sub&gt;L&lt;/sub&gt;})</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FP: Forward privacy; BP: Backward privacy (Type2); VF: Verifiable; W: a set of keywords; D: a set of documents; |W|: the total number of documents for w ∈ W; N: the total number of documents/keyword pairs; |D|: the number of keywords; μ: the number of updated documents/keyword pairs; n<sub>L</sub>: the number of documents matching the last search of w before time t; n<sub>L</sub>: the number of updates for w after the last search of w; τ<sub>0</sub>: the time when a keyword w is searched.

3. Proposed VDSSE Scheme

3.1. ID Tree

An ID tree for a keyword w ∈ W is a binary tree constructed from IDs of documents contain the keyword w. Each node is assigned a level and a node ID, and has a document ID, a tag for verification, a node address L to the left child, and a node address R for the right child. Fig.1 shows an example of an ID tree.

![Fig. 1: An ID Tree for keyword w with the number of documents 15](image)

3.2. Encrypted Index

Server-side index consists of document IDs and tags managed in the form of the ID tree. User-side index consists of the necessary data for search and verification for all keywords contained in the document. Fig.2 denotes data in a server-side index for a keyword “AI”. Each row has data stored in a node (see the top row in Fig.2), and L and R denote the addresses of the left and right children. The column Address is arranged for ease of viewing but is actually assigned in random. Fig.3 denotes a user-side index.

![Fig. 2: Server-side index](image)

3.3. Search Protocol

Fig. 4: Search diagram

1. User to Server: The user creates a trapdoor t = (T<sub>w</sub>, start, ht) for a search keyword w and sends it to the server.

2. Server to User: The server decrypts the sent data to get the IDs and tags, and sends the IDs to the user.

3. User to Server: The server sends the encrypted documents {c₁, ..., c<sub>n</sub>} corresponding to the IDs to the user.

4. Update Protocol

Update is an operation for addition or deletion of a document. Two operations are done as follows and the server cannot distinguish these: (1) for each keyword w of the updated document, make the encrypted data u = ((op, id, tag) ⊕ F<sub>sk₂</sub>(T<sub>u</sub>)[lev][nodeID]), L, R) ⊕ H(T<sub>w</sub>)[ht + 1](0) and send to the server, where if an operation is addition (deletion) then op = add (del). (2) The server puts u on top of the root of each ID tree as a new root. (3) The user updates the user-side index corresponding to the updated ID trees.

![Fig. 3: User-side index](image)

References

Implementation: JAVA, CPU: Xeon E3-1240 (4core) with 32GB

Dataset: Enron emails dataset (517,431 mails)