Flash Memory Summit

PRIVACY PRESERVING SEARCHABLE ENCRYPTION ON CONTENT INDEX

Vishwas Saxena
Senior Technologist, Western Digital

Background on Privacy Preserving Encryption Schemes

1. Homomorphic Encryption (HE)
2. Symmetric Searchable Encryption(SHE)

HE(Homomorphic Encryption): Traditional encryption schemes typically require decrypting data to perform computations, which exposes the sensitive information and may compromise privacy. In contrast, homomorphic encryption allows computations to be performed directly on the encrypted data, generating results that remain encrypted.

There are three kind of Homomorphic Encryption explained below:

Types	Actions	Number of Operations
Partially Homomorphic Encryption(PHE)	One(Addition or Multiplication)	Unlimited
Somewhat Homomorphic Encryption(SHE)	Two(Addition and Multiplication)	Limited
Fully Homomorphic Encryption(FHE)	Two(Addition and Multiplication)	Unlimited

FHE is the most ideal scheme with great security but is currently far from being practical.

SSE (Symmetric Searchable Encryption)

A symmetric searchable encryption (SSE) scheme is a cryptographic technique that allows searching encrypted data without decrypting it. It provides a way to securely search over encrypted data while preserving privacy and confidentiality.

Data (Documents) are encrypted and then stored in the cloud.

Search for any document(s) containing a keyword in the encrypted data, so that the cloud can not understand anything.

Search token is the encrypted keyword.

Documents containing the encrypted keyword are returned to the user.

User decrypts and use the document(s).

Leads to privacy preserving computation.

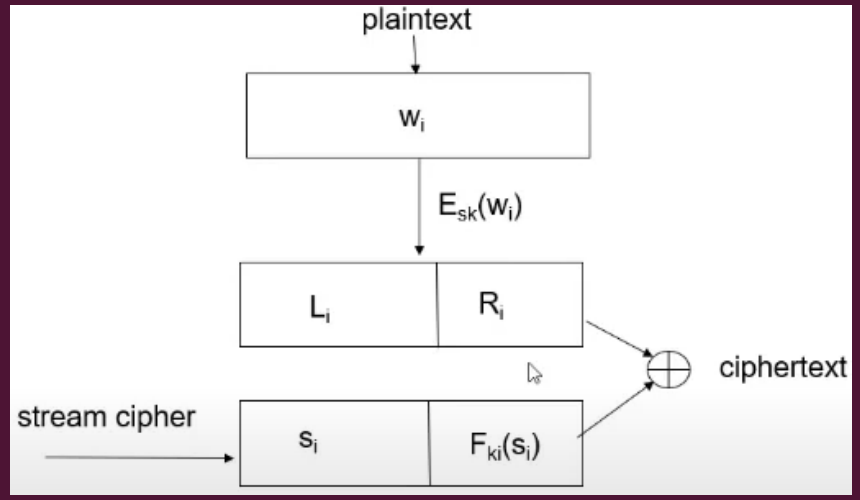
Sequential search and Reverse indexing

Sequential search and reverse indexing are two techniques used in the context of Symmetric Searchable Encryption (SSE) to enable efficient searching over encrypted data.

- **Sequential search (Song, D., Wagner, D., & Perrig, A. (2000)),** also known as linear search, is a basic search algorithm used to find a specific item within a collection of data. In the context of SSE, sequential search is applied to the encrypted data to locate matching encrypted documents or records.

- **Reverse indexing (Curtmola, R., Garay, J. A., Kamara, S., & Ostrovsky, R. (2006)),** also called inverted indexing, is a technique that helps improve search efficiency in SSE by creating an index that maps keywords or tokens to the corresponding encrypted documents. It involves creating a data structure that allows for quick lookup of documents containing specific keywords without having to sequentially search through all the encrypted documents.

(SONG, D., WAGNER, D., & PERRIG, A.)



- special two-layered encryption construct. Given a trapdoor, the server can strip the outer layer and assert whether the inner layer is of the correct form.
- Plaintext will be encrypted using a secret key sk.
- Compute a bitwise XOR of plain text with a sequence of stream cipher bits to get the keyword encrypted.

Flash Memory Summit

PROBLEMS WITH SEQUENTIAL SEARCH:

(SONG, D., WAGNER, D., & PERRIG, A.)

- Inefficiency in Searching
- Vulnerable to Statistical Attacks
- Full Document Decryption
- Lack of Indexing
- Scalability Issues

(CURTMOLA, R., GARAY, J. A., KAMARA, S., & OSTROVSKY, R. (2006))

Reverse indexing in SSE operates as follows:

1. The data owner preprocesses the plaintext documents and extracts keywords or tokens.
2. Each keyword is encrypted individually using the same encryption scheme used for document encryption.
3. A reverse index is created, associating each keyword with the encrypted documents that contain it.
4. When a search request is made, the search query is encrypted, and the server performs a lookup in the reverse index to find the encrypted documents relevant to the search query.
5. The server can then retrieve the encrypted documents from storage and evaluate them against the encrypted search query.

$D \leftarrow \{D_1, D_2, D_3\}$
$D_1 \leftarrow$ {search, encryption, symmetric, cryptography}
$D_2 \leftarrow$ {public, encryption, add}
$D_3 \leftarrow$ {add, public, cryptography}

Keyword	Locations
encryption	$[D_1, D_2]$
symmetric	$[D_1]$
cryptography	$[D_1, D_3]$
add	$[D_3, D_2]$
search	$[D_1]$
public	$[D_2, D_3]$

Inverted Index Corr. To D

Our Approach: (Improved Curtmola)

We are proposing an SSE Engine which is an Improved version of Curtmola. In our approach, keywords/ tokens are hashed rather than encrypted. We have used a standard Key Management Design (KMS) for generation and verification of keys. The table below displays further enhancements made in our research over the current Curtmola method:

Curtmola	Our Approach
Keyword is encrypted	Keyword is hashed
Uncompressed Map	Compressed map – giving 46.90% storage improvement
No Key management System (KMS) Design	Standard KMS design
Higher Search time as it includes decrypting the keywords	Improved searching time

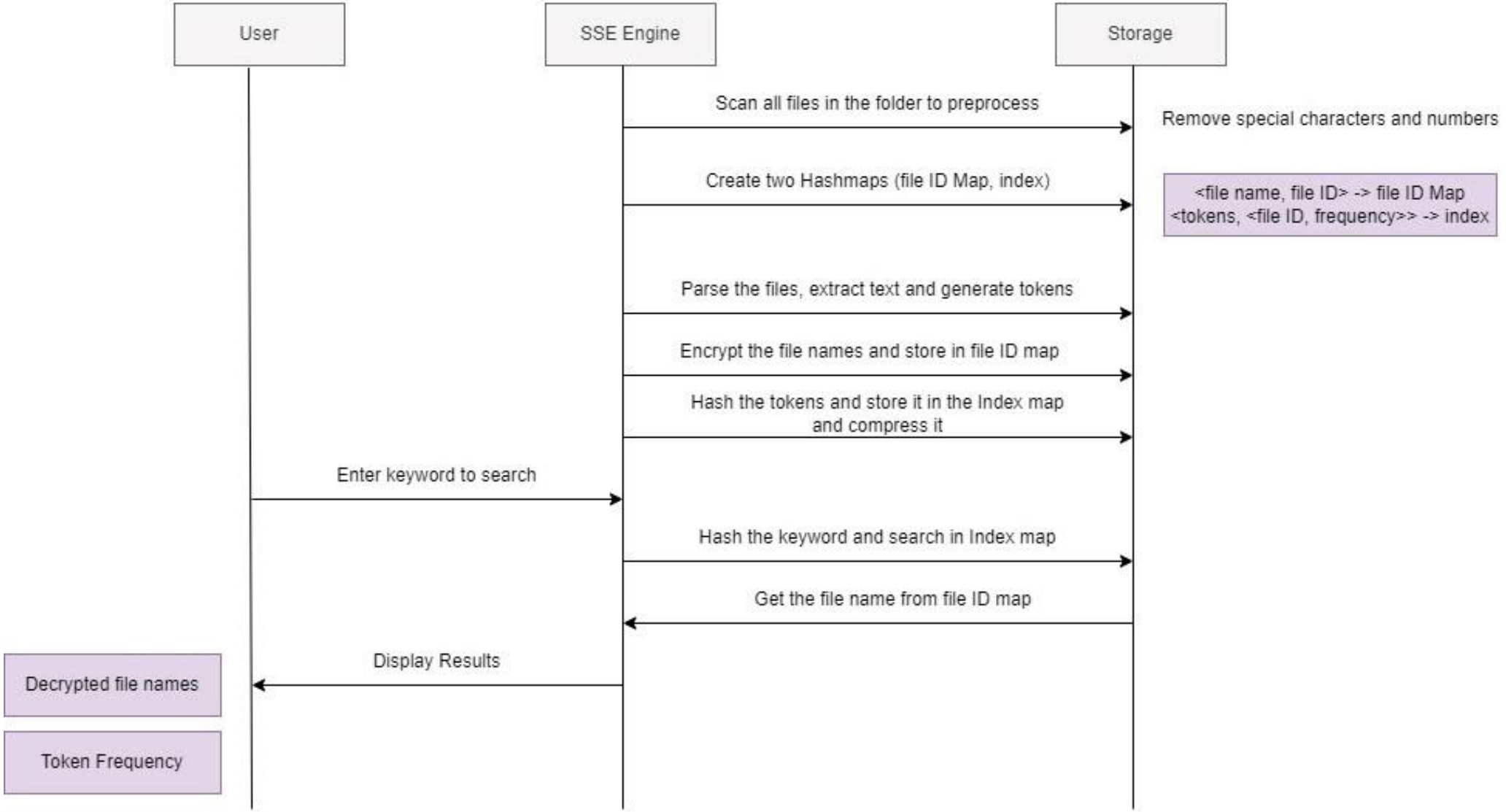
Steps involved in our approach:

Time Analysis Chart:

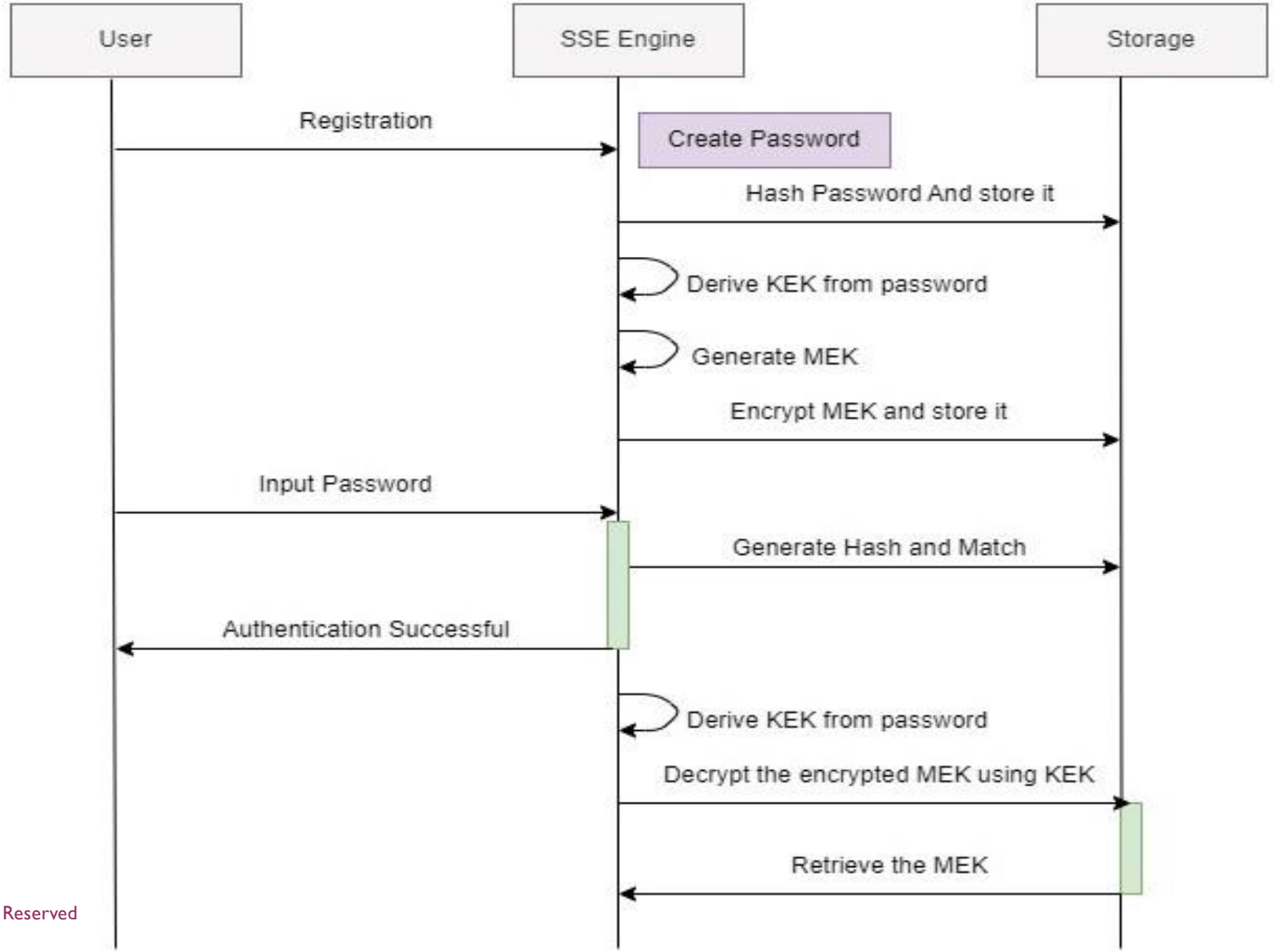
Following table displays the indexing and searching time (which includes encryption, decryption and hashing) on a 1.25 GB file:

Indexing Time	Searching Time	File Size	Map Size	Total Files Indexed	Unique Tokens
42.3 seconds	0.0150 seconds	1.25 GB	28.3 MB	149	150336

SSE Design Flow



Key Management System

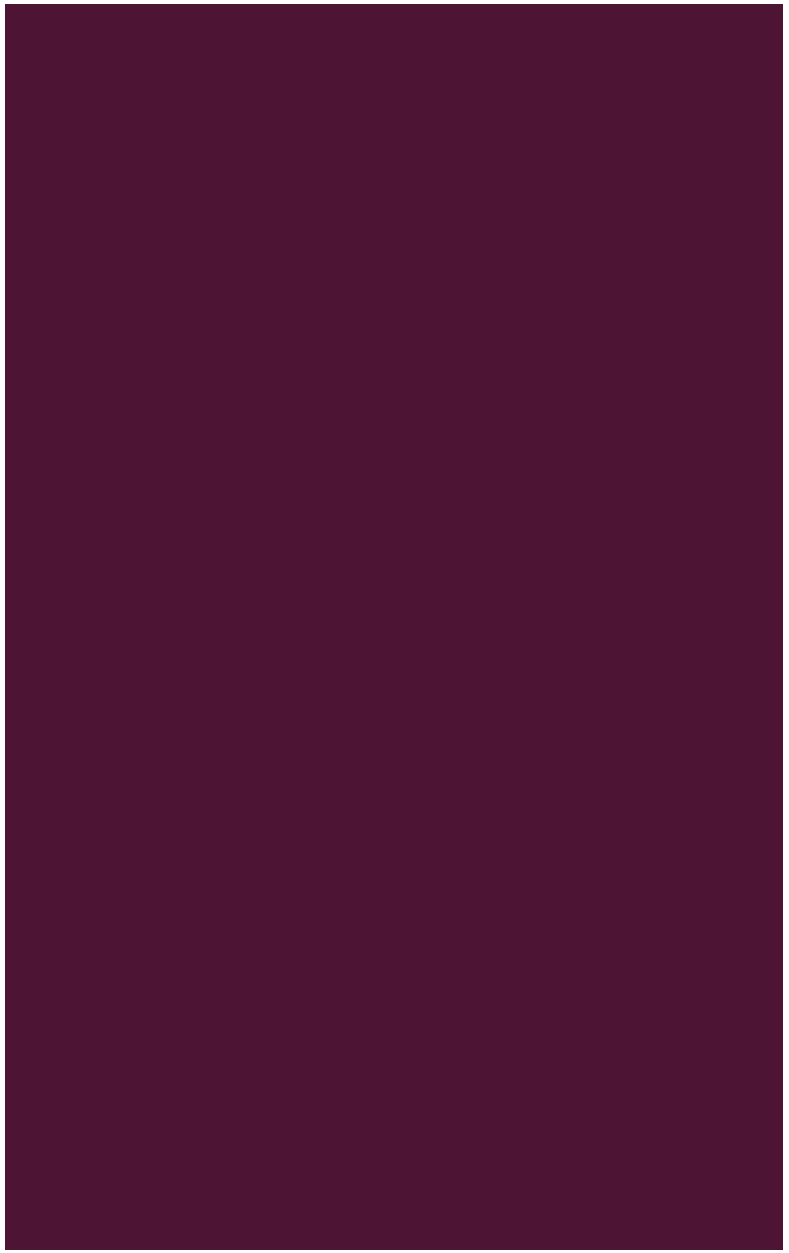


Future Work And Conclusion

In our work, we can focus on the following areas in future:

1. Multi-user SSE: allow multiple users or parties along with the owner to search and retrieve data.
2. Distributed Indexing: Spreads out the load on an index of stored data in a data storage system. Ensure that two copies of the same data item cannot be read or written by two concurrent transactions.

We Introduce New Security Definitions and Constructions to Overcome Limitations in Current Approaches for Privacy-Preserving Encrypted Search. Our Stronger Definitions Ensure Security for Realistic Search Scenarios. We Propose the Highly Efficient WD SSE Engine, a Provably Secure and Optimal Construction.



THANK YOU!