Searchable Symmetric Encryption: Sequential Scan Can Be Practical

Máté Horváth, István Vajda
Laboratory of Cryptography and Systems Security (CrySyS),
Budapest University of Technology and Economics
Email: mhorvath@crysys.hu, vajda@hit.bme.hu

***Abstract*—The proliferation of cloud computing highlights the importance of techniques that allow both securing sensitive data and flexible data management at the same time. One line of research with this double motivation is the study of Searchable Symmetric Encryption (SSE) that has provided several outstanding results in the recent years. These solutions allow sublinear keyword search in huge databases by using various data structures to store keywords and document identifiers. In this work, we focus on certain scenarios in which search over the whole database is not necessary and show that the otherwise inefficient sequential scan (in linear time) can be very practical. This is due to the fact that adding new entries to the database comes for free in this case while updating a complex data structure without information leakage is rather complicated. To demonstrate the practicality of our approach we build a simple SSE scheme based on bilinear pairings and prove its security against adaptive chosen-keyword attacks in the standard model under the widely used SXDH assumption.**

***Index Terms*—Searchable Symmetric Encryption, Forward Index, Type-3 Pairings, MAC.**

I. Introduction

Computation on hidden data and especially Searchable Symmetric Encryption (SSE) has become an extensively studied area of cryptography in the last more than one and a half decade, since the appearance of the seminal paper of [14]. The concept of SSE allows the secure storage of sensitive data on untrusted servers in the cloud without losing all the flexibility that plaintext data would allow. More precisely it supports keyword search over the ciphertexts in the following way: encrypted queries called trapdoors can be sent to the server which can test whether any of the stored ciphertexts matches the keyword underlying the trapdoor.

The two natural approaches towards realizing SSE are called "forward" and "inverted index". The first one is to attach (or even include) one-way mappings of searchable keywords to the encrypted data. This leads to linear search complexity in the number of documents as the server has to go through all of them with a sequential scan to find all the matches for a trapdoor. A more sophisticated arrangement of the ciphertexts is to build an "inverted index". In this case the documents (or their IDs) are sorted based on the one-way mappings of keywords which are related to them. The latter solution allows logarithmic search complexity in the number of keywords. This clear benefit caused that the inverted index approach became prevalent in SSE design which made significant progress in the past years [4].

At the same time these solutions are rather complex and while operating smoothly on huge *static* databases (DB), handling the *rapid expansion* of the DB turns out to be more troublesome as the underlying data structure has to be updated without information leakage (see Table I for details).

In this work we are looking for a simple solution to specific problems in the domain of SSE for which the inverted index based approach is not practical. Imagine the following scenarios!

Scenario 1. A device, deployed in an untrusted environment stores its own event logs in an SSE encrypted form with the event type as a keyword and possibly with a public time-stamp. It is often plausible to assume that the different events occur relatively often, resulting in frequent updates, and a remote operator is more likely to search for a specific event in a given time frame than in the entire DB. He can send a trapdoor together with a tract of time to the device that checks its encrypted DB and replies with the positive test results.

Scenario 2. The on-board unit of a vehicle regularly sends encrypted aggregate data to a honest but curious remote server. The data packets are tagged with a few predefined characteristic features of the given packet e.g., the oil level was under the limit, speeding was detected, the seatbelt was not fastened etc. An authority, possibly maintained by the car manufacturer as part of their services, can issue trapdoors for the car service/insurance company/police who can send these to the server and check whether some hypothetical event has occurred in a time period of interest. This process speeds up the actions as it can happen even in the absence of the vehicle and preserves the privacy of the car owner as no data, unrelated to an event, has to be decrypted as the relevance of some hidden data can be tested using SSE.

The first common characteristic feature of these use cases is that finding *all* the occurrences of a given keyword in the whole DB is not the goal. Particularly they highlight that finding some correspondence in a properly chosen part of the DB can also be meaningful. Secondly, new entries are frequently added to the DB and therefore rapid updates are crucial, hitting the Achilles heel of the widespread inverted index approach.

A. Related Works

Several aspects of searchable encryption have been studied in the past years. We mention only a few of them and refer to the survey of [4] for an extensive summary on SSE. [7] captured first formally the intuitive goal of minimizing information leakage during keyword search (see section III-B). Schemes were put forward that allow not only single but also conjunctive/disjunctive keyword search [5]. Ranked keyword search over encrypted data was proposed by [18], [17]. Dynamic SSE schemes were designed to handle large DBs with possible updates (see details in subsection IV-C). Keyword search in the public-key setting (PEKS) was introduced by [2] and later improved in several directions.

B. Our Contribution

In this work we aim to build SSE, that is optimized for the above scenarios and provably fulfils the strongest security requirements towards SSE schemes. For this we return to the forward index method to design an SSE scheme that handles newly encrypted data without requiring any updates on the already stored DB. Our solution is built on favourable asymmetric (Type-3) bilinear groups in which the Symmetric eXternal Diffie-Hellman (SXDH) assumption holds. The construction is built in a modular way. Its core is a keyword encryption and trapdoor generation method, both with randomized outputs that allows equality-test to determine whether a given trapdoor-ciphertext pair corresponds to the same keyword or not. These special keyword encryptions can then be attached to the ordinary encryption of a document (or file) and stored together on a honest but curious server.

Our keyword checking algorithm is built in bilinear groups and in order to use the same algebraic structure for the data encryption task we utilize the ElGamal cryptosystem without publishing its public key. The common structure also allows to tie the different components together through a random value, that is however not necessary if the integrity of the ciphertext is assumed to be granted. Nevertheless, our use of ElGamal is not inevitable and it can be substituted with any, more practical, symmetric key scheme with semantic security. The security of our scheme against adaptive chosen-keyword attacks (**IND-CKA2**) is proven in the standard model.

The rest of the paper is organized as follows. In section II we summarize the necessary background, while in section III the formal definition of the algorithms and security of SSE is given. Section IV is dedicated to the proposed scheme, its security analysis and comparisons with related concepts. Finally we conclude our findings in section V.

II. PRELIMINARIES

First of all we briefly introduce the underlying tools of our construction, namely bilinear maps and message authentication codes (MACs), and discuss our hardness assumption.

A. Bilinear Pairings

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative cyclic groups of prime order p. Let g_1 and g_2 be the generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. Let $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map (pairing), with the following properties:

1) Bilinearity: $\forall u \in \mathbb{G}_1$, $v \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$
2) Non-degeneracy: $e(g_1, g_2) \neq 1$.

We say that $\mathcal{G} = \{p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, g_1, g_2, e\}$ is a bilinear instance if all the group operations and the bilinear map e are efficiently computable. In this work we apply the so-called "Type-3" pairings, meaning that $\mathbb{G}_1 \neq \mathbb{G}_2$ and no efficiently computable isomorphism exists between them. We note that based on both its efficiency and security, this pairing type is considered to be the ideal choice when instantiating a cryptosystem [6].

The security of our SSE scheme can be reduced to the hardness of the so-called Symmetric eXternal Diffie-Hellman (SXDH) problem that we define next. Informally speaking, given (g^a, g^b, g^c), where g is a generator of group $\mathbb{G}$ and $a, b, c \in \mathbb{Z}_p^*$, the Decisional Diffie-Hellman (DDH) problem is to decide whether $c = ab$ or not. The SXDH assumption states that no efficient algorithm can solve the DDH problem either in $\mathbb{G}_1$ or in $\mathbb{G}_2$ of a bilinear instance.

Assumption 1 (SXDH). *Let $g_i \in \mathbb{G}_i$ be a generator element of the group and $a_i, b_i \in \mathbb{Z}_p^*$ are uniformly random values for $i = 1, 2$. We say that the SXDH assumption holds in a bilinear instance $\mathcal{G}$ if given*

$$(g_i, g_i^{a_i}, g_i^{b_i}, g_i^{R_i}),$$

for $i = 1, 2$, no polynomial time algorithm can decide whether $R_i = a_i b_i$ or R_i is also a uniformly random value from $\mathbb{Z}_p^*$.

B. Message Authentication Codes

In our construction we are going to make use of deterministic Message Authentication Codes (MAC) with a specific syntax.

Definition 1. *A deterministic MAC consists of the following two algorithms:*

MAC.KeyGen(λ)$\to \mathsf{sk}_{MAC}$ *This randomized algorithm chooses a secret key sk_{MAC} based on security parameter λ.*

MAC(sk_{MAC}, m) $\to \tau$ *Using the secret key sk_{MAC}, this deterministic algorithm produces a tag τ for the input message m.*

Note that in case of deterministic MACs, the verification of a pair (m, τ) can be done simply by checking whether **MAC**(sk_{MAC}, m) $= \tau'$ equals τ or not. In this work we are interested in MACs in special form i.e., we assume that tag $\tau = \alpha^{F(\mathsf{sk}_{MAC}, m)}$, where α is a generator element of either group $\mathbb{G}_1$ or $\mathbb{G}_2$ of the pairing group $\mathcal{G}$ and F is some function of the secret key and the message to be authenticated. In the literature several Pseudo-Random Functions (PRF) were described [13], [12], [3], [1] in the desired form under various hardness assumptions. However, for our purposes this stronger guarantee of pseudo-randomness is not required, any of these can serve as proper MAC functions that is existentially unforgeable under chosen message attack ($\mathbf{EU\text{-}CMA}$ secure). For instance the construction of [13] can be used assuming the hardness of the DDH problem that is implied by the SXDH assumption.

III. SEARCHABLE SYMMETRIC ENCRYPTION

The focus of this work is SSE, more precisely using the terminology of [4] we are interested in the single writer/single reader setting. In an abstract version of the scenarios, described in the introduction, the data owner generates the system parameters, secret keys, encrypts data and prepares the trapdoors using the secret keys. Besides storing the ciphertexts, the server is able to search over encrypted data using trapdoors for specific keywords, issued by the data owner. The server is assumed to be semi-trusted (honest-but-curious) and the communication channel between the user and the server supposed to be authenticated. Next we define the algorithms and the security of an SSE scheme that fits into this abstract scenario.

A. Definition of SSE

An SSE scheme consists of the following algorithms.

Setup$(\lambda) \to (\mathsf{P}, \mathsf{sk})$ Upon receiving a security parameter λ it outputs the system parameters P and a secret key sk.

TrpdGen$(\mathsf{P}, \mathsf{sk}, \hat{w}) \to (T)$ Using the secret key sk it computes a trapdoor T that can be used to test whether some ciphertext C was encrypted under keyword $\hat{w}$ or not.

Encrypt$(\mathsf{P}, \mathsf{sk}, m, w) \to (C)$ Using the secret key sk it computes ciphertext C that encrypts m under keyword w.

Decrypt$(\mathsf{P}, \mathsf{sk}, C) \to (m)$ It decrypts ciphertext C with secret-key sk and outputs the resulting plaintext m.

Test$(\mathsf{P}, T, C) \to \{0, 1\}$ The equality testing algorithm outputs 1 if T and C encodes the same keyword i.e., $w = \hat{w}$ and 0 otherwise.

B. Security Model for SSE

The commonly used security model for SSE was defined by [7] to capture the intuition that, in the course of using the scheme, the remotely stored files and search queries together do not leak more information about the underlying data than the search pattern and the search outcome. In our security definition we follow [7], but we formulate it - to the best of our knowledge first time - in the context of a forward index.

While in the inverted index-based approach the index and the ciphertexts are handled separately, in our case of a forward index, it is natural to view the "index" as part of the ciphertext.We formulate the model in this way, defining indistinguishability under adaptive chosen keyword attack ($\mathbf{IND\text{-}CKA2}$) through a game between a challenger and an adversary. In the game the adversary has to recognize which one of the two challenge data sets (consisting of messages and their keywords chosen by herself) was encrypted by the challenger. Note that in a forward index even the knowledge of the order of ciphertexts can help the attacker, that is why our challenger provides her with a random permutation of ciphertexts prepared from the randomly chosen challenge message set. The adversary has access not only to the encryption itself, but also to a trapdoor generation oracle that can be queried adaptively with pairs of keywords corresponding to the two challenge sets. The oracle answers consistently with a trapdoor for that keyword which belongs to the encrypted challenge data set. The only restriction is that the queried keywords cannot separate the two challenge sets, as we are interested in information leakage beyond the search result.

For the ease of exposition we assume that there is a single keyword for each message, and the challenge DBs are unchanged during the game[1], but these can be easily generalized. More formally we use the subsequent definition of security following [7, §4.2.2].

Definition 2 (Adaptive indistinguishably). *Let SSE = (**Setup,TrpdGen,Encrypt,Decrypt,Test**) be a secret-key searchable encryption scheme, $\lambda \in \mathbb{N}$ a security parameter, and $\mathcal{A} = (\mathcal{A}_0, \ldots, \mathcal{A}_{q+1})$ a non-uniform adversary. Consider the probabilistic experiment $\mathbf{Ind\text{-}CKA2}_{\mathbf{SSE},\mathcal{A}}(\lambda)$ depicted on Figure 1 with the restriction that the number of keyword matches between the challenge message sets and the corresponding test-key queries are equal i.e.,*

$$\#\{i | \hat{w}_j^0 = w_i^0 \text{ for } j \in [k]\} = \#\{i | \hat{w}_j^1 = w_i^1 \text{ for } j \in [k]\}$$

for all $k = 1, \ldots, q$, where q is some polynomial of the security parameter λ. We say that an SSE scheme is secure in the sense of adaptive indistinguishability if for all polynomial-time adversaries $\mathcal{A} = (\mathcal{A}_0, \ldots, \mathcal{A}_{q+1})$,

$$\Pr(\mathbf{Ind\text{-}CKA2}_{\mathbf{SSE},\mathcal{A}}(\lambda) = 1) \leq \frac{1}{2} + \mathsf{negl}(\lambda).$$

```
$\mathbf{Ind\text{-}CKA2}_{\mathbf{SSE},\mathcal{A}}(\lambda)$ Security Game
$\mathsf{sk} \leftarrow_\$ \mathsf{Setup}(1^\lambda)$
$b \leftarrow_\$ \{0,1\}$
$(\mathsf{state}_{\mathcal{A}_0}, D^0, D^1) \leftarrow \mathcal{A}_0(1^\lambda)$
parse $D^b$ as $\{(m_1^b, w_1^b), \ldots, (m_n^b, w_n^b)\}$
for $1 \leq j \leq n$,
    $CT_i^b \leftarrow_\$ \mathsf{Enc}(\mathsf{sk}, m_i^b, w_i^b)$,
$CT^b := (CT_{\pi_1}^b, \ldots, CT_{\pi_n}^b)$ for a random permutation $\pi$,
for $1 \leq j \leq q$,
    $(\mathsf{state}_{\mathcal{A}_j}, \hat{w}_j^0, \hat{w}_j^1) \leftarrow \mathcal{A}_j(\mathsf{state}_{\mathcal{A}_{j-1}}, CT^b, \{T_i^b\}_{i \in [j]})$
    $T_j^b \leftarrow_\$ \mathsf{TrpdGen}(\mathsf{sk}, \hat{w}_j^b)$
$b' \leftarrow \mathcal{A}_{q+1}(\mathsf{state}_{\mathcal{A}_q}, CT^b, \{T_j\}_{i=1,\ldots,q})$
return $b = b'$
```

Figure 1. $\mathbf{IND\text{-}CKA2}$ security game for forward index SSE schemes.

[1]The simplest meaningful way of modelling a dynamically growing database is to provide $\mathcal{A}$ with additional ciphertexts of unknown messages under unknown keywords, but for simplicity we disregard these in the model, especially as these extra ciphertexts would not affect our security proof.

IV. PROPOSED SCHEME

In this part we describe the algorithms of the proposed SSE scheme and analyse both its security and performance.

A. SSE Construction

The intuition behind the construction of our **Test** algorithm is fairly simple. We build the trapdoors for a specific keyword and the keyword related ciphertext components in a symmetric manner: both are randomised MACs of the underlying keyword, however, represented in distinct groups $\mathbb{G}_1$ or $\mathbb{G}_2$. This allows to test equality by "mixing" the ciphertext and the trapdoor in two different ways (using the pairing operation) that are equal only if the underlying keywords are the same. Using distinct groups prevents the testability between both ciphertexts and trapdoors. In more detail, the algorithms are the following.

Setup$(\lambda) \rightarrow (\mathsf{P}, \mathsf{sk})$ It proceeds with the following steps:

- samples a Type-3, asymmetric bilinear instance: $\mathcal{G} = \{p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e\}$,
- runs the **MAC.KeyGen** algorithm of a MAC function (that implicitly defines $F(.,.)$) as described in §II-B,
- samples secret keys $k_1, k_2 \leftarrow_{\$} \mathbb{Z}_p^*$,
- outputs $\mathsf{P} = (\mathcal{G}, F(.,.))$ and secret key $\mathsf{sk} = \{k_1, k_2, \mathsf{sk}_{MAC}\}$.

TrpdGen$(\mathsf{P}, \mathsf{sk}, \hat{w}) \rightarrow (T)$ Upon receiving a keyword $\hat{w}$ it

- samples $r' \leftarrow_{\$} \mathbb{Z}_p^*$,
- and generates the following trapdoor:

$$T = \left(t_1 = g_2^{r'}, t_2 = g_2^{r' k_2 F(\mathsf{sk}_{MAC}, \hat{w})}\right).$$

Encrypt$(\mathsf{P}, \mathsf{sk}, m, w) \rightarrow (C)$ In order to encrypt some data $m \in \mathbb{G}_1$ under the keyword $w \in \mathbb{Z}_p^*$ the algorithm

- first chooses $r \leftarrow_{\$} \mathbb{Z}_p^*$,
- and computes the following ciphertext:

$$C = \left(c_1 = g_1^{r}, c_2 = g_1^{r k_1} m, c_3 = g_1^{r k_2 F(\mathsf{sk}_{MAC}, w)}\right).$$

Decrypt$(\mathsf{P}, \mathsf{sk}, C) \rightarrow (m)$ After parsing C as (c_1, c_2, c_3) and sk as $(k_1, k_2, \mathsf{sk}_{MAC})$, in order to recover the encrypted data an ElGamal-decryption style computation is executed: $c_2/(c_1)^{k_1} = m$.

Test$(\mathsf{P}, T, C) \rightarrow \{0, 1\}$ To test whether a ciphertext C was encoded using the same keyword that is hidden in trapdoor T the following equality is checked:

$$e(c_3, t_1) = e(c_1, t_2).$$

If the equality holds the output is 1, otherwise 0.

The correctness of the **Decrypt** and **Test** algorithms follows after substitution of the proper values into the formulas i.e., in case of the latter one $e(c_3, t_1) = e(g_1, g_2)^{rr' k_1 F(k_2, w)} = e(g_1, g_2)^{rr' k_1 F(k_2, \hat{w})} = e(c_1, t_2)$ iff $w = \hat{w}$.

B. Security Analysis

Because of space limitations in this part we formulate our main theorem without its proof which is deferred to the full version of this paper. However, we provide a rough intuition of our approach and of a further possible safeguard.

Theorem IV.1. *If the SXDH assumption holds, then the proposed SSE scheme is IND-CKA2 secure according to Definition 2.*

Our strategy to prove the theorem is to define a sequence of hybrid games where in the last hybrid the attacker receives random values instead of the encryption of the challenge message and trapdoors for the queried keywords, thus she cannot have a non-negligible advantage in that game. It is possible to systematically show that the subsequent hybrids are indistinguishable for the adversary and thus her advantage in the original game is essentially negligible as well. To see these, we use the observation that the scheme comprise $\mathbf{IND\text{-}CPA}$ and $\mathbf{EU\text{-}CMA}$ secure components.

Forward privacy guarantees that trapdoors can only be used to test keywords of documents which were already part of the DB at the time of issuing the trapdoor. By default, our scheme is not forward secure (trapdoors can be stored by the server and can be used to test future ciphertexts) but in case of the considered applications proper keyword usage can remedy this deficiency. As we focus on searching in relatively small parts of the whole data set, the DB can be divided into separate parts, based on publicly available parameters like time. Following this separation keywords can also contain the identifier for the time frame. E.g., instead of "Speeding", "Speeding*:Year:Month*" can be used both in keywords and trapdoors restricting searchability to predefined time frames thus achieving forward (and also backward) privacy between the periods. Note that the same solution of time-specific keywords would result in an infinitely growing inverted index.

C. Evaluation and Comparisons

We compare our results with dynamic SSE schemes which are the most suitable in the literature for the use cases that we considered in this work. Table I shows a comparison using the following notations: n denotes the number of documents (data entries), w_D is the number of keywords per a specific document, W is the total number of distinct keywords in the DB, n_w is the number of documents matching the searched keyword w, a is the total number of additions to the DB and d is the total number of deletions, b is the bit length of encrypted documents. * indicates that update requires some rounds of interaction between the server and the client and ** denotes amortized complexity.

As predicted in the introduction, Table I confirms that our search strategy of sequential scan is not competitive, unless only a small portion of the DB is enough to scan. More precisely the size of the scanned set of ciphertexts should be at most $\log W$, that is realistic in the investigated scenarios. The most important benefits of our scheme include resistance against adaptive chosen-keyword attacks in the standard model

Table I
COMPARISON OF OUR RESULTS AND DYNAMIC SSE SCHEMES.

Scheme	Model	Security	Fw/Bw Privacy	Update Time	Update Privacy	Search Time
[14]	Standard	**IND-CPA**	× / ×	$O(b)$	×	$O(n \cdot b)$
[16]	Standard	**IND-CKA2**	× / ×	$O(w_D)^*$	×	$O(\log W)$
[11]	ROM	**CKA2**	× / ×	$O(w_D)$	×	$O(n_w)$
[10]	ROM	**CKA2**	× / ×	$O(\log n)^*$	✓	$O(n_w \log n)$
[5]	ROM	**CKA2**	× / ×	$O(w_D + W \log n)$	×	$O(n_w + a + d)$
[15]	ROM	**CKA2**	✓/ ×	$O(w_D \log(nW))^*$	✓	$O(n_w + d)$
[9]	ROM	**CKA2**	× / ×	$O(nw_D/D)^{**}$	×	$O(nw_D/D)^{**}$
[19]	ROM	**CKA2**	✓/ ✓	$O(W)$	✓	$O(W)$
[8]	Standard	**IND-CKA2**	× / ×	$O(w_D \cdot W)$	×	$O(\log n)$
Our scheme	Standard	**IND-CKA2**	✓/ ✓	$O(w_D)$	✓	$O(n \cdot w_D)$

and non-interactive update of ciphertexts with low complexity, depending only on the number of keywords. Moreover our ciphertexts (including the index) and trapdoors are very short, consisting of $w_D + 2$ and 2 group elements respectively.

Let us emphasize that updating the DB with a new record is straightforward in our approach. The client encrypts the data together with the keywords and the server only has to store the received ciphertexts contrary to other solutions where the server has to "find the place" of the new entry in the index. This latter process also harms the privacy of updates in most cases by leaking information about the added keywords (e.g., all documents with common keywords can be identified). In our case only the number of added keywords is leaked, however, in the targeted applications it is plausible to assume that the number of keywords are not varying among the different "documents".

V. CONCLUSION

In this work we revisited the role of sequential scan in searchable encryption and showed a construction that outperforms the existing solutions in certain scenarios from real-life. Our scheme was proven **IND-CKA2** secure in the standard model assuming the widely used SXDH holds. To the best of our knowledge this is the first scheme with a forward index that is proven secure in this strong model. Our modular design allows further performance improvements in a future implementation: ElGamal encryption can be substituted with any, more efficient, semantically secure symmetric-key encryption scheme and the used MAC function might be also substituted with a group generator raised to the power of a MAC of the keyword w.

ACKNOWLEDGMENT

The authors would like to thank the valuable remarks of Levente Buttyán. The research presented in this paper was supported by the National Research, Development and Innovation Office – NKFIH of Hungary under grant contract no. 116675 (K).

REFERENCES

[1] Abdalla, M., Benhamouda, F., and Passelègue, A. An algebraic framework for pseudorandom functions and applications to related-key security. In *Advances in Cryptology - CRYPTO 2015, Proceedings, Part I*, pages 388–409.

[2] Boneh, D., Crescenzo, G. D., Ostrovsky, R., and Persiano, G. Public key encryption with keyword search. In *Advances in Cryptology - EUROCRYPT 2004, Proceedings*, pages 506–522.

[3] Boneh, D., Montgomery, H. W., and Raghunathan, A. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *Proceedings of the 17th ACM CCS 2010*, pages 131–140.

[4] Bösch, C., Hartel, P. H., Jonker, W., and Peter, A. A survey of provably secure searchable encryption. *ACM Comput. Surv.*, 47(2):18:1–18:51.

[5] Cash, D., Jaeger, J., Jarecki, S., Jutla, C. S., Krawczyk, H., Rosu, M., and Steiner, M. Dynamic searchable encryption in very-large databases: Data structures and implementation. In *21st Annual Network and Distributed System Security Symposium, NDSS 2014*.

[6] Chatterjee, S. and Menezes, A. On cryptographic protocols employing asymmetric pairings - the role of Ψ revisited. *Discrete Applied Mathematics*, 159(13):1311–1322.

[7] Curtmola, R., Garay, J. A., Kamara, S., and Ostrovsky, R. Searchable symmetric encryption: improved definitions and efficient constructions. In Juels, A., Wright, R. N., and di Vimercati, S. D. C., editors, *Proceedings of the 13th ACM CCS, 2006*, pages 79–88. ACM.

[8] Gajek, S. Dynamic symmetric searchable encryption from constrained functional encryption. In *Topics in Cryptology - CT-RSA 2016, Proceedings*, pages 75–89.

[9] Hahn, F. and Kerschbaum, F. Searchable encryption with secure and efficient updates. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, 2014*, pages 310–320.

[10] Kamara, S. and Papamanthou, C. Parallel and dynamic searchable symmetric encryption. In *Financial Cryptography and Data Security - FC 2013, Revised Selected Papers*, pages 258–274.

[11] Kamara, S., Papamanthou, C., and Roeder, T. Dynamic searchable symmetric encryption. In *the ACM Conference on Computer and Communications Security, CCS'12*, pages 965–976.

[12] Lewko, A. B. and Waters, B. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In *Proceedings of the ACM CCS 2009*, pages 112–120.

[13] Naor, M. and Reingold, O. Number-theoretic constructions of efficient pseudo-random functions. In *38th Annual Symposium on Foundations of Computer Science, FOCS '97*, pages 458–467.

[14] Song, D. X., Wagner, D., and Perrig, A. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy, 2000*, pages 44–55.

[15] Stefanov, E., Papamanthou, C., and Shi, E. Practical dynamic searchable encryption with small leakage. In *21st Annual Network and Distributed System Security Symposium, NDSS 2014*.

[16] van Liesdonk, P., Sedghi, S., Doumen, J., Hartel, P. H., and Jonker, W. Computationally efficient searchable symmetric encryption. In *Secure Data Management, 7th VLDB Workshop, SDM 2010, Proceedings*, pages 87–100.

[17] Xia, Z., Wang, X., Sun, X., and Wang, Q. A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. *IEEE Trans. Parallel Distrib. Syst.*, 27(2):340–352.

[18] Yang, Y., Li, H., Liu, W., Yao, H., and Wen, M. Secure dynamic searchable symmetric encryption with constant document update cost. In *IEEE GLOBECOM 2014*, pages 775–780.

[19] Yavuz, A. A. and Guajardo, J. Dynamic searchable symmetric encryption with minimal leakage and efficient updates on commodity hardware. In *Selected Areas in Cryptography - SAC 2015, Revised Selected Papers*, pages 241–259.