Cryptography
Research
Centre

Searchable Encryption

and a practical construction

March 15, 2022

Chiara Marcolla

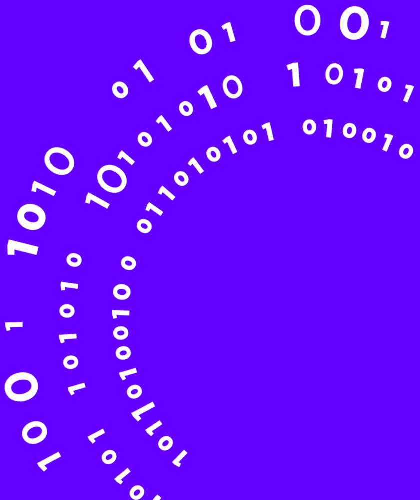
chiara.marcolla@tii.ae

Technology
Innovation
Institute

Contents

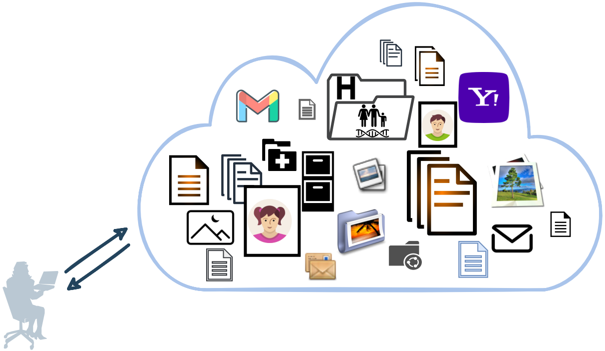
1. Introduction on Searchable Encryption
2. Exipnos: An efficient VDSSE scheme
3. Comparison with the state of the art

Introduction on Searchable Encryption



Why Searchable Encryption

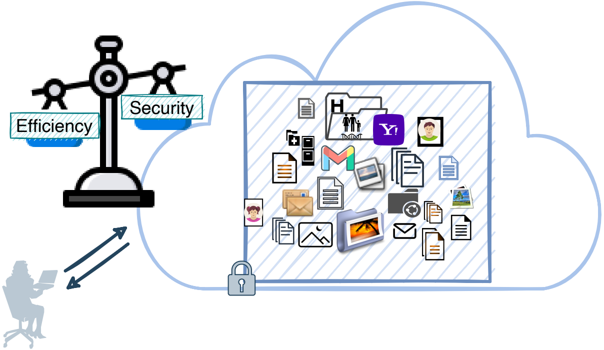
Remote cloud storage services (such as Dropbox, Google Drive and Google Photo) are used for backups or outsourcing data.



Cloud has full access to (sensitive) data: medical data, emails, financial data, pictures, . . .

Why Searchable Encryption

Remote cloud storage services (such as Dropbox, Google Drive and Google Photo) are used for backups or outsourcing data.



Cloud has full access to (sensitive) data: medical data, emails, financial data, pictures, . . .

Symmetric Searchable Encryption

A Searchable Encryption (SE) scheme allows a server to search in encrypted data on behalf of a client without learning information about the plaintext data.

$DB[w] = (\mathsf{id}_{i_1}, \cdots, \mathsf{id}_{i_n})$

Keyword

$\mathsf{Enc}_k(DB) = \mathrm{EDB}$

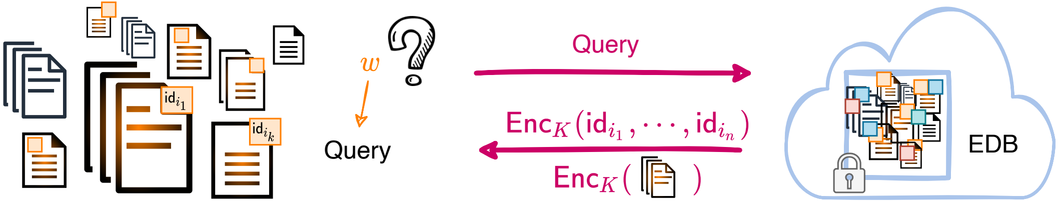
keywords	documents ids
w_1	$1, 5, 6$
w_2	$7, 100$
$\vdots$	$\vdots$
w_m	$1, 6, 15, 24, 100$

It is composed by an algorithm:

- Setup$(DB) = (\mathrm{EDB}, K)$, where K is a secret key, EDB the encrypted DB,

Symmetric Searchable Encryption

A Searchable Encryption (SE) scheme allows a server to search in encrypted data on behalf of a client without learning information about the plaintext data.



It is composed by an algorithm:

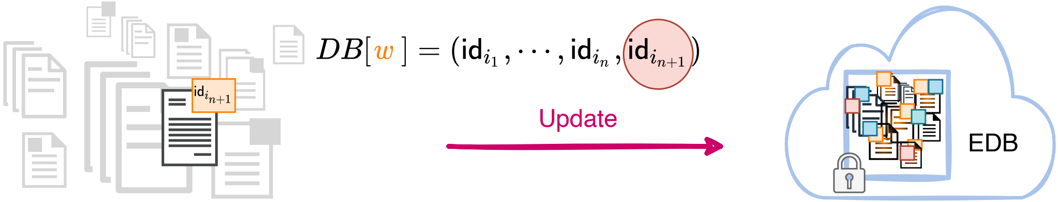
- Setup$(DB) = (\mathrm{EDB}, K)$, where K is a secret key, EDB the encrypted DB,

and a protocol between a client and a server:

- Search$(K, q, \mathrm{EDB}) = ($Search$_C(K, q)$, Search$_S(\mathrm{EDB}))$, where q is the search query.

Dynamic Symmetric Searchable Encryption

A Dynamic Symmetric Searchable Encryption (DSSE) scheme supports modifications to the encrypted dataset such as document insertion or deletion.



$DB[w] = (\mathsf{id}_{i_1}, \cdots, \mathsf{id}_{i_n}, \mathsf{id}_{i_{n+1}})$

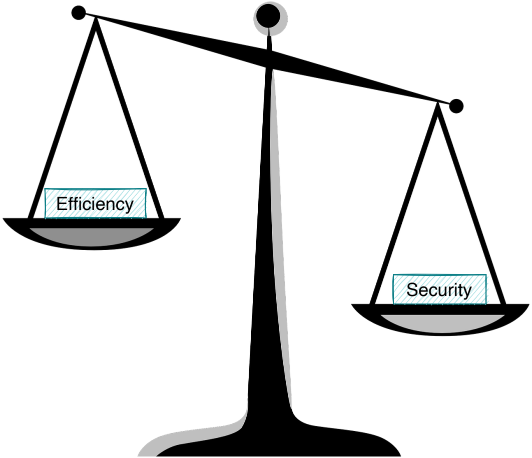
It is composed by an algorithm

- Setup$(DB) = (\mathrm{EDB}, K, \sigma)$, where K and EDB as before and and σ the client's state.

and two protocols between a client and a server:

- Search$(K, q, \sigma, \mathrm{EDB}) = (\mathsf{Search}_C(K, q, \sigma), \mathsf{Search}_S(\mathrm{EDB}))$, where q is the search query.
- Update$(K, \sigma, op, in, \mathrm{EDB}) = (\mathsf{Update}_C(K, \sigma, op, in), \mathsf{Update}_S(\mathrm{EDB}))$, where update operations $op \in \{add, del\}$ and an input in parsed as the index id and the keyword w.

Trade-off between security and efficiency



Security - Leackages

To achieve efficient SSE scheme we allow the server to learn some information, which can be divided into three groups:

Setup Phase

- Ciphertext size pattern includes the size of encrypted data: the number of encrypted files containing the keyword.

Search Phase

- Search pattern is induced by a search query: how many times has the keyword been searched (and when).
- Access pattern refers to the information of query results: which file are retrieve.

Security - Forward and Backward privacy

Cryptography Research Centre

A DSSE scheme leaks information on the update files and the keyword tokens attached to them.

Update Phase

- Forward privacy guarantees that an updated document would not be linked to previous searches.
- Backward privacy prevents information leakage from deleted data.

  Type I Reveals the timestamp of matched inserted files, the total number of updates, and matching documents associated with w.

  Type II Additionally reveals timestamp of updates.

  Type III Additionally reveals the type of updates, i.e., insert or delete.

Verifiable Dynamic Symmetric Searchable Encryption

A malicious server can send back an incorrect or incomplete result.
If a DSSE scheme can detect a malicious server behaviour, it is called Verifiable DSSE.

Our scheme

Our scheme, is a Verifiable Dynamic Symmetric Searchable Encryption (VDSSE) scheme, secure against an active adversary, which achieves both forward and backward privacy of type II.

Exipnos: An efficient VDSSE scheme

How it works..

Our scheme is based on the principle of additive secret sharing.
A keyword w is assigned a secret s, which is then split into a number of shares equal to the total number of data records containing the keyword plus one.

$$DB[w] = (\mathsf{id}_1, \cdots, \mathsf{id}_{n-1})$$

No Random

obfuscate

$$s = \underbrace{r_{\mathsf{id}_1} \oplus \cdots \oplus r_{\mathsf{id}_{n-1}}}_{\text{Random}} \oplus r$$

How it works: Setup phase

Client side

$$w_1 \longleftrightarrow s \leftarrow 1^\lambda$$

random

$$K_1, K_2 = F(k, w_1)$$

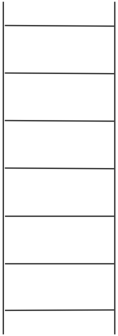
$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3)$$

r_{id_1} random

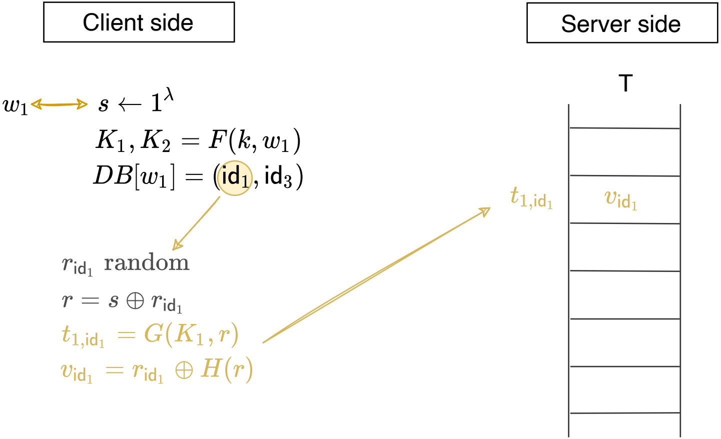
$$r = s \oplus r_{\mathsf{id}_1}$$

Server side

T



How it works: Setup phase



Client side

Server side

$$w_1 \longleftrightarrow s \leftarrow 1^\lambda$$

$$K_1, K_2 = F(k, w_1)$$

$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3)$$

r_{id_1} random

$$r = s \oplus r_{\mathsf{id}_1}$$

$$t_{1,\mathsf{id}_1} = G(K_1, r)$$

$$v_{\mathsf{id}_1} = r_{\mathsf{id}_1} \oplus H(r)$$

How it works: Setup phase

Client side

$w_1 \longleftrightarrow s \leftarrow 1^\lambda$

$K_1, K_2 = F(k, w_1)$

$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3)$

r_{id_1} random

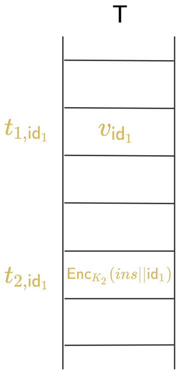
$r = s \oplus r_{\mathsf{id}_1}$

$t_{1,\mathsf{id}_1} = G(K_1, r)$

$v_{\mathsf{id}_1} = r_{\mathsf{id}_1} \oplus H(r)$

$t_{2,\mathsf{id}_1} = G(K_1, r_{\mathsf{id}_1})$

Server side



How it works: Setup phase

Client side

$$w_1 \longleftrightarrow s \leftarrow 1^\lambda$$

$$K_1, K_2 = F(k, w_1)$$

$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3)$$

$$\downarrow$$

r_{id_3} random

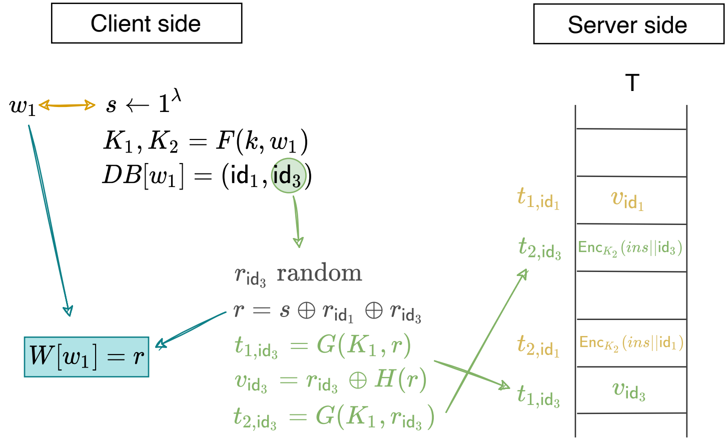
$r = s \oplus r_{\mathsf{id}_1} \oplus r_{\mathsf{id}_3}$

Server side

T

t_{1,id_1}	v_{id_1}
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$

How it works: Setup phase



Client side

$w_1 \longleftrightarrow s \leftarrow 1^\lambda$

$K_1, K_2 = F(k, w_1)$

$DB[w_1] = (\text{id}_1, \text{id}_3)$

r_{id_3} random

$r = s \oplus r_{\text{id}_1} \oplus r_{\text{id}_3}$

$t_{1,\text{id}_3} = G(K_1, r)$

$v_{\text{id}_3} = r_{\text{id}_3} \oplus H(r)$

$t_{2,\text{id}_3} = G(K_1, r_{\text{id}_3})$

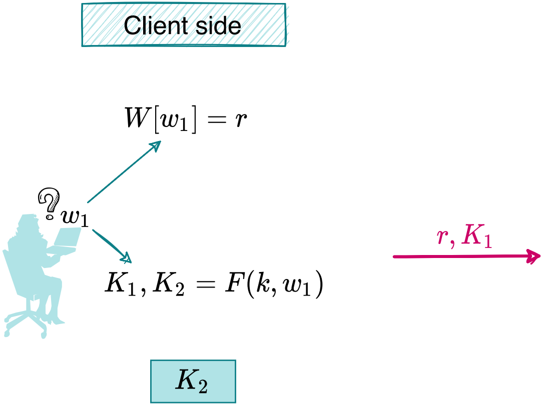
$\boxed{W[w_1] = r}$

Server side

T

t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\text{Enc}_{K_2}(ins\|\|\text{id}_3)$
t_{2,id_1}	$\text{Enc}_{K_2}(ins\|\|\text{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Query phase



Client side

$$W[w_1] = r$$

w_1

$$K_1, K_2 = F(k, w_1)$$

$$\boxed{K_2}$$

Server side

r, K_1

	T
t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Search phase

Client side

Server side

$$W[w_1] = r$$

$$r, K_1 \downarrow$$

$$G(K_1, r) = t_{1,\text{id}_3}$$

$$\boxed{r = s \oplus r_{\text{id}_1} \oplus r_{\text{id}_3}}$$

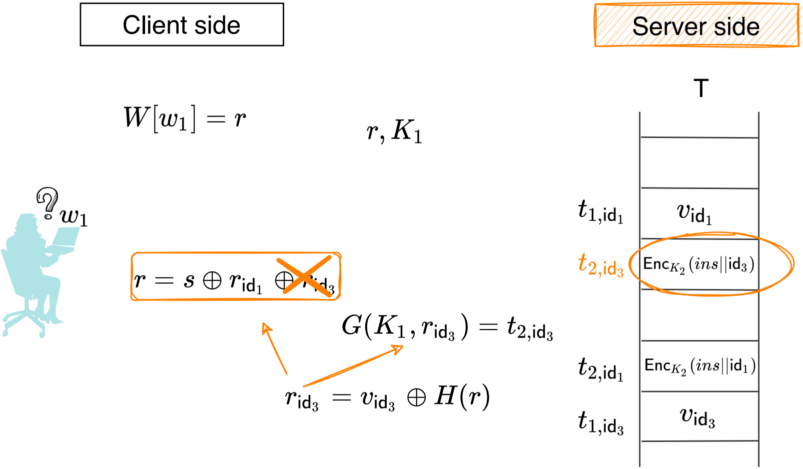
$$r_{\text{id}_3} = v_{\text{id}_3} \oplus H(r)$$

T

t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\text{Enc}_{K_2}(ins\|\|\text{id}_3)$
t_{2,id_1}	$\text{Enc}_{K_2}(ins\|\|\text{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Search phase

Cryptography Research Centre



Client side

Server side

$W[w_1] = r$

r, K_1

w_1

$\boxed{r = s \oplus r_{\text{id}_1} \oplus \cancel{r_{\text{id}_3}}}$

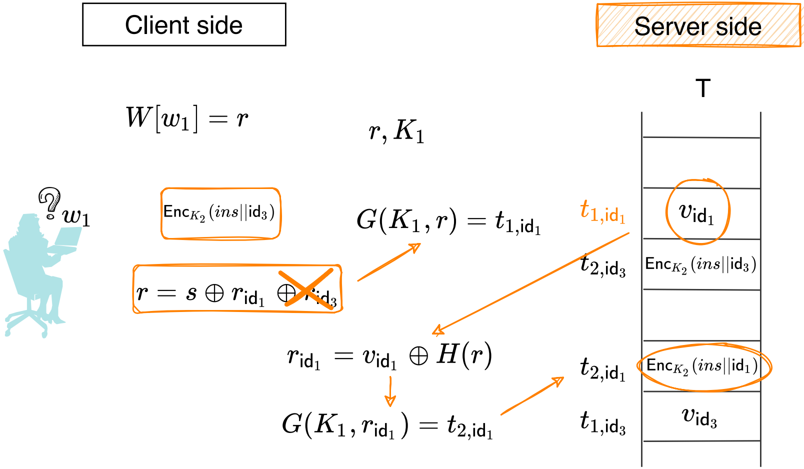
$G(K_1, r_{\text{id}_3}) = t_{2,\text{id}_3}$

$r_{\text{id}_3} = v_{\text{id}_3} \oplus H(r)$

T

t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\text{Enc}_{K_2}(ins \| \text{id}_3)$
t_{2,id_1}	$\text{Enc}_{K_2}(ins \| \text{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Search phase



Client side

Server side

$W[w_1] = r$

r, K_1

w_1

$\boxed{\mathsf{Enc}_{K_2}(ins||\mathsf{id}_3)}$

$\boxed{r = s \oplus r_{\mathsf{id}_1} \cancel{\oplus r_{\mathsf{id}_3}}}$

$G(K_1, r) = t_{1,\mathsf{id}_1}$

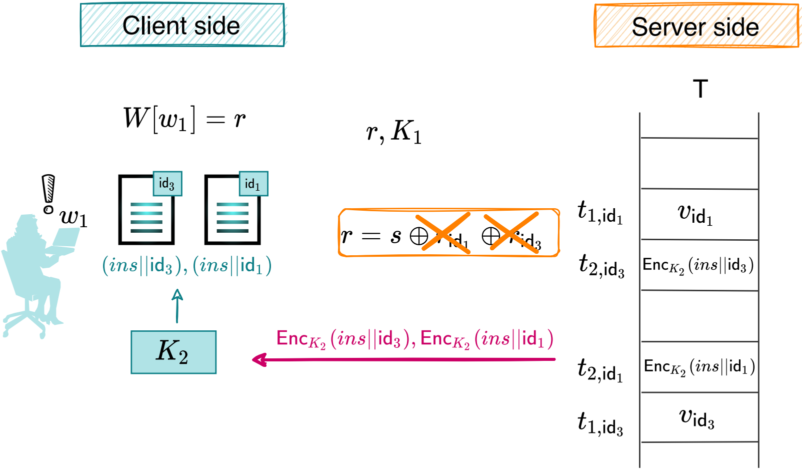
$r_{\mathsf{id}_1} = v_{\mathsf{id}_1} \oplus H(r)$

$G(K_1, r_{\mathsf{id}_1}) = t_{2,\mathsf{id}_1}$

T

t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Search phase



Client side

Server side

$W[w_1] = r$

r, K_1

w_1

id_3 id_1

$(ins||\mathsf{id}_3), (ins||\mathsf{id}_1)$

K_2

$r = s \oplus \cancel{r_{\mathsf{id}_1}} \oplus \cancel{r_{\mathsf{id}_3}}$

$\mathsf{Enc}_{K_2}(ins||\mathsf{id}_3), \mathsf{Enc}_{K_2}(ins||\mathsf{id}_1)$

T

t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Update phase - file insertion

Client side

$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3, \mathsf{id}_8)$$

$W[w_1] = r$ $\qquad$ $K_1, K_2 = F(k, w_1)$

↓

r_{id_8} random

$$r = s \oplus r_{\mathsf{id}_1} \oplus r_{\mathsf{id}_3} \oplus r_{\mathsf{id}_8}$$

↓

$$\boxed{W[w_1] = r}$$

Server side

	T
t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins \Vert \mathsf{id}_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins \Vert \mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Update phase - file insertion

Client side

$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3, \mathsf{id}_8)$$

$$\boxed{W[w_1] = r}$$

$$K_1, K_2 = F(k, w_1)$$

r_{id_8} random

$$r = s \oplus r_{\mathsf{id}_1} \oplus r_{\mathsf{id}_3} \oplus r_{\mathsf{id}_8}$$

$$t_{1,\mathsf{id}_8} = G(K_1, r)$$

$$v_{\mathsf{id}_8} = r_{\mathsf{id}_8} \oplus H(r)$$

$$t_{2,\mathsf{id}_8} = G(K_1, r_{\mathsf{id}_8})$$

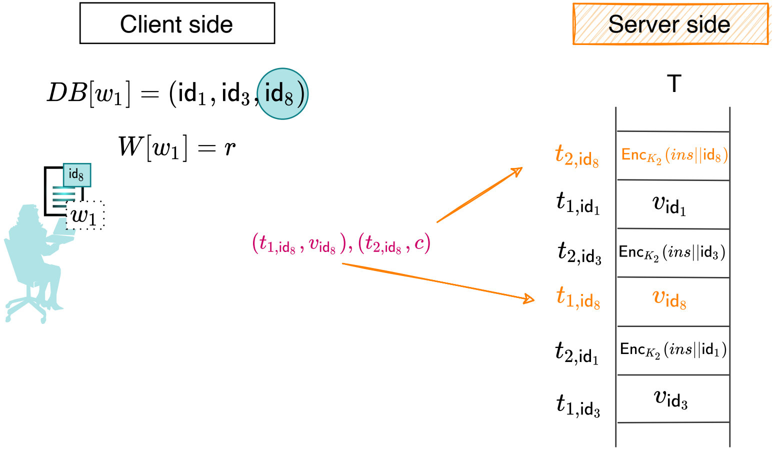
$$c = \mathsf{Enc}_{K_2}(ins||\mathsf{id}_8)$$

$$\xrightarrow{(t_{1,\mathsf{id}_8}, v_{\mathsf{id}_8}), (t_{2,\mathsf{id}_8}, c)}$$

Server side

	T
t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Update phase - file insertion



Client side

$$DB[w_1] = (\mathsf{id}_1, \mathsf{id}_3, \mathsf{id}_8)$$

$$W[w_1] = r$$

$(t_{1,\mathsf{id}_8}, v_{\mathsf{id}_8}), (t_{2,\mathsf{id}_8}, c)$

Server side

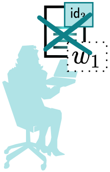
	T
t_{2,id_8}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_8)$
t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{1,id_8}	v_{id_8}
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

How it works: Update phase - file deletion

Client side

$$DB[w_1] = (\mathsf{id}_1, \cancel{\mathsf{id}_3})$$

$$\boxed{W[w_1] = r} \qquad K_1, K_2 = F(k, w_1)$$



$r_{\mathsf{id}'_3}$ random

$$r = s \oplus r_{\mathsf{id}_1} \oplus r_{\mathsf{id}_3} \oplus r_{\mathsf{id}'_3}$$

$$t_{1,\mathsf{id}'_3} = G(K_1, r)$$

$$v_{\mathsf{id}'_3} = r_{\mathsf{id}'_3} \oplus H(r)$$

$$t_{2,\mathsf{id}'_3} = G(K_1, r_{\mathsf{id}'_3})$$

$$c = \mathsf{Enc}_{K_2}(del||\mathsf{id}'_3)$$

$$\xrightarrow{(t_{1,\mathsf{id}'_3}, v_{\mathsf{id}'_3}), (t_{2,\mathsf{id}'_3}, c)}$$

Server side

	T
t_{1,id'_3}	$v_{\mathsf{id}'_3}$
t_{1,id_1}	v_{id_1}
t_{2,id_3}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_3)$
t_{2,id'_3}	$\mathsf{Enc}_{K_2}(del\|\|\mathsf{id}'_3)$
t_{2,id_1}	$\mathsf{Enc}_{K_2}(ins\|\|\mathsf{id}_1)$
t_{1,id_3}	v_{id_3}

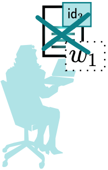
How it works: Verifiable scheme

Client side

Server side

$$DB[w_1] = (\mathsf{id}_1, \cancel{\mathsf{id}_3})$$

$$\boxed{W[w_1] = r} \quad K_1, K_2 = F(k, w_1)$$



$r_{\mathsf{id}'_3}$ random

$$r = s \oplus r_{\mathsf{id}_1} \oplus r_{\mathsf{id}_3} \oplus r_{\mathsf{id}'_3}$$

$$t_{1,\mathsf{id}'_3} = G(K_1, r)$$

$$v_{\mathsf{id}'_3} = r_{\mathsf{id}'_3} \oplus H(r)$$

$$t_{2,\mathsf{id}'_3} = G(K_1, r_{\mathsf{id}'_3})$$

$$c = \mathsf{Enc}_{K_2}(del || \mathsf{id}'_3 || r_{id'_3})$$

$$\xrightarrow{(t_{1,\mathsf{id}'_3}, v_{\mathsf{id}'_3}), (t_{2,\mathsf{id}'_3}, c)}$$

	T
t_{1,id'_3}	$v_{\mathsf{id}'_3}$
t_{1,id_1}	v_{id_1}
t_{2,id_3}	c_{id_3}
t_{2,id'_3}	$c_{\mathsf{id}'_3}$
t_{2,id_1}	c_{id_1}
t_{1,id_3}	v_{id_3}

Comparison with the state of the art

Performance comparison of DSSE schemes

Cryptography Research Centre

Scheme		Computation			Communication			Client	Security		Ver.
		query	search	update	search	update	RT	storage	FP	BP	
Π^{dyn}	Cash et al. (2014)	$O(1)$	$O(a_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(1)$	✗	✗	✗
$\sum o\phi o\varsigma$	Bost (2016)	$O(1)$	$O(a_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	✗	✗
Moneta		–	$\tilde{O}(a_w \log N + \log^3 N)$	$\tilde{O}(\log^2 N)$	$\tilde{O}(a_w \log N + \log^3 N)$	$\tilde{O}(\log^3 N)$	3	$O(1)$	✓	Type 1	✗
Fides		$O(a_w)$	$O(a_w)$	$O(1)$	$O(a_w)$	$O(1)$	2	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 2	✗
Diana	Bost et al. (2017)	$O(\log a_w)$	$O(a_w)$	$O(\log a_w)$	$O(n_w + \log a_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	✗	✗
Diana$_{del}$		$O(d_w \log a_w)$	$O(a_w)$	$O(\log a_w)$	$O(n_w + d_w \log a_w)$	$O(1)$	2	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 3	✗
Janus		$O(1)$	$O(n_w d_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 3	✗
Scheme	Etemad et al. (2018)	$O(n_w)$	$O(a_w/p)$	$O(1)$	$O((a_w + n_w)/p)$	O(1)	2	$O(\lvert\mathbf{W}\rvert + D)$	✓	✗	✗
Orion		–	$O(n_w \log^2 N)$	$O(\log^2 N)$	$O(n_w \log^2 N)$	$O(\log^2 N)$	$O(\log N)$	$O(1)$	✓	Type 1	✗
Mitra	Chamani et al. (2018)	$O(a_w)$	$O(a_w)$	$O(1)$	$O(a_w)$	$O(1)$	2	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 2	✗
Horus		–	$O(n_w \log d_w \log N)$	$O(\log^2 N)$	$O(n_w \log d_w \log N)$	$O(\log^2 N)$	$O(\log d_w)$	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 3	✗
SD$_a$	Demertzis et al. (2020)	–	$O(a_w + \log N)$	$O(\log N)$	$O(a_w + \log N)$	$O(\log N)$	2	$O(1)$	✓	Type 2	✗
SD$_d$		–	$O(a_w + \log N)$	$O(\log^3 N)$	$O(a_w + \log N)$	$O(\log N)$	2	$O(1)$	✓	Type 2	✗
Janus++	Sun et al. (2018)	$O(\log D)$	$O(n_w d)$	$O(d)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 3	✗
Aura	Sun et al. (2021)	$O(\log D)$	$O(n_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert D)$	✓	Type 2	✗
Fb-dsse	Zuo et al. (2019)	$O(1)$	$O(a_w)$	$O(1)$	$O(1)$	$O(n_w)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 1^-	✗
vsps	Bost et al. (2016)	$O(1)$	$O(m \log^3 N)$	$O(\log^2 N)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert D)$	✓	✗	✓
vfsse	Zhang et al. (2019)	$O(1)$	$O(a_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	✗	✓
Our scheme		$O(1)$	$\mathrm{O}(a_w)$	$O(1)$	$\mathrm{O}(a_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 2	✓

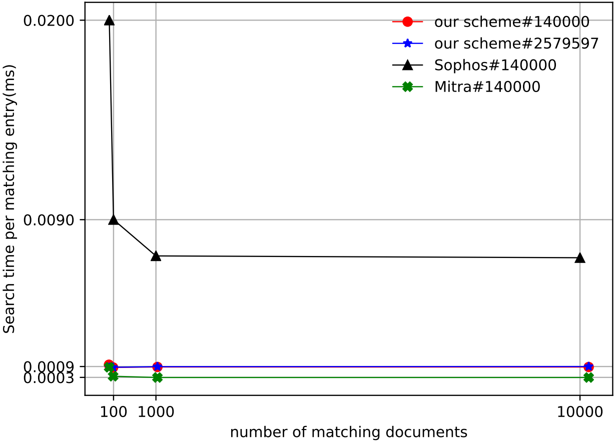
$D = \#$ documents, $|\mathbf{W}| = \#$ keywords, $N = \#$ keyword/document pairs, $a_w = \#$ times queried keyword w was added to the database, $d_w = \#$ deleted entries for w, $p = \#$ processors, $n_w = \#$ documents matching w, i.e., $n_w = a_w - d_w$, $d = \max_w d_w \cdot n_w$ is the size of search result matching w, $- =$ much larger than other listed schemes. RT is roundtrip. Here, we assume two rounds of result-hiding scheme is equal to one round of result-revealing scheme. FP is forward privacy, BP is backward privacy. $\widetilde{O}$ notation hides polylogarithmic factors.

Performance comparison of DSSE schemes

Scheme		Computation			Communication			Client	Security		Ver.
		query	search	update	search	update	RT	storage	FP	BP	
$\sum o\phi o\varsigma$	Bost (2016)	$O(1)$	$O(a_w)$	$O(1)$	$O(n_w)$	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	✗	✗
Mitra	Chamani et al. (2018)	$O(a_w)$	$O(a_w)$	$O(1)$	$O(a_w)$	$O(1)$	2	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 2	✗
Our scheme		$O(1)$	O(a_w)	$O(1)$	O(a_w)	$O(1)$	1	$O(\lvert\mathbf{W}\rvert \log D)$	✓	Type 2	✓

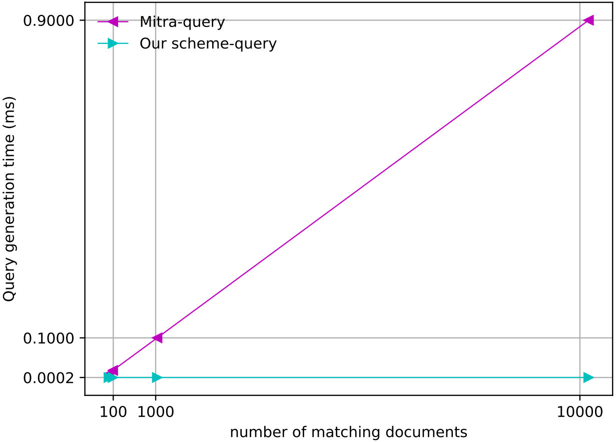
- Implementation in C on a laptop with a 2.6 GHz Intel Core i7 processor, 16 GB of RAM, and 500 GB flash storage.
- Security parameter $\lambda = 128$.
- AES is adopted as the applied symmetric encryption algorithm.
- 256-bit collision resistant hash function.
- We use the dataset 20 *Newsgroups* (http://qwone.com/ jason/20Newsgroups/):
  - 61,187 keywords,
  - 20,000 documents,
  - 2,579,597 keyword/document.

Performance comparison of DSSE schemes



Performance comparison of DSSE schemes

Cryptography
Research
Centre



Thank you!

References I

Bost, R. (2016). $\Sigma o\varphi o\varsigma$: Forward Secure Searchable Encryption. In *ACM SIGSAC Conf. on Computer and Communications Security*, pages 1143–1154. ACM.

Bost, R., Fouque, P.-A., and Pointcheval, D. (2016). Verifiable dynamic symmetric searchable encryption: Optimality and forward security. *IACR Cryptology ePrint Arch.*, 2016:62.

Bost, R., Minaud, B., and Ohrimenko, O. (2017). Forward and backward private searchable encryption from constrained cryptographic primitives. In *ACM SIGSAC Conference on Computer and Communications Security*, pages 1465–1482. ACM.

Cash, D., Jaeger, J., Jarecki, S., Jutla, C. S., Krawczyk, H., Rosu, M.-C., and Steiner, M. (2014). Dynamic searchable encryption in very-large databases: data structures and implementation. In *NDSS*, volume 14, pages 23–26.

References II

Chamani, J. G., Papadopoulos, D., Papamanthou, C., and Jalili, R. (2018). New constructions for forward and backward private symmetric searchable encryption. In *ACM SIGSAC Conf. on Computer and Communications Security*, pages 1038–1055.

Demertzis, I., Chamani, J. G., Papadopoulos, D., and Papamanthou, C. (2020). Dynamic searchable encryption with small client storage. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society.

Etemad, M., Küpçü, A., Papamanthou, C., and Evans, D. (2018). Efficient dynamic searchable encryption with forward privacy. *PET - Privacy Enhancing Technologies*, (1):5–20.

Sun, S., Yuan, X., Liu, J. K., Steinfeld, R., Sakzad, A., Vo, V., and Nepal, S. (2018). Practical backward-secure searchable encryption from symmetric puncturable encryption. In *ACM SIGSAC Conf. on Computer and Communications Security*, pages 763–780.

References III

Sun, S.-F., Steinfeld, R., Lai, S., Yuan, X., Sakzad, A., Liu, J., Nepal, S., and Gu, D. (2021). Practical non-interactive searchable encryption with forward and backward privacy. In *Proceedings 2021 Network and Distributed System Security Symposium. Internet Society*.

Zhang, Z., Wang, J., Wang, Y., Su, Y., and Chen, X. (2019). Towards efficient verifiable forward secure searchable symmetric encryption. In *European Symposium on Research in Computer Security*, pages 304–321.

Zuo, C., Sun, S.-F., Liu, J. K., Shao, J., and Pieprzyk, J. (2019). Dynamic searchable symmetric encryption with forward and stronger backward privacy. In *European Symposium on Research in Computer Security*, pages 283–303.

TII
Technology
Innovation
Institute

Cryptography
Research
Centre

tii.ae