Efficiency Lower Bounds and Optimal Constructions of Searchable Encryption

Raphael Bost[1,2]

[1]Direction Générale de l'Armement - Maîtrise de l'Information

[2]Université de Rennes 1

ESSA2, 10/07/2018, Bertinoro

Security *vs.* Efficiency

Searchable encryption is all about a security-performance tradeoff

Security *vs.* Efficiency

Searchable encryption is all about a security-performance tradeoff

With searchable encryption, as in life, nothing is free!

Efficiency

Many possible measurements:

- Computational complexity
- Communication complexity
- Number of interactions
- Size of the encrypted database
- Size of the client's state
- Memory locality & read efficiency

Security

We can evaluate the security

- formally: from the leakage in the security proofs
- practically: from actual attacks (*e.g.* leakage-abuse attacks)

This presentation

Lower bounds on the efficiency of:

- static searchable encryption schemes hiding the repetition of search queries;
- dynamic searchable encryption schemes with forward-private updates;
- dynamic searchable encryption schemes secure against malicious adversaries.

This presentation

We restricted ourselves to:

- symmetric searchable encryption (SSE)
- single-keyword search queries
- database structure: atomic keyword/document pairs (a.k.a. entries)

Security model

- Indistinguishability-based security definition: two executions with the same leakage cannot be distinguished by an adversary
- Only the non-adaptive version of the definition is needed here

Notations

- $N = |\mathrm{DB}|$: total number of entries
- K: number of distinct keywords
- $|\mathrm{DB}(w)|$: number of entries matching w
- $H = (\mathrm{DB}, r_1, \ldots, r_i)$: query history (r_i can be a search query, or an update query)

Schemes hiding the search pattern

- Static schemes only revealing the number of results of a query (hides the repetition of queries — the search pattern)
- Related to ORAM (# results of each query is 1) Called File-ORAM in [ACNPRS'17]
- ORAM lower bound [GO'96]: $\Omega\left(\frac{\log N}{\log \sigma}\right)$

Lower bound on search-pattern-hiding SSE

Theorem

Let Σ be a static SSE scheme leaking (N, K) and $|\mathrm{DB}(w)|$. Then the complexity of the search protocol is

$$\Omega\left(\frac{\log \binom{\overline{N}(H,w)}{n_w}}{\log|\sigma| \cdot \log\log \binom{\overline{N}(H,w)}{n_w}}\right)$$

where

$$\overline{N}(H, w) = |\mathrm{DB}| - \sum_{\substack{j=1 \\ |\mathrm{DB}(w_j)| \neq |\mathrm{DB}(w)|}}^{i} |\mathrm{DB}(w_j)|.$$

Explanations

- Suppose the client queries w and w' with $|\mathrm{DB}(w)| \neq |\mathrm{DB}(w')|$. The adversary knows from the leakage that $w \neq w'$.
- As $w \neq w'$, the adversary knows that the accessed entries will be different. Hence the term in $\overline{N}$.
- The order in which the entries are touched does not matter. Hence the binomial coefficient.
- The proof essentially proceeds an in [GO'96].
- The log log term in an artefact.

Tightness of the lower bound

w_0	2
w_1	3
w_2	1
w_3	56
w_4	3
$\vdots$	$\vdots$

OMap for w s.t. $|\mathrm{DB}(w)| = 1$

OMap for w s.t. $|\mathrm{DB}(w)| = 2$

OMap for w s.t. $|\mathrm{DB}(w)| = 3$

$\dots$

OMap for w s.t. $|\mathrm{DB}(w)| = 56$

$\dots$

Query complexity of an OMap of size n: $\mathcal{O}(\log n)$.
The search complexity of the SE construction is $\mathcal{O}(\log K)$.

Tightness of the lower bound

w_0	2
w_1	3
w_2	1
w_3	56
w_4	3
$\vdots$	$\vdots$

OMap for w s.t. $|\mathrm{DB}(w)| = 1$

OMap for w s.t. $|\mathrm{DB}(w)| = 2$

OMap for w s.t. $|\mathrm{DB}(w)| = 3$

...

OMap for w s.t. $|\mathrm{DB}(w)| = 56$

...

Query complexity of an OMap of size n: $\mathcal{O}(\log n)$.
The search complexity of the SE construction is $\mathcal{O}(\log K)$.

Tightness of the lower bound

w_0	2
w_1	3
w_2	1
w_3	56
w_4	3
$\vdots$	$\vdots$

OMap for w s.t. $|\mathrm{DB}(w)| = 1$

OMap for w s.t. $|\mathrm{DB}(w)| = 2$

OMap for w s.t. $|\mathrm{DB}(w)| = 3$

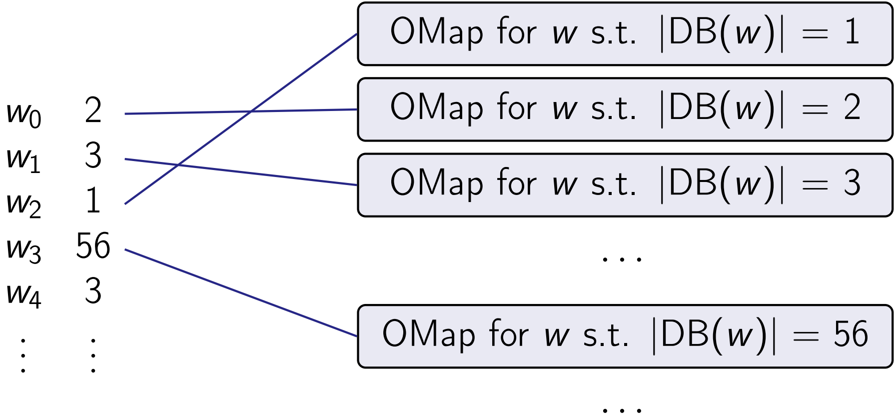
$\ldots$

OMap for w s.t. $|\mathrm{DB}(w)| = 56$

$\ldots$

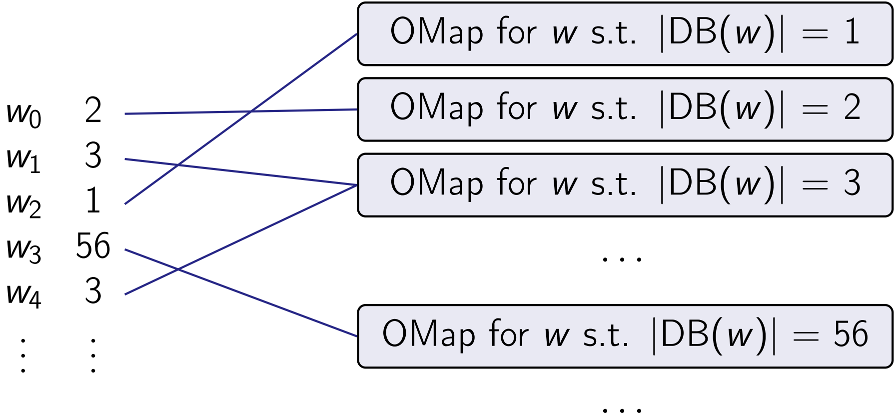
Query complexity of an OMap of size n: $\mathcal{O}(\log n)$.
The search complexity of the SE construction is $\mathcal{O}(\log K)$.

Tightness of the lower bound



Query complexity of an OMap of size n: $\mathcal{O}(\log n)$.
The search complexity of the SE construction is $\mathcal{O}(\log K)$.

Tightness of the lower bound



Query complexity of an OMap of size n: $\mathcal{O}(\log n)$.
The search complexity of the SE construction is $\mathcal{O}(\log K)$.

Tightness of the lower bound

The previous construction breaks the lower bound when $K \ll N$ (common case).
During setup, the *profile* of the database is leaked: $(K_i)_{i=1}$ where $K_i = \#\{w \text{ s.t. } |\mathrm{DB}(w)| = i\}$.

With a small additional leakage, we can break the lower bound on SP-hiding SSE.

Forward Privacy

File injection attacks [ZPK'16]

Leaking information about the updated keywords leads to devastating adaptive attacks.

Forward privacy

An update does not leak any information on the updated keywords (often, no information at all)

Introduced in [SPS'14], must have security feature for modern dynamic schemes

The cost of forward privacy

Scheme	Computation: Search	Computation: Update	Client Storage	FP	
Π^{dyn}	$\mathcal{O}(a_w)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	✗	
SPS	$\mathcal{O}(a_w + \log N)$ $\mathcal{O}\left(n_w \log^3 N\right)$	$\mathcal{O}\left(\log^2 N\right)$	$\mathcal{O}(N^\alpha)$	✓	Supports deletions well
$\Sigma o\phi o\varsigma$	$\mathcal{O}(a_w)$	$\mathcal{O}(1)$	$\mathcal{O}(K)$	✓	TDP
EKPE	$\mathcal{O}(a_w)$	$\mathcal{O}(1)$	$\mathcal{O}(K)$	✓	write during search
KKLPK	$\mathcal{O}(a_w)$	$\mathcal{O}(1)$	$\mathcal{O}(K)$	✓	
Diana	$\mathcal{O}(a_w)$	$\mathcal{O}(\log a_w)$	$\mathcal{O}(K)$	✓	CPRF
FAST	$\mathcal{O}(a_w)$	$\mathcal{O}(1)$	$\mathcal{O}(K)$	✓	

Lower bound on forward-private SE

Theorem

Let Σ be a forward-private SSE scheme. Then the sum of the amortized complexity of the search and update protocols is

$$\Omega\left(\frac{\log K}{\log|\sigma| \cdot \log\log K}\right)$$

Lower bound on forward-private SE

Theorem

Let Σ be a forward-private SSE scheme. Then the sum of the amortized complexity of the search and update protocols is

$$\Omega\left(\frac{\log K}{\log|\sigma| \cdot \log\log K}\right)$$

Sloppy proof

There might be some issues with the proof.
Details are important (thanks Tarik!).

Tightness of the FP lower bound

- $\Sigma o\phi o\varsigma$, KKLPK, EKPE and FAST show that the lower bound is tight ($|\sigma| = K$).
- FAST shows that the lower bounds can be reached relying only on a PRF, without rewriting the DB during the search algorithm to 'cache' the results.
- Outsource the client's counter map using an oblivious map data structure.
  $|\sigma| = \mathcal{O}(1)$, $\mathcal{O}(\log K)$ search & update complexity.
- Open question: is there a middle point?
  e.g. $|\sigma| = \mathcal{O}\left(\sqrt{K}\right)$ & $\mathcal{O}(1)$ update complexity.

Verifiable Searchable Encryption

The security against malicious adversaries can be split in two parts.

Confidentiality

No information leaks about the DB/query.
Often simple (single interaction).

Soundness (integrity)

The server cannot return incorrect results.
Does not depend on the leakage.

Memory checking

Problem

How to outsource memory to an untrusted party, while ensuring authenticity and using limited trusted local storage?

Lower bound [DNRV'09]: a memory checker outsourcing n values, with $|\sigma| < n^{1-\varepsilon}$ for some $\varepsilon > 0$ has computational overhead

$$\Omega\left(\frac{\log n}{\log \log n}\right).$$

VSSE lower bound

Using a simple reduction from memory checking, we get a lower bound on verifiable SSE schemes.

Theorem

Let Σ be a dynamic verifiable SSE scheme with $|\sigma| < K^{1-\varepsilon}$ for some $\varepsilon > 0$. Then the computational complexity of the search or of the update protocol is

$$\Omega\left(\frac{\log K}{\log\log K}\right).$$

A practical VSSE lower bound

Using a less generic result on hash-based memory checker by Tamassia and Triandopolous [TT'05], we can improve the lower bound to

$$\Omega\left(\log \frac{K}{|\sigma|}\right).$$

Why is this interesting?

- This lower bound does not depend on the leakage.
- If a scheme, hides the search pattern, or is forward-private, we should be able to get verifiability for free: $\Omega\left(\frac{\log K}{\log|\sigma| \cdot \log\log K}\right)$ *vs.* $\Omega\left(\log\frac{K}{|\sigma|}\right)$.
- And we can …

Set hash functions

Some kind of incremental hashing:

- The value of the hash does not depend on the order
- It is easy to compute $\mathcal{H}(A \cup \{x\})$ from $\mathcal{H}(A)$ and x.
  More generally $\mathcal{H}(A \cup B) = \mathcal{H}(A) +_{\mathcal{H}} \mathcal{H}(B)$
- It is easy to compute $\mathcal{H}(A \setminus \{x\})$ from $\mathcal{H}(A)$ and x.
  More generally $\mathcal{H}(A \setminus B) = \mathcal{H}(A) -_{\mathcal{H}} \mathcal{H}(B)$

Efficiently instantiable using elliptic curves

Collision resistance of set hash functions

It must be hard for an adversary to find two different sets hashing to the same value.

Definition of collision resistance

$$\begin{aligned}\mathsf{Adv}^{\mathrm{col}}_{\mathcal{H},A}(\lambda) = \mathbb{P}[K \xleftarrow{\$} \mathcal{K}, (S, S') \leftarrow A(K) :\\ S \neq S' \wedge \mathcal{H}_K(S) \equiv_{\mathcal{H}_K} \mathcal{H}_K(S')]\end{aligned}$$

$\mathcal{H}$ is collision resistant if $\mathsf{Adv}^{\mathrm{col}}_{\mathcal{H},A}(\lambda)$ is negligible in 1^λ.

Efficiently instantiable using elliptic curves

Generic VSSE

Two simple ideas:

1. For each keyword w, store H(DB(w)) in a table T
   When searching for w and returned the result set R, check that $H(R) = T$.
   When updating on w, update H(DB(w)) incrementally.
2. Outsource T using a verifiable oblivious map

Generic VSSE

- Additional client storage: $\mathcal{O}(1)$
- Additional server storage: $\mathcal{O}(K)$
- Computational overhead: $\mathcal{O}(\log K + |\mathrm{DB}(w)|)$
- Additional leakage: K (from the size of the OMap)

Can be applied to any forward-private scheme to make it verifiable

Conclusion

- Three lower bounds showing the tradeoffs between security and efficiency
- Can they be extended to a more general setting?
- Forward-private schemes: is there a lower bound on the locality? Which parameter does it involve?

Thank you!

Slides: https://r.bost.fyi/slides/essa2.pdf