Efficient Oblivious Substring Search via Architectural Support

Nicholas Mainardi Davide Sampietro Alessandro Barenghi Gerardo Pelosi

Email: name.surname@polimi.it

ACSAC 2020: December 10\textsuperscript{th}, 2020
Scenario: A data owner with a document collection $\mathbf{D} = \{D_1, \ldots, D_z\}$ of $z$ documents:

1. Outsources $\mathbf{D}$ in encrypted form to an untrusted server
2. Performs privacy-preserving substring search queries over $\mathbf{D}$

Privacy-Preserving Substring Search Queries

- Given a string $q$, find positions of all the repetitions of $q$ in each document of $\mathbf{D}$
- Privacy requirement: confidentiality of data involved in the query should be preserved

Leakage $\mathcal{L}$ of the protocol $\equiv$ information learnt by the untrusted server
We propose the first privacy-preserving substring search protocol with optimal communication cost and a limited leakage $\mathcal{L}$ amounting to:

- $n \approx \sum_{i=1}^{z} |D_i|$: size of the document collection $D$
- $m = |q|$: length of the searched string $q$
- $occ$: overall number of occurrences of $q$ in $D$

**Threat Model**

We consider a powerful malicious adversary $\mathcal{A}$

- $\mathcal{A}$ has full control of the untrusted server
- $\mathcal{A}$ is willing to learn as much information as possible
- $\mathcal{A}$ can arbitrarily tamper with the protocol execution
Our solution relies on **Intel SGX technology**

- Available in Intel CPUs since *skylake* micro-architecture (2015)
- Provide trusted execution environments, called **secure enclaves**, to securely run applications on an untrusted machine

  Secure enclave ≡ protected memory region:

  ![Diagram showing Intel SGX](image)

  - HW enforced access control + memory encryption → Guarantee confidentiality and integrity of code and data stored in an enclave
    - Even against the OS or the hypervisor
  - Computation over plaintext data → Minimal performance overhead
Intel SGX Security Weaknesses

- Crypto primitives employed in SGX are sound
- But SGX enclaves are vulnerable to several side channel attacks

Side channel attacks are split according to their effectiveness:

### Key Recovery Attacks
- Allow to recover per-CPU encryption keys → they completely subvert SGX
- They are mitigated by Intel 😊

### Memory Access Pattern Reconstruction Attacks
- They monitor accesses to memory pages and cache lines of the enclave application
- Data dependent memory accesses → Data leakage
- Excluded from SGX threat model! 😞
No countermeasure proposed so far can prevent or detect all existing side channel attacks.

Alternative approach: making an application oblivious \( \equiv \) data independent control-flow and memory accesses \( \rightarrow \) the information leaked through side-channels become useless for the attacker. How?

Existing solutions rely on an Oblivious RAM (ORAM) protocol.
No countermeasure proposed so far can prevent or detect all existing side channel attacks.

Alternative approach: making an application oblivious \(\equiv\) data independent control-flow and memory accesses \(\rightarrow\) the information leaked through side-channels become useless for the attacker. How?

Existing solutions rely on an Oblivious RAM (ORAM) protocol.
Combining ORAM and SGX

- ORAM client is moved inside the enclave
- Memory accesses of ORAM client leaked through SGX side channels
- Sufficient to subvert ORAM privacy guarantees

An ORAM with an oblivious client is needed
Combining ORAM and SGX

ORAM client is moved inside the enclave

- Memory accesses of ORAM client leaked through SGX side channels
- Sufficient to subvert ORAM privacy guarantees

An ORAM with an oblivious client is needed $\equiv$ Doubly Oblivious RAM
Our Solution

1. We design algorithms for substring search queries with data-independent control flow
2. We employ a DORAM to conceal the memory access pattern to data structures

ObSQRE, our SGX-based privacy-preserving substring search protocol

Comparison with existing protocols exhibiting similar privacy guarantees

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Computation Cost</th>
<th>Communication Cost</th>
<th>Adversary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ishimaki et al. [IY17]</td>
<td>$O((m + occ)n \log n)$</td>
<td>$O((m+occ) \log(n))$</td>
<td>Semi-honest</td>
</tr>
<tr>
<td>SA-ORAM [MB15]</td>
<td>$\Omega(m \log^5(n) + occ \log^2(n))$</td>
<td>$\Omega(m \log^5(n) + occ \log^2(n))$</td>
<td>Semi-honest</td>
</tr>
<tr>
<td>Qin et al. [QZZX20]</td>
<td>$O(m \cdot n)$</td>
<td>$O((m + occ)z)$</td>
<td>Semi-honest</td>
</tr>
<tr>
<td>Faust et al. [FHV18]</td>
<td>$\Omega((\frac{n}{m} + occ) \log(n))$</td>
<td>$O((m + occ) \log(n))$</td>
<td>Malicious</td>
</tr>
<tr>
<td>ObSQRE</td>
<td>$O((m + occ) \log^3(n))$</td>
<td>$O(m+occ)$</td>
<td>Malicious</td>
</tr>
</tbody>
</table>
Oblivious Substring Search Algorithm

Our algorithm relies on backwards search method:
- Its control flow depends only on $m$ and $occ$ ✓
- It employs 2 data structures $\tilde{L}$ and $SA$ constructed from $D \rightarrow$
  Accessed through DORAMs for obliviousness ✓

Our DORAMs

We propose doubly oblivious versions of 3 tree-based ORAMs:
- Path DORAM: performance improvements over existing doubly oblivious designs
- First doubly oblivious designs of Circuit and Ring ORAMs
- We enrich all our DORAMs with an integrity check mechanism based on Merkle Trees
ObSQRE

Setup

Data Owner

Encrypted data structures for D

Untrusted Server

Authenticated encryption to ensure integrity of outsourced data

Nicholas Mainardi
Remote Attestation allows data owner to:

1. Verify the proper setup of the enclave
2. Establish a secure communication channel with the enclave
Load

Data Owner

Untrusted Server

DORAM

$\langle \bar{L}, SA \rangle$

Secure Enclave

DORAMClient

Oblivious Substring Search

Integrity of index is verified upon decryption
Queries require only a single communication round.
Information Leakage

- $n$ is leaked by the size of $\tilde{L}$ and $SA$
- $m$ is leaked by the number of accesses to DORAMs storing $\tilde{L}$
- $occ$ is leaked by the number of accesses to DORAMs storing $SA$

ObSQRE is able to detect any tampering with data and computation

- Operations performed inside enclave cannot be affected by adversaries
- Only DORAM accesses are performed outside the enclave $\rightarrow$ integrity check mechanism ensures correctness of the retrieved data
Experimental Evaluation

Experimental Setup
- Intel Xeon E3-1220 equipped with SGX and 64 GiB RAM
- Ubuntu 16.04 Linux OS

DORAM Benchmarking

- All our DORAMs are faster than existing DORAMs
- Path DORAM is fastest among our DORAMs
ObSQRE Experimental Evaluation

Dataset

- **Chr**: 21\textsuperscript{st} human chromosome, alphabet size: 7
- **Prot**: set of human proteins, alphabet size: 25
- **Enron**: real-world email dataset, alphabet size: 91

Substring Search Query Performance

- Poly-logarithmic trend
- Alphabet size leads to performance
- Best DORAM: Circuit DORAM

Nicholas Mainardi
We evaluate our best solution on two real-world use cases:
- Look-up of a protein in the human genome (3GiB data)
- Look-up of short keywords over Enron emails (> 1 Gib data)

We compare ObSQRE with two less secure solutions:
- **SGX + Path ORAM**: secure solution if side channels excluded from threat model
- **No SGX, No ORAM**: solution with no security guarantees

<table>
<thead>
<tr>
<th>Dataset</th>
<th>m</th>
<th>#Occ.</th>
<th>ObSQRE (ms)</th>
<th>SGX + ORAM (ms)</th>
<th>no SGX + no ORAM (µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genome</td>
<td>3050</td>
<td>1</td>
<td>1019</td>
<td>347</td>
<td>167</td>
</tr>
<tr>
<td>Enron</td>
<td>5</td>
<td>2657</td>
<td>3.7</td>
<td>1.4</td>
<td>11.2</td>
</tr>
<tr>
<td>Enron</td>
<td>13</td>
<td>290</td>
<td>5.5</td>
<td>1.9</td>
<td>4.2</td>
</tr>
<tr>
<td>Enron</td>
<td>20</td>
<td>154</td>
<td>8.4</td>
<td>3</td>
<td>5.7</td>
</tr>
</tbody>
</table>

Practical performance, limited overhead 😊
Conclusions

- ObSQRE is the first privacy-preserving substring search protocol with optimal communication cost and limited information leakage.
- ObSQRE is secure against malicious adversaries.
- Security guarantees are based on Intel SGX technology and on our proposed DORAMs.
- ObSQRE exhibits practical performance on real-world use cases.

Future Works:

- Enable multiple simultaneous queries from distinct users.
- Enable pattern-matching queries.
[FHV18] Sebastian Faust, Carmit Hazay, and Daniele Venturi. 
Outsourced pattern matching. 

[IY17] Yu Ishimaki, Hiroki Imabayashi, and Hayato Yamana. 
Private Substring Search on Homomorphically Encrypted Data. 

Oblivious Substring Search with Updates. 

[QZZX20] Shiyue Qin, Fucai Zhou, Zongye Zhang, and Zifeng Xu. 
Privacy-preserving substring search on multi-source encrypted gene data. 