Efficient Oblivious Substring Search via Architectural Support

Nicholas Mainardi Davide Sampietro
Alessandro Barenghi Gerardo Pelosi

Email: name.surname@polimi.it

ACSAC 2020: December 10th, 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

Our Privacy-Preserving Substring Search Protocol

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 $\mathbf{D}$
- $m = |q|$: length of the searched string q
- occ: overall number of occurrences of q in $\mathbf{D}$

Threat Model

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

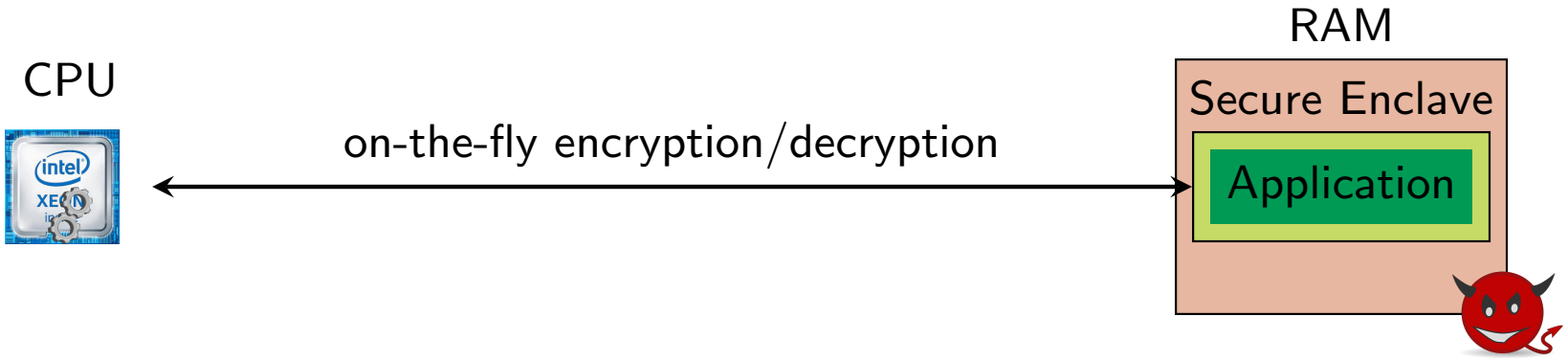
- $\mathcal{A}$ has full control of the untrusted server
- $\mathcal{A}$ is willing to learn as much information as possible
- $\mathcal{A}$ can arbitrarily tamper with the protocol execution

Intel Software Guard Extension (SGX)

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:



- 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 ☺

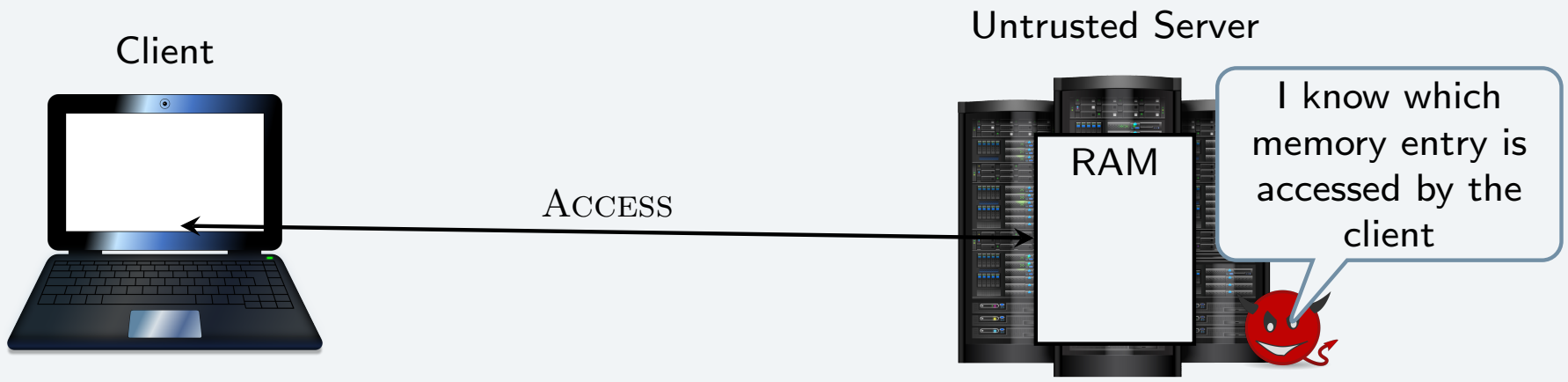
Memory Access Pattern Reconstruction Attacks

- They monitor accesses to memory pages and cache lines of the enclave application
- Data dependent memory accesses → Data leakage
- Excluded from SGX threat model! ☹

Dealing with SGX Side Channels

- 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

ORAM



Dealing with SGX Side Channels

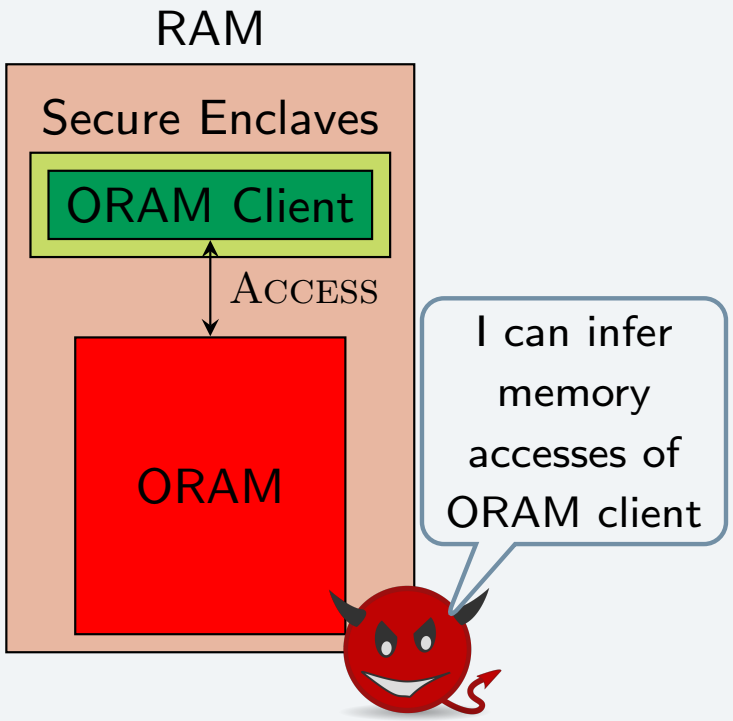
- No countermeasure proposed so far can prevent or detect all existing side channel attacks
- Alternative approach: making an application oblivious $\equiv$ data independent control-flow and memory accesses $\rightarrow$ the information leaked through side-channels become useless for the attacker. How?
- Existing solutions rely on an Oblivious RAM (ORAM) protocol

ORAM

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

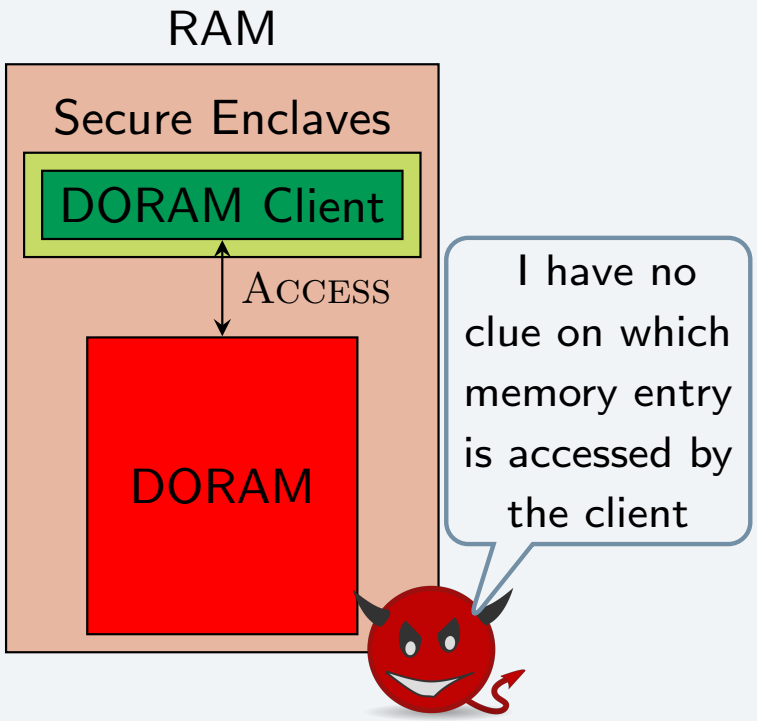


Combining ORAM and SGX

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 ≡ 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

Protocol	Computation Cost	Communication Cost	Adversary
Ishimaki et al. [IIY17]	$O((m+occ)n\log n)$	$O((m+occ)\log(n))$	Semi-honest
SA-ORAM [MB15]	$\Omega(m\log^5(n)+occ\log^2(n))$	$\Omega(m\log^5(n)+occ\log^2(n))$	Semi-honest
Qin et al. [QZZX20]	$O(m{\cdot}n)$	$O((m+occ)z)$	Semi-honest
Faust et al. [FHV18]	$\Omega((\frac{n}{m}+occ)\log(n))$	$O((m+occ)\log(n))$	Malicious
ObSQRE	$O((m+occ)\log^3(n))$	$O(m+occ)$	Malicious

Building Blocks of ObSQRE

Oblivious Substring Search Algorithm

Our algorithm relies on backwards search method:

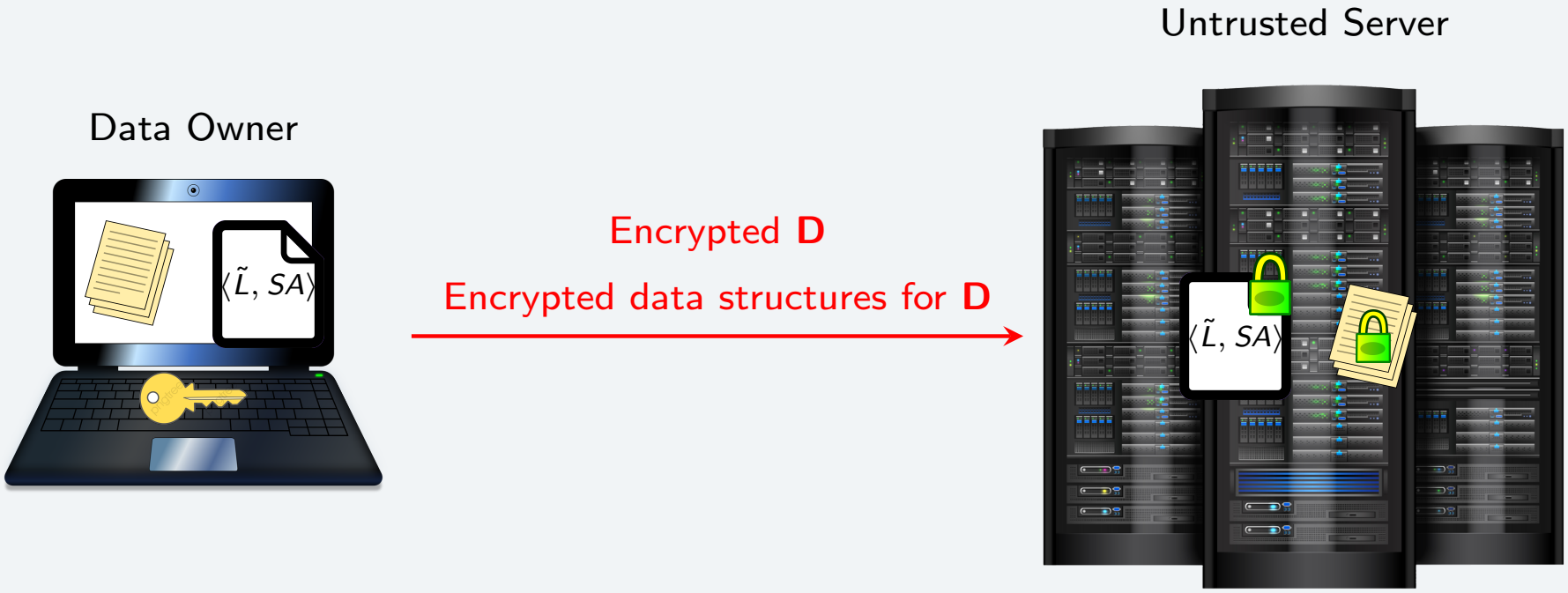
- Its control flow depends only on m and occ ✓
- It employs 2 data structures $\tilde{L}$ and SA constructed from $\mathbf{D} \rightarrow$ Accessed through DORAMs for obliviousness ✓

Our DORAMs

We propose doubly oblivious versions of 3 tree-based ORAMs:

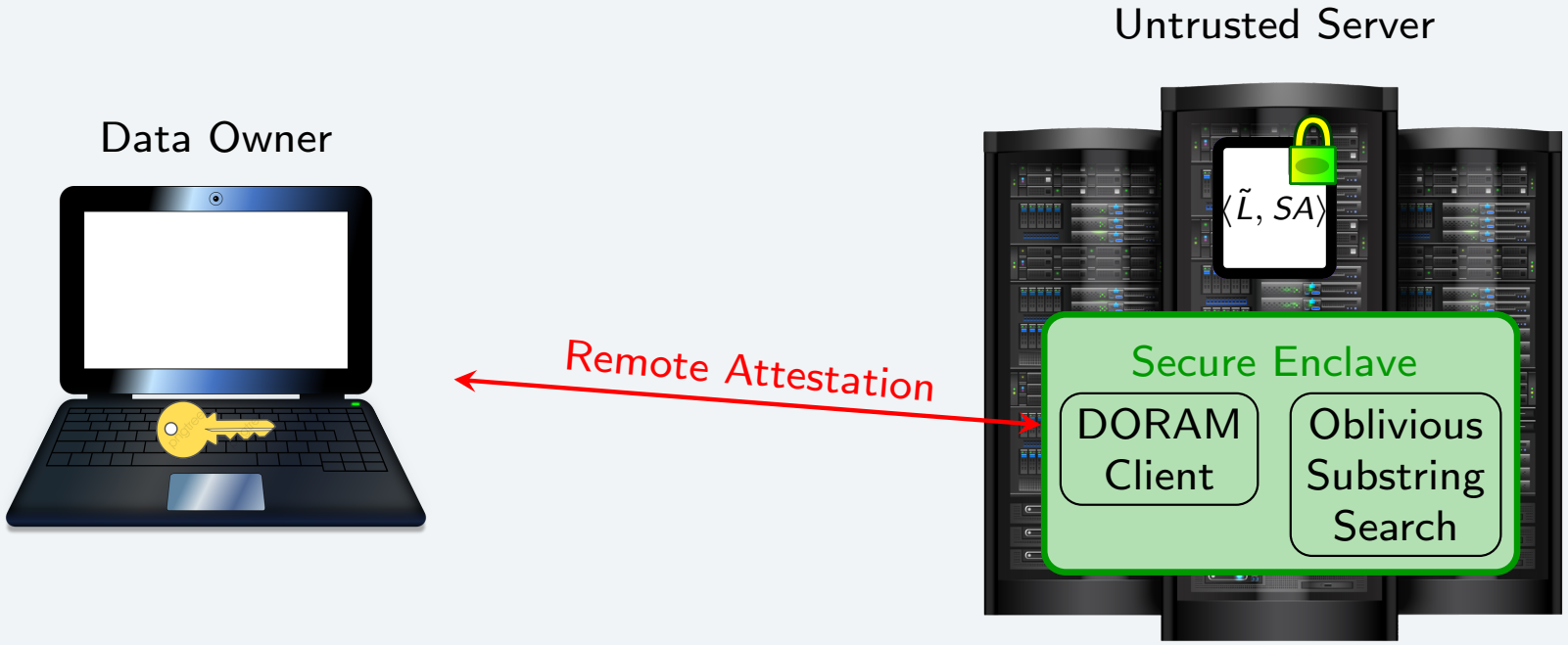
- Path DORAM: performance improvements over existing doubly oblivious designs
- First doubly oblivious designs of Circuit and Ring ORAMs
- We enrich all our DORAMs with an integrity check mechanism based on Merkle Trees

Setup



Authenticated encryption to ensure integrity of outsourced data

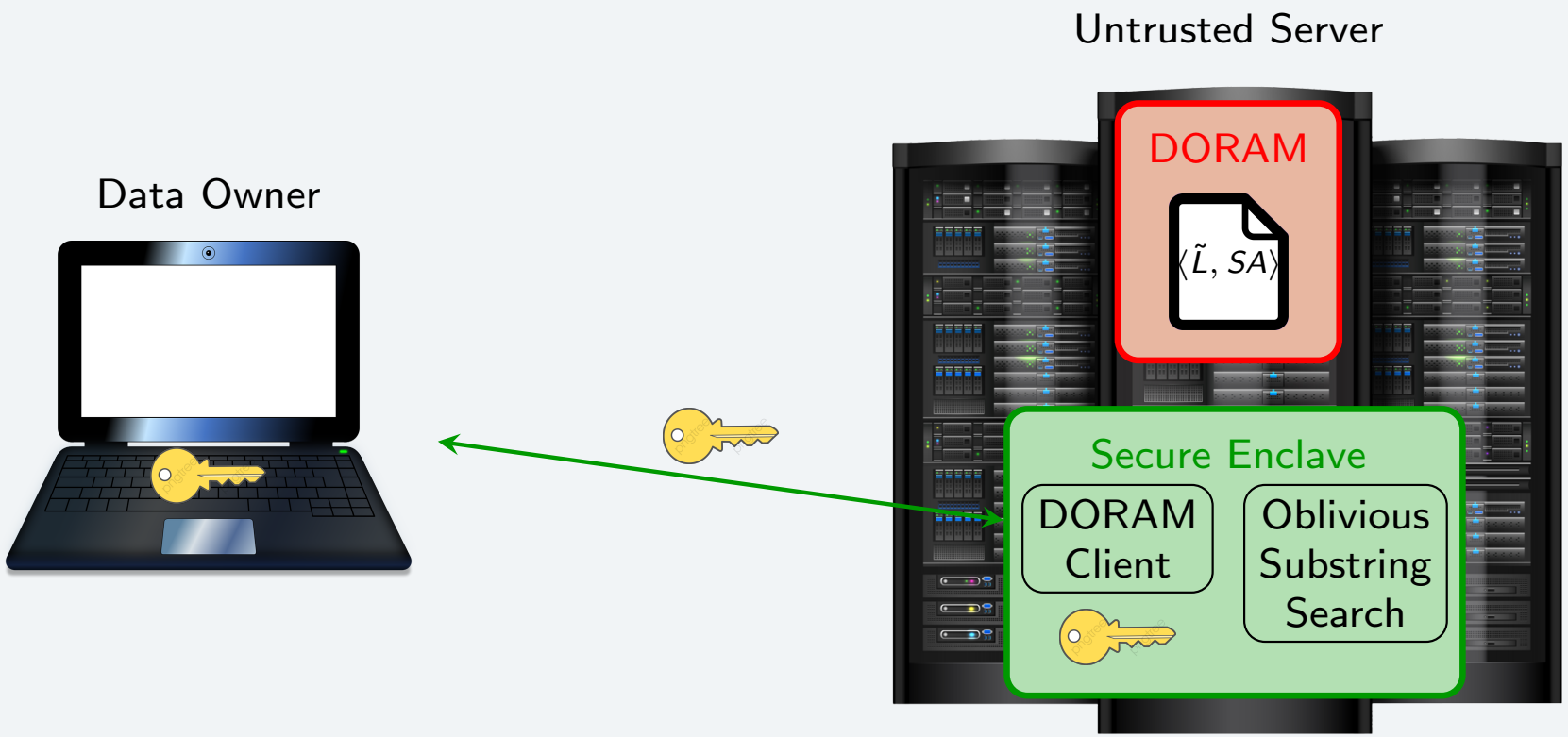
Load



Remote Attestation allows data owner to:

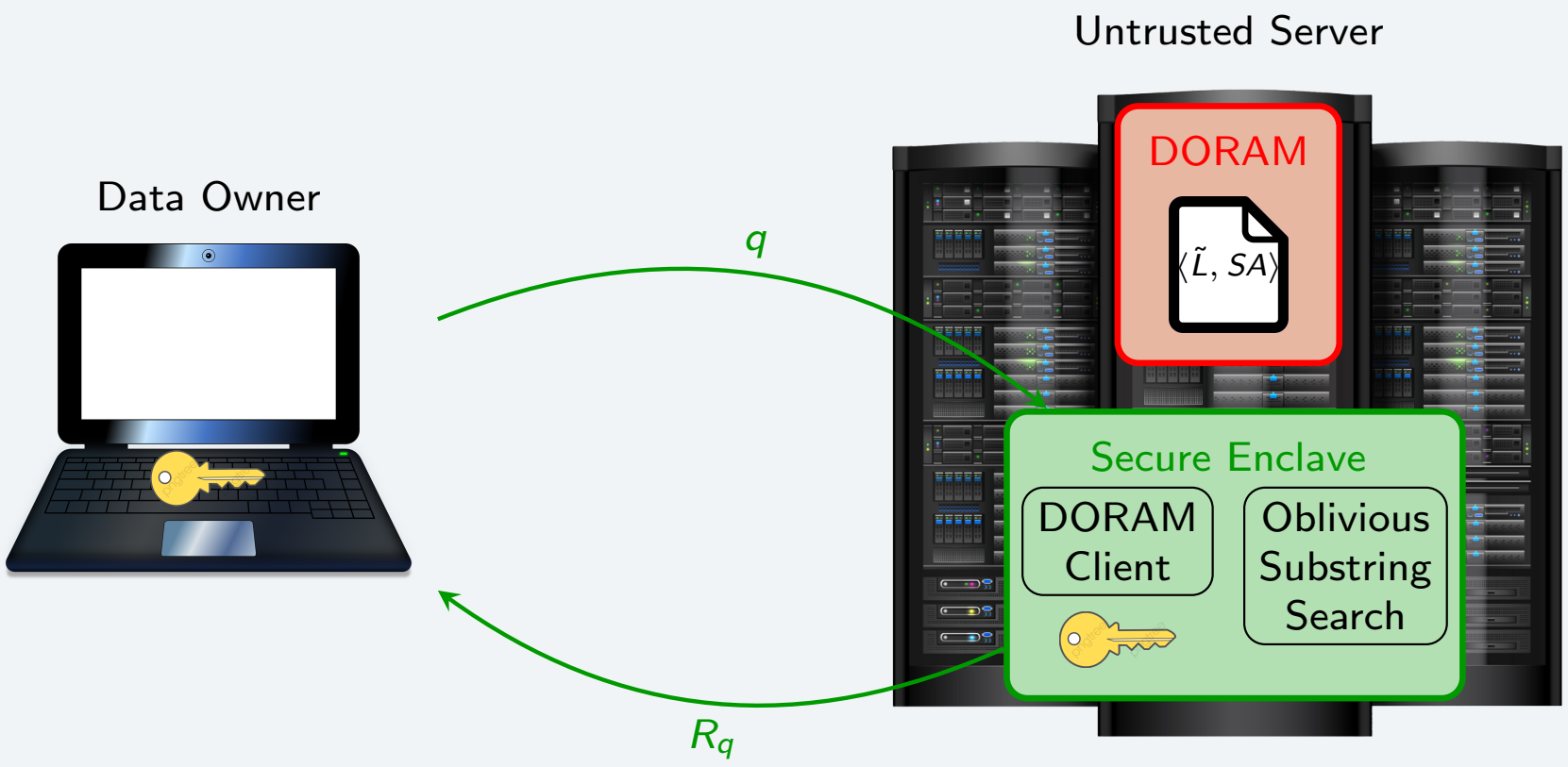
1. Verify the proper setup of the enclave
2. Establish a secure communication channel with the enclave

Load



Integrity of index is verified upon decryption

Query



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

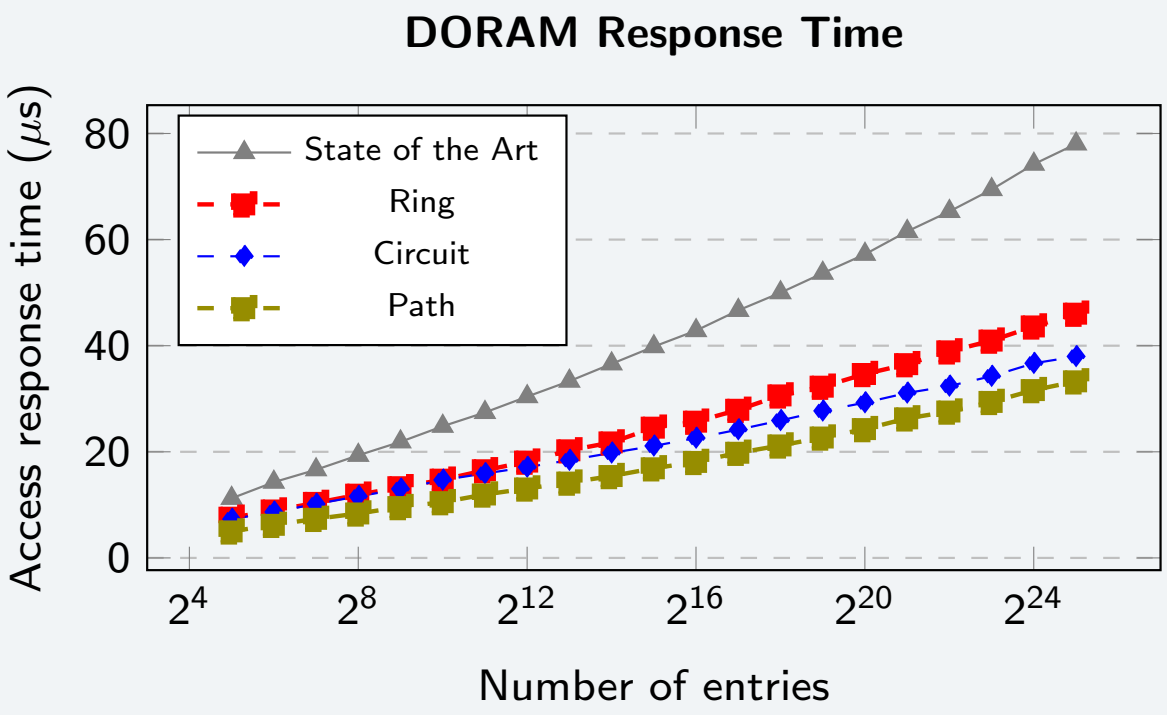
ObSQRE is able to detect any tampering with data and computation

- 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 Setup

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

DORAM Benchmarking

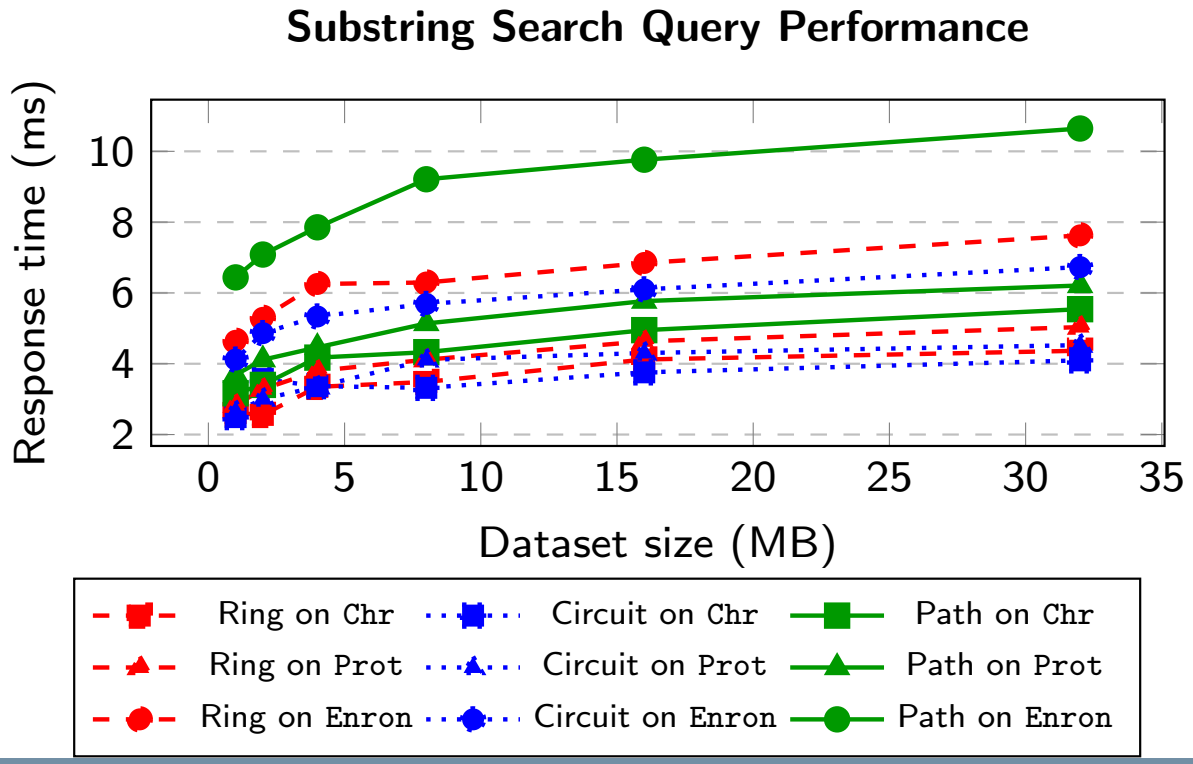


- All our DORAMs are faster than existing DORAM
- Path DORAM is fastest among our DORAMs

ObSQRE Experimental Evaluation

Dataset

- `Chr`: 21st human chromosome, alphabet size: 7
- `Prot`: set of human proteins, alphabet size: 25
- `Enron`: real-world email dataset, alphabet size: 91



- Poly-logarithmic trend
- ↗ alphabet size leads to ↘ performance
- Best DORAM: Circuit DORAM

ObSQRE in Real-World Scenarios

We evaluate our best solution on two real-world use cases:

- Look-up of a protein in the human genome (3*GiB* 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

Dataset	m	**#Occ.**	**ObSQRE (ms)**	**SGX+ ORAM (ms)**	**no SGX+ no ORAM (μs)**
Genome	**3050**	1	1019	347	167
Enron	**5**	2657	3.7	1.4	11.2
Enron	**13**	290	5.5	1.9	4.2
Enron	**20**	154	8.4	3	5.7

Practical performance, limited overhead ☺

Conclusions and Future Works

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

References

[FHV18] Sebastian Faust, Carmit Hazay, and Daniele Venturi.
Outsourced pattern matching.
Int. J. Inf. Sec., 17(3):327–346, 2018.

[IIY17] Yu Ishimaki, Hiroki Imabayashi, and Hayato Yamana.
Private Substring Search on Homomorphically Encrypted Data.
In *2017 IEEE International Conference on Smart Computing, SMARTCOMP 2017, Hong Kong, China, May 29-31, 2017*, pages 1–6. IEEE Computer Society, 2017.

[MB15] Tarik Moataz and Erik-Oliver Blass.
Oblivious Substring Search with Updates.
IACR Cryptology ePrint Archive, 2015:722, 2015.

[QZZX20] Shiyue Qin, Fucai Zhou, Zongye Zhang, and Zifeng Xu.
Privacy-preserving substring search on multi-source encrypted gene data.
IEEE Access, 8:50472–50484, 2020.