SEAL\textsuperscript{1}: Mitigating Attacks on Encrypted Databases via Adjustable Leakage

\textsuperscript{1}Searchable Encryption with Adjustable Leakage

Ioannis Demertzis
University of Maryland
yannis@umd.edu

Dimitrios Papadopoulos
HKUST
dipapado@cse.ust.hk

Charalampos Papamanthou
University of Maryland
cpap@umd.edu

Saurabh Shintre
NortonLifeLock Research Group
saurabh.shintre@nortonlifelock.com
What is Searchable Encryption (SE)?

Client

+ Untrusted Cloud

T_1
John Smith CMU $3,000

T_2
Alice Lu UCLA $4,000

...  

T_N
Bruce Don UMD $2,000

Untrusted Cloud
What is Searchable Encryption (SE)?

**Search pattern:** whether a search query is repeated

**Access pattern:** encrypted tuples that satisfy the search query

**Setup leakage:** total leakage prior to query execution, e.g. size of the encrypted database

**Leakage** is the amount of information that the untrusted cloud learns

**Untrusted Cloud**

- **T1**: John Smith, CMU, $3,000
- **T2**: Alice Lu, UCLA, $4,000
- **TN**: Bruce Don, UMD, $2,000

**search query:** Bruce

**Searchable Encryption**

- Client: John Smith, CMU, $3,000
- Alice Lu, UCLA, $4,000
- Bruce Don, UMD, $2,000
What is Searchable Encryption (SE)?

**Search pattern:** whether a search query is repeated

**Access pattern:** encrypted tuples that satisfy the search query

**Volume pattern:** result size

**Overlapping pattern:** the tuple overlaps between previous queries

**Setup leakage:** total leakage prior to query execution, e.g., size of the encrypted database

**Security (informal):** The adversary does not learn anything beyond the above leakages!
Assume that the adversary knows a fraction $N^\gamma$ ($\gamma \in [0,1]$) of the plaintext input.
Attacks on SE

Limitations of prior attacks:

i) Do not attack state-of-the-art schemes (e.g., range attacks)

ii) Assume that the attacker knows a great percentage of the input distribution

iii) Assume that the query distribution is known to the attacker

iv) Assume that the input database has a specific structure
Assume that the adversary knows a fraction $N^\gamma$ ($\gamma \in [0,1]$) of the plaintext input.
SEAL: Searchable Encryption with Adjustable Leakage

- Hides the search and overlapping patterns
- Hides the volume pattern

Our experimental results

Applies to point, range, group-by and join queries

<table>
<thead>
<tr>
<th>Security</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Low</td>
<td>Low</td>
</tr>
</tbody>
</table>

SE: Searchable Encryption

ORAM: On-Device Random Access Memory

ORAM + Worst Case Padding
Contribution

- **SEAL**: Searchable Encryption with Adjustable Leakages
  - ADJable-ORAM-α (hides search and overlapping leakages)
  - ADJable-Padding-x (hides volume leakage)
- Attacks for point, range, join and group-by queries
  - First attack sketch for state-of-the-art range schemes
- New constructions for point, range, join, group-by queries
  - Using SEAL as black-box
- New customized Range Scheme, robust against attacks
- Experimental adjustment of search/overlapping/volume leakages
Focus of this talk

- **SEAL: Searchable Encryption with Adjustable Leakages**
  - ADJable-ORAM-\( \alpha \) (hides search and overlapping leakages)
  - ADJable-Padding-x (hides volume leakage)

- Attacks for point, range, join and group-by queries
  - First attack sketch for state-of-the-art range schemes

- New constructions for point, range, join, group-by queries
  - Using SEAL as black-box

- New customized Range Scheme, robust against attacks

- Experimental adjustment of search/overlapping/volume leakages
Adjustable-ORAM-\(\alpha\) (ADJ-ORAM-\(\alpha\))

Retrieve the \(i\)-th tuple without revealing the value \(i\) to the server

\[
\begin{array}{cccccccc}
1 & 2 & 3 & 4 & 5 & \ldots & i & i+1 & \ldots & n-1 & n \\
v_1 & v_2 & v_3 & v_4 & v_5 & \ldots & v_i & v_{i+1} & \ldots & v_{n-1} & v_n
\end{array}
\]

**ADJ-ORAM-\(\alpha\):** Leak \(\alpha\) bits of the accessed memory locations!
ADJ-ORAM-α

\[(i_1, v_1), (i_2, v_2), (i_3, v_3), \ldots, (i_n, v_n)\]

Cost per access = \(T(n)\)

ADJ-ORAM-0

\[\text{Cost per access} = T\left(\frac{n}{2}\right)\]

ADJ-ORAM-1

\[\text{Cost per access} = T(n)\]
\[
\text{ADJ-ORAM-}\alpha (i_1, v_1, i_2, v_2, i_3, v_3, \ldots, i_n, v_n) \n\]

\[
\begin{align*}
\text{Cost per access} & = T(n) \\
\text{MSB}^\alpha(\pi_k(i)) & \rightarrow \text{ORAM-0..0, ORAM-0..1, \ldots, ORAM-1..1} \\
\text{Cost per access} & = T\left(\frac{n}{2^\alpha}\right)
\end{align*}
\]
Adjustable-Padding-x

- **Observation 1:** In a dataset of size $N$ a query result can have up to $N$ different sizes

- **Observation 2:** We can perform worst-case padding to eliminate the volume pattern leakage (1 unique size)

- **Adjustable Padding:** Pad all the query results to the closest power of $x$.
  - The server can observe up to $\log_x N + 1$ different sizes
  - Volume Pattern leakage $\log \log_x N + 1$ bits

- At the end, we pad the dataset to have $x*N$ entries to avoid leaking extra information
SEAL($\alpha, x$)

- Uses ADJ-ORAM-$\alpha$, ADJ-Padding-$x$ and an oblivious dictionary as black-boxes.

- Parameter $\alpha$ is defined in the range $[0, \log N]$
  - $\alpha=0$ all the search/overlapping pattern bits are protected
  - $\alpha=\log N$ all the search/overlapping pattern bits are leaked

- For larger $x$ values less volume pattern bits are leaked
  - $x=N$ no volume pattern bits are leaked

- SEAL($\alpha, x$) can be used as a building block for point/range/join/group-by queries providing a security/efficiency trade-off.
Outline

- SEAL: Searchable Encryption with Adjustable Leakages
  - ADJable-ORAM-α (hides search and overlapping pattern leakages)
  - ADJable-Padding-x (hides volume pattern leakage)
- Attacks for point, range, join and group-by queries
  - First attack-sketch for the state-of-the-art range schemes
- New constructions for point, range, join, group-by queries
  - Using SEAL as black-box
- New more efficient customized Range Scheme robust against attacks
- Experimentally adjusting these leakages
Threat Model and Attacks

**Attacker’s Goals:**
(i) Decrypt the client’s encrypted queries (Query Recovery attack)
(ii) Decrypt the encrypted database (Database Recovery attack)

**Attacker’s Power:**
- Has access to the server observing all the possible encrypted queries
- Has plaintext access to the input dataset

**Query Recovery-Success Rate** ($QR_{SR}$) = \( \frac{\text{Correctly Decrypted Queries}}{|Q|} \)

**Database Recovery-Success Rate** ($DR_{SR}$) = \( \frac{\text{Correctly Decrypted Tuples}}{N} \)
Attack Configuration

- Modified Frequency Analysis Attack proposed by Naveed et al. [CCS2016]

- 1 real dataset with 6,123,276 records of reported crime incidents
  - 22 attributes with different distributions:
    - ID, Case Number, Date, Block, ICR, Primary Type, Description, Location Description, Arrest, Domestic, Beat, District, Ward,
    - Community Area, FBI Code, X Coordinate, Y Coordinate, Year,
    - Updated On, Latitude, Longitude, Location.

- TPC-H Benchmark
  - 8 tables (61 different attributes)
    - PART, SUPPLIER, PARTSUPP, CUSTOMER, NATION, LINEITEM, REGION, ORDERS
Parameter $\alpha$ controls the search/overlapping pattern leakage ($\alpha=[0...\log N]$)
Parameter $x$ controls the volume pattern leakage ($x=[$No padding, 2, ... N$]$)

(a) $x = \bot$

(b) $x = 2$
Database Recovery Attack for Join Queries

Parameter $\alpha$ controls the search/overlapping pattern leakage ($\alpha=[0 \ldots \log N]$)

Parameter $x$ controls the volume pattern leakage ($x=\text{[No padding, 2, \ldots N]}$)

(a) SUPPLIER ∩ NATION  
(b) CUSTOMER ∩ NATION
Adjusting Parameters “α” and “x” in Practice

Finding appropriate parameter values is *data-dependent*:

- Size of the database
- Number of distinct values
- Distribution of the searchable attribute

**Approach:** Before outsourcing the database, for a given attribute, use existing/our all-powerful attacks and try different values of “a” and “x”

**General Guidelines:**

- Point/Join/Group-by queries: $\alpha = \log N - 3$ and $x=4$ ($\sim 32x$ overhead)
- Range Queries: $x=8$ ($\sim 12x$ overhead)
Thank you!! Questions?

- Efficiency
- Security

Hides the search and overlapping patterns
Hides the volume pattern
Worst Case Padding

SE
ORAM
ORAM + SEAL