Order-Revealing Encryption:
How to Search on Encrypted Data

David Wu
Stanford University

based on joint works with Nathan Chenette, Kevin Lewi, and Stephen A. Weis
The information accessed from potentially exposed accounts “may have included names, email addresses, telephone numbers, dates of birth, hashed passwords (using MD5) and, in some cases, encrypted or unencrypted security questions and answers...”
Searching on Encrypted Data

The database was discovered by MacKeeper researcher Chris Vickery on March 31, in the course of searching for random phrases on the domain s3.amazonaws.com.

“It's as bad as I expected,” he tweeted. “Bank-related. Plaintext passwords. Big name company. I've reached out to them.”
A Republican contractor’s database of nearly every voter was left exposed on the Internet for 12 days, researcher says

By Brian Fung, Craig Timberg and Matea Gold  June 19
Searching on Encrypted Data

Data Breach at Anthem May Forecast a Trend

By REED ABELSON and JULIE CRESWELL    FEB. 8, 2015
Searching on Encrypted Data

Technology

Millions of hacked LinkedIn IDs advertised 'for sale'

18 May 2016 | Technology
data breaches have become the norm rather than the exception...
Why Not Encrypt?

data breaches have become the norm rather than the exception...
Why Not Encrypt?

“because it would have hurt Yahoo’s ability to index and search messages to provide new user services”

~Jeff Bonforte (Yahoo SVP)
Searching on Encrypted Data

client holds a secret key (needed to encrypt + query the server)

server stores encrypted database

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Alice</td>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>47</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Charlie</td>
<td>41</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Inigo</td>
<td>45</td>
<td>4</td>
</tr>
</tbody>
</table>
Security for Encrypted Search

- **active adversary**
  - adversary sees encrypted database + queries and can interact with the database
  - online attacks (e.g., active corruption)
  - offline attacks (e.g., passive snapshots)

- **snapshot adversary**
  - adversary only sees contents of encrypted database
Security for Encrypted Search

adversary sees encrypted database + queries and can interact with the database

online attacks (e.g., active corruption)
offline attacks (e.g., passive snapshots)

typical database breach: contents of database are stolen and dumped onto the web
Order-Revealing Encryption [BLRSZZ’15]

secret-key encryption scheme

\[ \text{ct}_1 = \text{Enc} (\text{sk}, 123) \]
\[ \text{ct}_2 = \text{Enc} (\text{sk}, 512) \]
\[ \text{ct}_3 = \text{Enc} (\text{sk}, 273) \]

Which is greater: the value encrypted by \( \text{ct}_1 \) or the value encrypted by \( \text{ct}_2 \)?

(client)

(server)

(legacy-friendly) range queries on encrypted data
Order-Revealing Encryption [BLRSZZ’15]

given any two ciphertexts

\[
ct_1 = \text{Enc}(sk, x) \quad \text{ct}_2 = \text{Enc}(sk, y)
\]

\[x > y\]

there is a public function for performing comparisons
Order-Revealing Encryption [BLRSZZ’15]

given any two ciphertexts

c\textsubscript{1} = \text{Enc}(\text{sk}, x) \quad \text{ct}_1 \quad \text{ct}_2 = \text{Enc}(\text{sk}, y)

\begin{align*}
\text{ct}_1 &= \text{Enc}(\text{sk}, x) \\
\text{ct}_2 &= \text{Enc}(\text{sk}, y)
\end{align*}

best-possible security: reveal just the ordering and nothing more

\begin{align*}
x &> y
\end{align*}

in practice: constructions reveal some additional information
Existing Approaches

Something in between?

Space

Efficiency

Security

OPE [BCLO’09]

constructions based on mmaps [BLRSZZ’15] or obfuscation [GGGJKLSSZ’14]

Not drawn to scale
A Simple ORE Construction [CLWW’16]

For each index $i$, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \mod 3$.

$F : \mathcal{K} \times \{0,1\}^* \rightarrow \{0,1,2\}$
A Simple ORE Construction [CLWW’16]

For each index $i$, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod{3}$
For each index $i$, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod{3}$.
A Simple ORE Construction [CLWW’16]

For each index $i$, apply a PRF (e.g., AES) to the first $i - 1$ bits, then add $b_i \pmod{3}$.

$$F_k(\epsilon) + 1$$

$$F_k(1) + 0$$

$$F_k(10) + 0$$
A Simple ORE Construction [CLWW’16]

\[ F_k(\epsilon) + 1 \]

\[ F_k(1) + 0 \]

\[ F_k(10) + 0 \]

\[ F_k(100) + 1 \]

\[ F_k(1001) + 0 \]

\[ F_k(10010) + 1 \]

same prefix = same ciphertext block

first block that differs

different prefix = value computationally hidden
Efficiency

Each ciphertext block is element in \{0,1,2\}

For $n$-bit messages, can obtain ciphertexts of length $\approx 1.6n$

Encryption only requires PRF evaluations while decryption just requires bitwise comparisons
Security follows directly from security of the PRF.

Construction reveals the first bit on which two message differ (in addition to the ordering).
Inference Attacks [NKW’15, DDC’16, GSBNR’17]

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>wpjOos</td>
<td>2wzXW8</td>
<td>SqX9I9</td>
<td>KqLUXE</td>
</tr>
<tr>
<td>XdXdg8</td>
<td>y9GFpS</td>
<td>gwilE3</td>
<td>MJ23b7</td>
</tr>
<tr>
<td>P6vKhW</td>
<td>EgN0Jn</td>
<td>S0pRJe</td>
<td>aTaeJk</td>
</tr>
<tr>
<td>orJRe6</td>
<td>KQWy9U</td>
<td>tPWF3M</td>
<td>4FBE00</td>
</tr>
</tbody>
</table>

encrypted database

public information

frequency and statistical analysis

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>???</td>
<td>Alice</td>
<td>30-35</td>
<td>2</td>
</tr>
<tr>
<td>???</td>
<td>Bob</td>
<td>45-50</td>
<td>3</td>
</tr>
<tr>
<td>???</td>
<td>Charlie</td>
<td>40-45</td>
<td>2</td>
</tr>
<tr>
<td>???</td>
<td>???</td>
<td>40-45</td>
<td>4</td>
</tr>
</tbody>
</table>

plaintext recovery
Inference Attacks [NKW’15, DDC’16, GSBNR’17]

ORE schemes always reveal order of ciphertexts and thus, are vulnerable to offline inference attacks.

Can we fully defend against offline inference attacks while remaining legacy-friendly?
ORE with Additional Structure

Desired functionality: range queries on encrypted data

Key primitive: order-revealing encryption scheme where ciphertexts have a “decomposable” structure

ciphertexts naturally split into two components
ORE with Additional Structure

Enc_L(101) \rightarrow ct_L
Enc_R(100) \rightarrow ct_R

Comparison can be performed between left ciphertext and right ciphertext.

Right ciphertexts provide semantic security!

Robustness against offline inference attacks!
Encrypted Range Queries

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Alice</td>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>47</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Charlie</td>
<td>41</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Inigo</td>
<td>45</td>
<td>4</td>
</tr>
</tbody>
</table>

store right ciphertexts in sorted order

build encrypted index

record IDs encrypted under independent key

record IDs

separate index for each searchable column, and using independent ORE keys
Encrypted Range Queries

Encrypted database:

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Alice</td>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>47</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Charlie</td>
<td>41</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Inigo</td>
<td>45</td>
<td>4</td>
</tr>
</tbody>
</table>

columns (other than ID) are encrypted using a semantically-secure encryption scheme

clients hold (secret) keys needed to decrypt and query database

encrypted search indices
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}_L(40) \text{ Enc}_L(45)$
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

<table>
<thead>
<tr>
<th>Age</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enc_R(31)</td>
<td>Enc(0)</td>
</tr>
<tr>
<td>Enc_R(41)</td>
<td>Enc(2)</td>
</tr>
<tr>
<td>Enc_R(45)</td>
<td>Enc(3)</td>
</tr>
<tr>
<td>Enc_R(47)</td>
<td>Enc(1)</td>
</tr>
</tbody>
</table>
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

- Use binary search to determine endpoints (comparison via ORE)
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

- $\text{Enc}_L(40)$
- $\text{Enc}_L(45)$

use binary search to determine endpoints (comparison via ORE)

<table>
<thead>
<tr>
<th>Age</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Enc}_R(31)$</td>
<td>Enc(0)</td>
</tr>
<tr>
<td>$\text{Enc}_R(41)$</td>
<td>Enc(2)</td>
</tr>
<tr>
<td>$\text{Enc}_R(45)$</td>
<td>Enc(3)</td>
</tr>
<tr>
<td>$\text{Enc}_R(47)$</td>
<td>Enc(1)</td>
</tr>
</tbody>
</table>
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

- Use binary search to determine endpoints (comparison via ORE).
- Return encrypted indices that match query.

<table>
<thead>
<tr>
<th>Age</th>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>EncR(31)</td>
<td>Enc(0)</td>
</tr>
<tr>
<td>EncR(41)</td>
<td>Enc(2)</td>
</tr>
<tr>
<td>EncR(45)</td>
<td>Enc(3)</td>
</tr>
<tr>
<td>EncR(47)</td>
<td>Enc(1)</td>
</tr>
</tbody>
</table>
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

Encrypted range query is performed by the server. The client decrypts the indices to obtain the set of matching records.
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}(2) \quad \text{Enc}(3) \\
\text{Records 2, 3} \\
\text{Enc}(r_2) \quad \text{Enc}(r_3)$
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

$\text{Enc}(2) \quad \text{Enc}(3)\quad \text{Records 2, 3} \quad \text{Enc}(r_2) \quad \text{Enc}(r_3)\quad \text{client decrypts to obtain records}$
Encrypted Range Queries

Query for all records where $40 \geq \text{age} \geq 45$:

Some online leakage: access pattern + ORE leakage.
Encrypted Range Queries

Encrypted database (view of the snapshot adversary):

<table>
<thead>
<tr>
<th>ID</th>
<th>Name</th>
<th>Age</th>
<th>Diagnosis</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Alice</td>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Bob</td>
<td>47</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Charlie</td>
<td>41</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Inigo</td>
<td>45</td>
<td>4</td>
</tr>
</tbody>
</table>

encrypted database is semantically secure!

Perfect offline security

encrypted search indices
A New ORE Scheme [LW’16]

“small-domain” ORE with best-possible security

domain extension technique from [CLWW’16]

ORE with some leakage

first practical ORE construction that can provide best-possible offline security!
### Performance Evaluation

| Scheme                     | Encrypt (μs) | Compare (μs) | |ct| (bytes) |
|----------------------------|--------------|--------------|-----------------|
| OPE [BCLO’09]              | 3601.82      | 0.36         | 8               |
| [CLW’16] ORE               | 2.06         | 0.48         | 8               |
| [LW’16] ORE (8-bit blocks) | 54.87        | 0.63         | 224             |

Benchmarks taken for C implementation of different schemes (with AES-NI). Measurements for encrypting 32-bit integers.
Performance Evaluation

| Scheme                        | Encrypt (μs) | Compare (μs) | |ct| (bytes) |
|-------------------------------|--------------|--------------|---------------|
| OPE [BCLO’09]                | 3601.82      | 0.36         | 8             |
| [CLWW’16] ORE                | 2.06         | 0.48         | 8             |
| [LW’16] ORE (8-bit blocks)   | 54.87        | 0.63         | 224           |

Encrypting byte-size blocks is 65x faster than OPE, but ciphertexts are 30x longer. Security is substantially better.
The Landscape of ORE

- Constructions based on mmaps [BLRSZZ’15] or obfuscation [GGGJLKSSZ’14]
- Broken by inference attacks [NKW’15, DDC’16, GSBNR’16]
- Can provide perfect offline security

- Concurrent work [CLOZ’16, JP’16]
- Not drawn to scale
Conclusions

- Inference attacks render direct usage of ORE insecure
- However, ORE is still a useful building block for encrypted databases

- Introduced new paradigm for constructing ORE that enables range queries in a way that is mostly legacy-compatible and provides offline semantic security
- New ORE construction that is concretely efficient with strong security
Questions?

Website: https://crypto.stanford.edu/ore/
Code: https://github.com/kevinlewi/fastore