Secure Anonymous Database Search

Mariana Raykova
Binh Vo
Steven Bellovin
Tal Malkin
The underlying problem

- **Goal:** Controlled data sharing
- When protecting content, how do parties know if they have *data worth sharing*?
- Anonymous search
Further system requirements

- Search efficiency - sublinear
- Multiple parties
  - authentication – limit parties that can search
  - anonymization - hide querier identity
Our solution

- System architecture
- Building blocks
- Analysis
- Implementation
- Test results
Search

- What is efficient search? – sublinearity
  - decryption capability for matching ciphertext does not work

- How to achieve?
  - deterministic encryption [BBO07] – high min entropy of plaintext domain, replace randomness with hash
  - Bloom filters

- Trade-offs
  - relaxed security notions – equality pattern leaked
  - false positives – can be bounded
System architecture

- Index Server – encrypted search
- Query Router – authentication and user anonymity
Re-routable encryption

- Goal
  - A has some information
  - A trusts B to distribute, but not to see
  - How to control distribution?
- Ciphertext transformation under different keys
  - Encryption scheme with group property
PH-DSAEP+

- Private key deterministic encryption – following BBO07
- Pohlig-Hellman function
  - Group property:
    \[ \text{PH}_{k_1}( \text{PH}_{k_2}(x)) = \text{PH}_{k_1*k_2}(x) \]
- Message padding SAEP+ [Boneh01]
  - Randomness r replaced by a hash
Bloom Filter Efficient Search

- **Bloom filters** – extend the idea of hashing
BFs for Document Search

- BF per document with stemmed words entries
Secure Anonymous Database Search (SADS)

```
BF_Search(i1, ..., ik) = {r1, ..., rn} = res_v

BF_indices(c') = {i1, ..., ik}

c' = PH-DSAEP+(c, transform key for S)

c = PH-DSAEP+(query, Client's key)

res' = PH-SAEP+(res_v, IS key)

res" = PH-SAEP+(res', transform key for Client)
```
Trust Assumptions – IS, QR

- Trust distribution – semi-honest IS, QR:
  - QR - correct key transformation
  - IS - correct BF search

- Privacy with respect to IS
  - IS does not know relation of BF to documents
  - Client anonymity - cannot link queries of one client

- Privacy with respect to QR
  - Query privacy – up to equality, PH-DSAEP+
  - Result privacy
Security Guarantees

- **Server** participates only in preprocessing
- **Client**
  - Authenticated by QR
  - Learns only relevant result – *adjustable false positive rate, no false negatives*
- **Collusion of IS and QR:**
  - *Search pattern* in results leaked
  - *No search capability* - cannot submit queries
Index implementation

- What is bitslicing?
  - View a set of BFs as a matrix
  - Transpose
  - Track 'zeroed' slices

- What is gained?
  - Don’t read unnecessary
  - Cache behavior
Better Boolean queries

- The naïve way to do and/or queries
  - Run term queries in parallel
  - Union/intersect
- How we can do it better in sliced indexes
  - AND queries unioned in query indices
  - OR queries processed in parallel
  - OR query indices are handled in order of frequency in queries
Performance

- *Constant search* time per BF
- *Parallel search* over multiple BFs (minimal overhead)
- What is considered “acceptable”, compare with network delay

<table>
<thead>
<tr>
<th></th>
<th>Local server</th>
<th>trans US</th>
<th>Europe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ping time (ms)</td>
<td>0.227</td>
<td>90.615</td>
<td>110.978</td>
</tr>
</tbody>
</table>
Corpus size

Average Query Search Time for Different Database Sizes
OR improvement

Ratio of Search Times for One N-Term Query and N Single Queries

Number of terms in OR query

N-Term query/N single queries
Conclusion

- New search problem
- Efficient solution
- Introduction of a new encryption method
- Re-routable encryption primitive
Thank You!

- Questions?