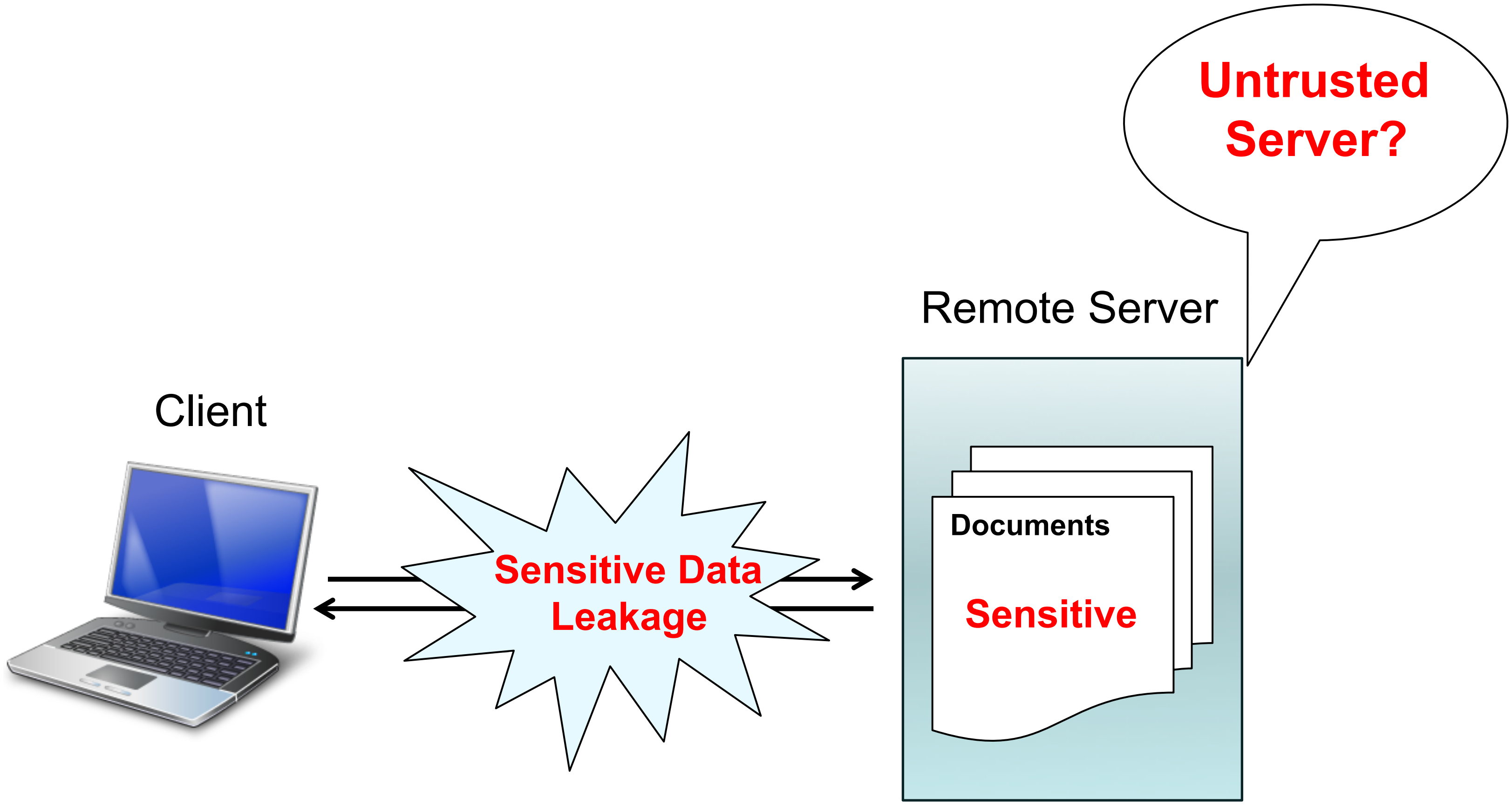
Access Pattern disclosure on Searchable Encryption: Ramification, Attack and Mitigation

Murat Kantarcioglu
Joint work with Mohammad Saiful Islam,
Mehmet Kuzu,

Introduction



Introduction

Solution: Data Encryption

Untrusted Server?

Remote Server

Client

Search over Encrypted Data

Selective Encrypted Items

Encrypted

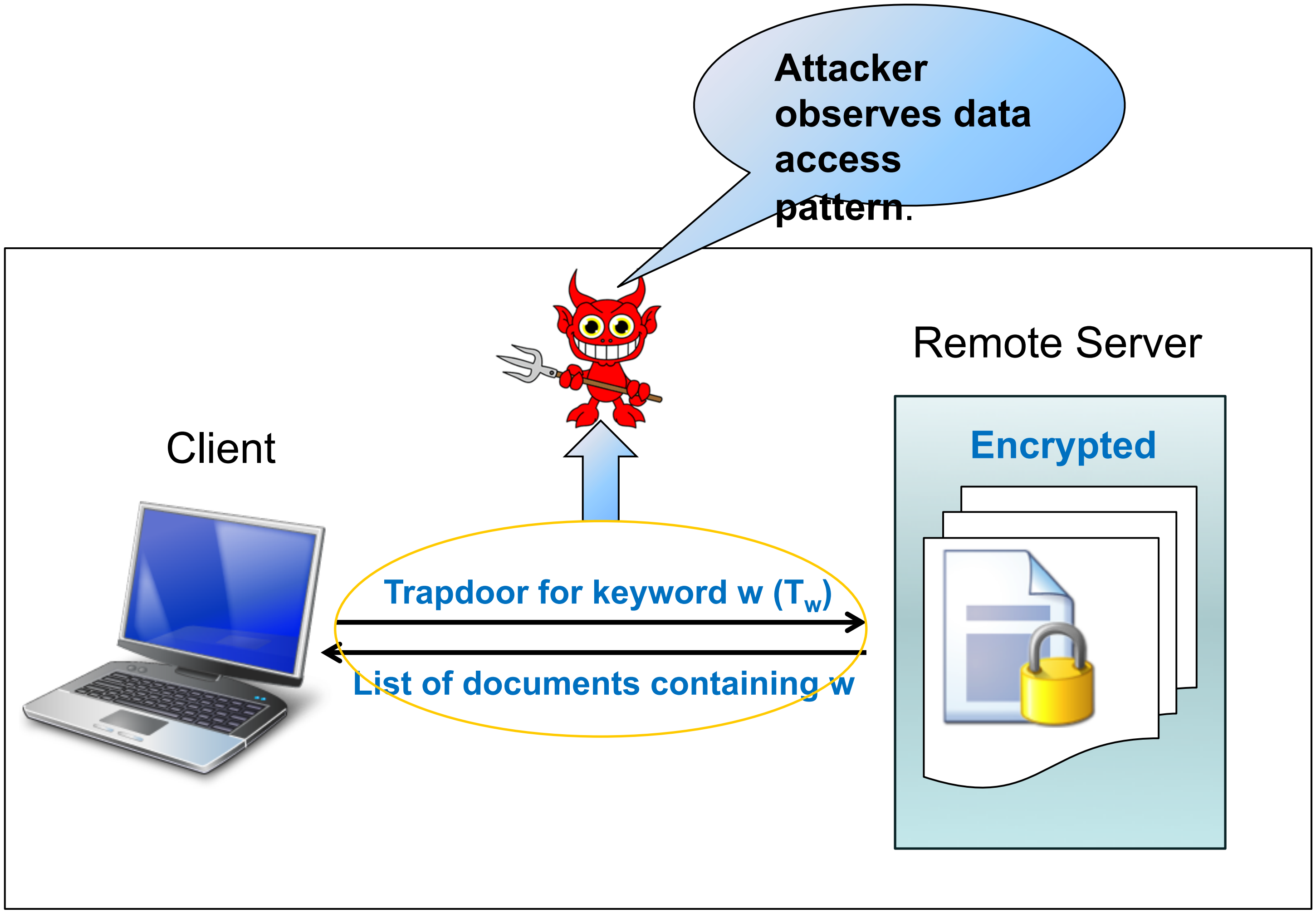
Requires: Efficient but Secure Searchable Encryption Protocols

Existing Protocols

- Oblivious RAM Type Protocols (ORAM)
    - E.g., Goldreich et. al., Williams et. al.
  - Secure: reveals no information to an adversary.
  - Too expensive for large data sets.

- Efficient Searchable Encryption Protocols
    - E.g., Song et. al., Goh et. al., Curtmola et. al.
  - Efficient: practically usable.
  - **Reveals Access Patterns.**

Access Pattern Disclosure

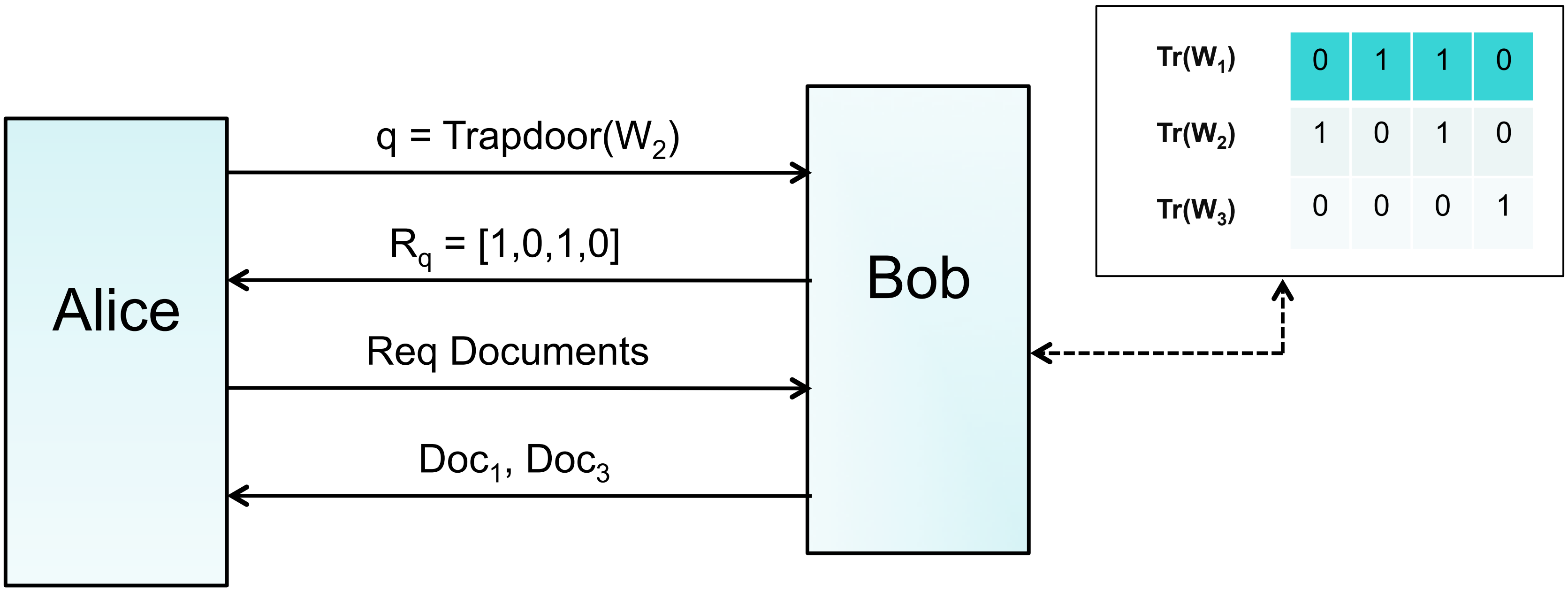


A Searchable Encryption Protocol that reveals Access Pattern

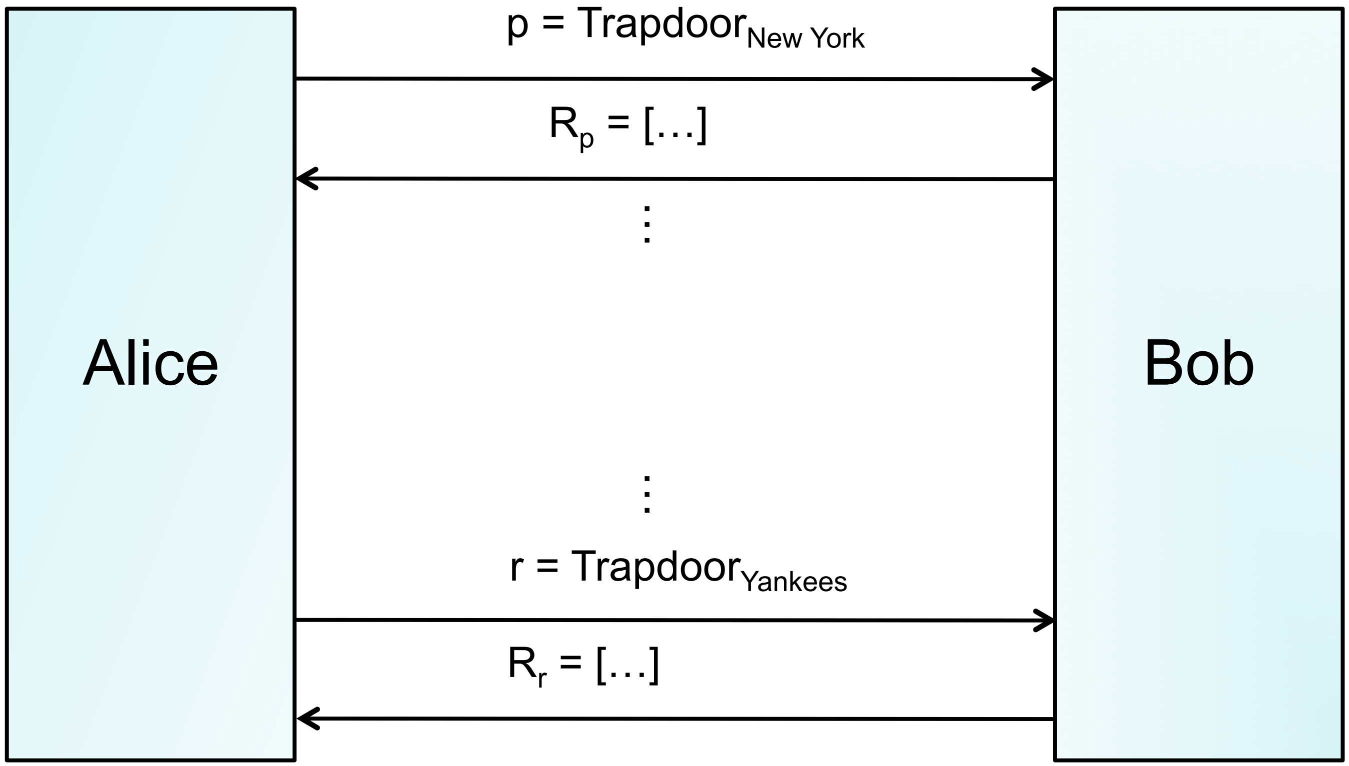
Contributions

- Investigate the ramification of Access Pattern Disclosure.
- Formalize a query identity inference attack model based on access pattern disclosure.
- Empirically verify the efficacy of such a model.
- Propose a noise addition technique to mitigate such an attack.

Simplified Searchable Encryption



Motivation



- Mallory can calculate the probability of {'New York', 'Yankees'} to appear in a document.
- What if the document corpus is about Major league baseball?

Notations

Notation	Meaning
D_i	The i^{th} Document.
$\mathcal{K}_i$	The i^{th} Keyword.
n	Number of documents.
m	Number of keywords.
Q	Set of l queries <Q_1, …, Q_l>.
R_q	Result sent by the server for query q.
K_Q	The set of known queries.
S	Set of keywords for which queries are known.
$Trapdoor_w$	Output of the trapdoor function for w.

Threat Model

- Attacker Mallory has access to the communication channel. Therefore, she observes $Q = \langle Q_1, \ldots . Q_l \rangle$ and their responses $\langle R_{Q1}, \ldots . R_{Ql} \rangle$.

- Mallory knows the underlying keywords for a set of k queries: K_Q .

- Mallory has access to a *(m x m) matrix M* s.t. $M_{i,j} = Pr[(\mathcal{K}_i \in d) \wedge (\mathcal{K}_j \in d)]$, here *d* is sampled uniformly from *D*.

Proposed Model

Objective: Given a set of queries Q, a set of known queries K_Q, a background matrix *M*, and the set of known keywords *S;* ascertain the sequence of indices $<a_1, \ldots, a_l>$ s.t. the following holds.

$$\arg\min_{\langle a_1,\cdots,a_l\rangle} \sum_{Q_i, Q_j \in Q} \left(\frac{R_{Q_i} \cdot R_{Q_j}^T}{n} - \left(K_{a_i} \cdot M \cdot K_{a_j}^T\right) \right)^2$$

Constraints :

$$\forall j \; s.t. \; Q_j \in S, a_j = x_j \; s.t. \langle \kappa_{x_j}, Q_j \rangle \in K_Q$$

$$\forall j, \| Q_j \| = 1$$

NP Completeness Theorem: Theorem 1

Finding an optimal assignment of keywords to a given set of queries w.r.t. the objective function defined in the simplified model is NP-Complete.

Experimental Setup

- Datasets Used: 30109 emails contained in the Enron Dataset _sent_mail folder.
  - Discarded the first few lines of metadata.
- Stemming Algorithm: Used Porter Stemming Algorithm to find the root of each keyword.
- Simulated Annealing: Used Simulated Annealing to solve the approximation of the simplified model.

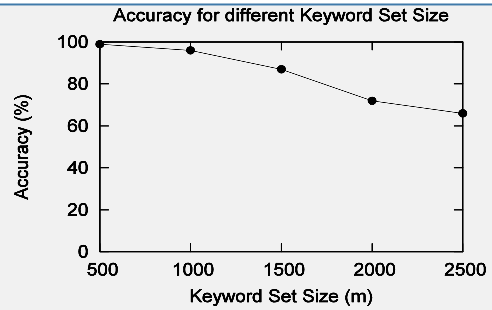
Experimental Setup Contd.

- Keyword Generation: We use the most frequent x keywords as our keyword set.
  - Discarded the most common words like *a, an ,the etc.*

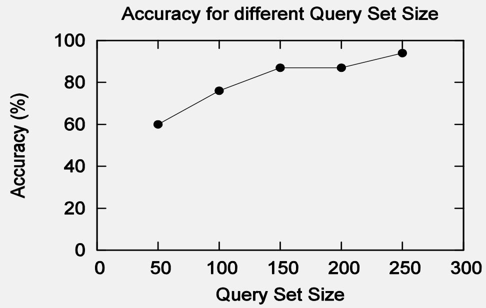
- Query Generation: We use Zipfian distribution to generate Query Set.

- Execution Time: All the experiments ran under 14 hours in a AMD Phenom II X6 1045T 2.70 GHz Windows 7 with 8 GB RAM .

Experiment Results

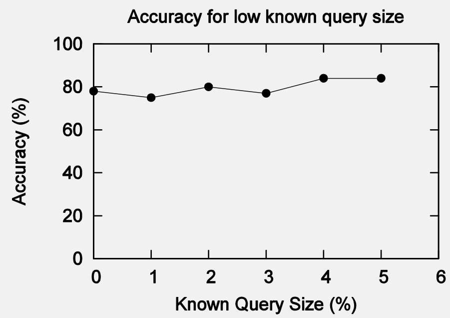


Parameters
- Query Set Size: 150
- Known Query Set Size: 15% of Query Set Size.
- # Documents: 30109



Parameters
- Keyword Set Size: 1500
- Known Query Set Size: 15%
- # Documents: 30109

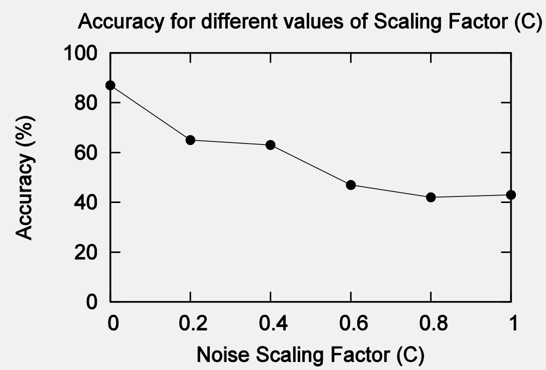
Experiment Results Contd.



Parameters

- Keyword Set Size: 1500
- Query Set Size: 150
- # Documents: 30109

Experiment Results Contd.



Parameters

- Keyword Set Size: 1500
- Query Set Size: 150
- # Documents: 30109

Noise Addition:

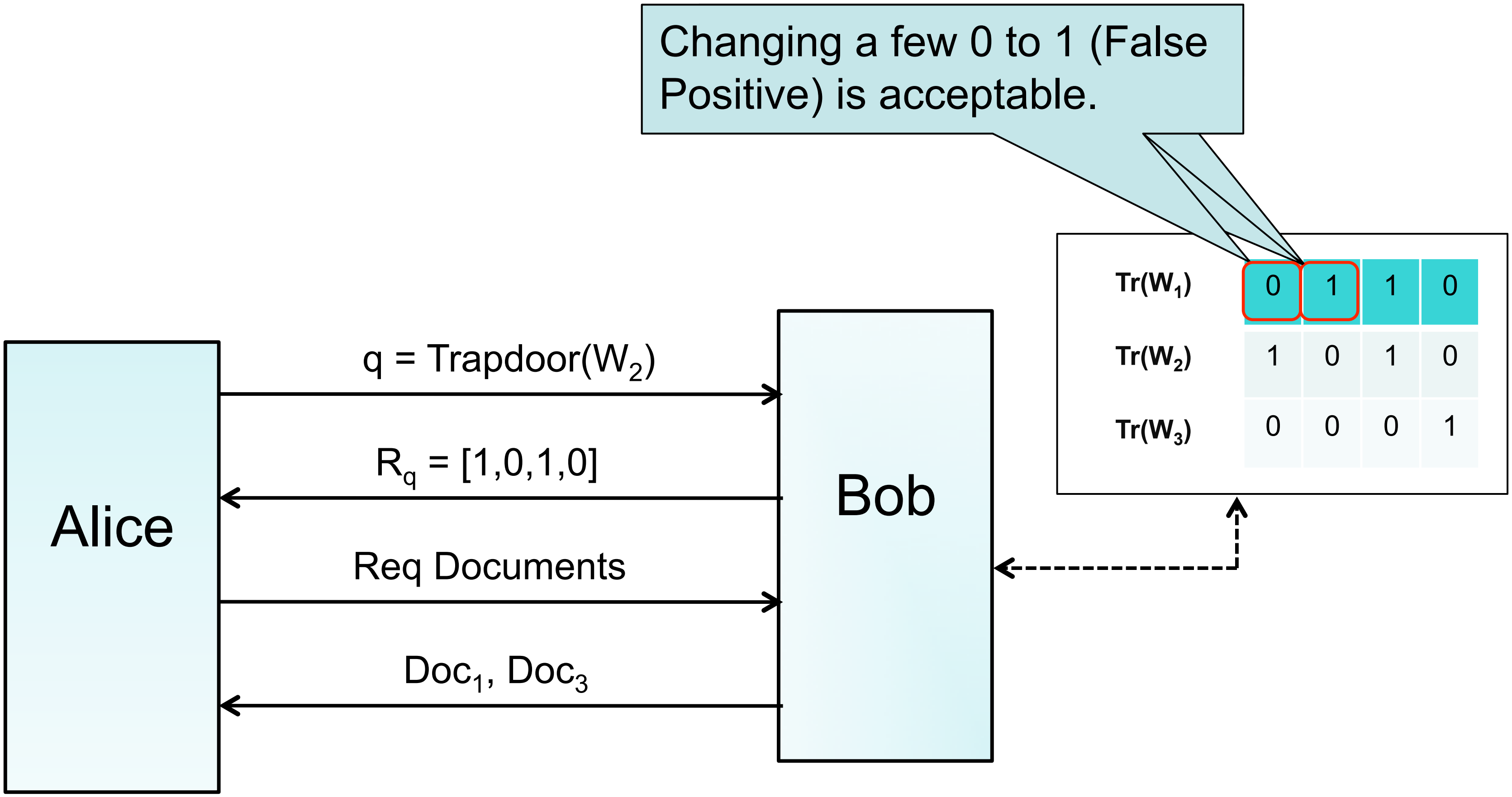
- $\sigma^2 = Var\{M_{i,j}\}$
- Add Noise: $\mathcal{N}(0, C\sigma^2)$
- C is noise scaling factor

Mitigating Inference Attack

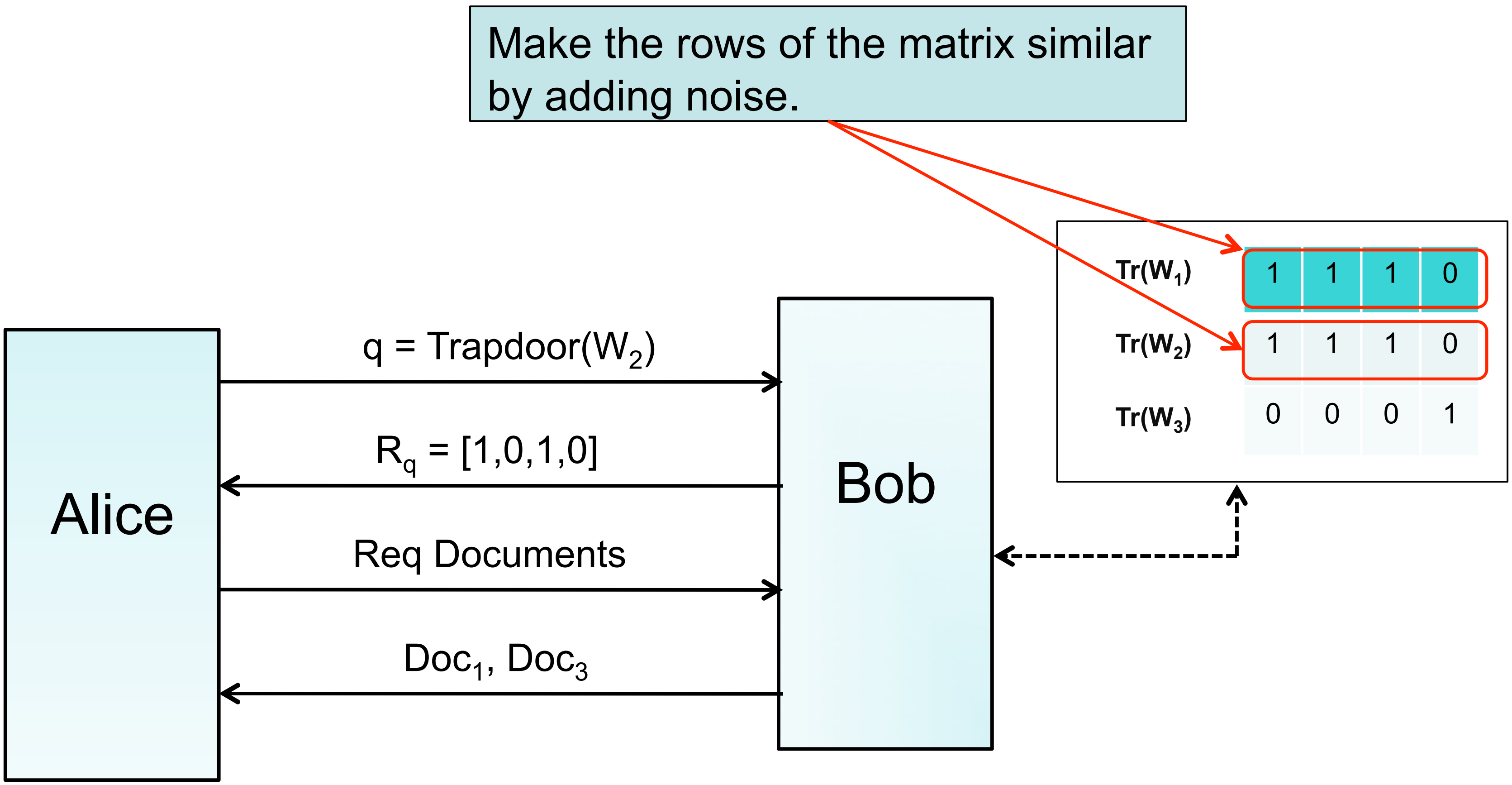
- Propose a simple noise addition based technique to counter against the attacks discussed in our work.

- Can work on any searchable encryption that leaks data access pattern.

Outline


Changing a few 0 to 1 (False Positive) is acceptable.
Tr(W1) 0 1 1 0
Tr(W2) 1 0 1 0
Tr(W3) 0 0 0 1
Alice
Bob
q = Trapdoor(W2)
Rq = [1,0,1,0]
Req Documents
Doc1, Doc3

Outline

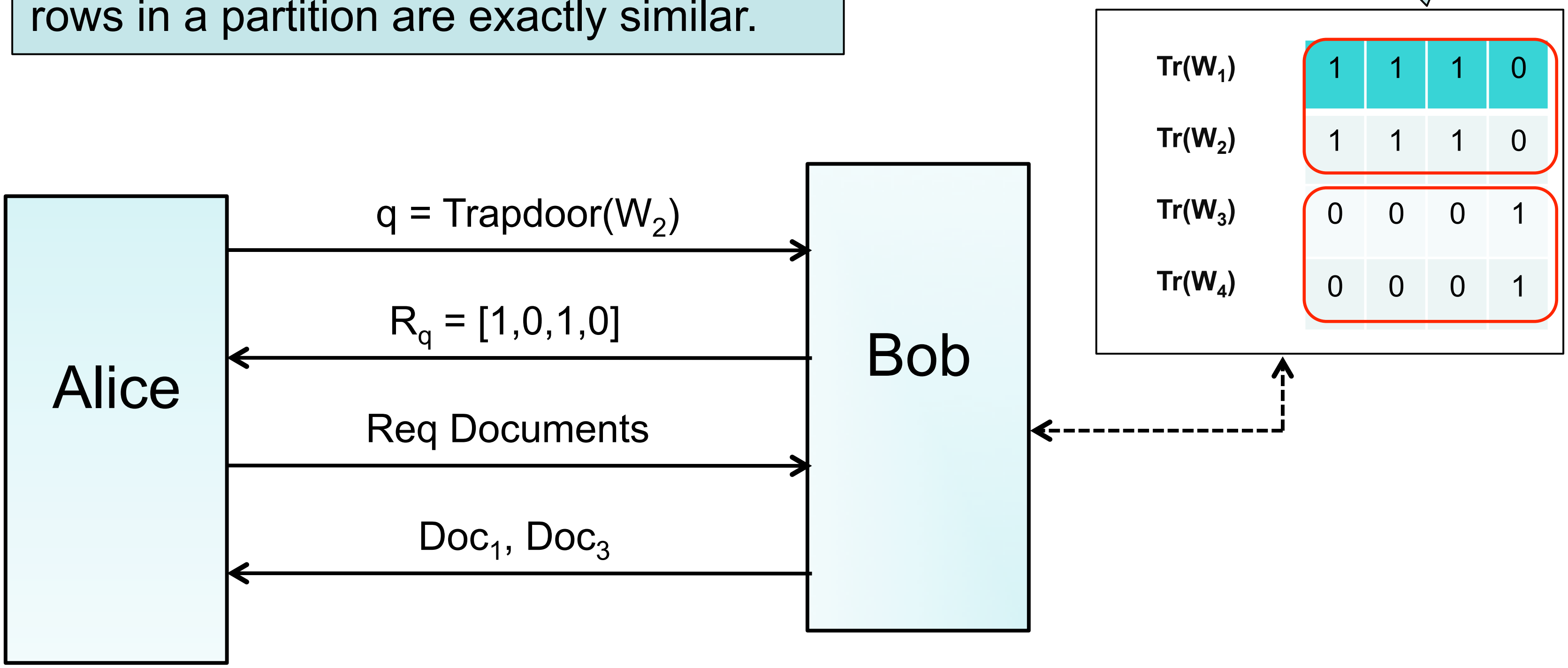


Make the rows of the matrix similar by adding noise.
Alice
Bob
q = Trapdoor(W2)
Rq = [1,0,1,0]
Req Documents
Doc1, Doc3
Tr(W1) 1 1 1 0
Tr(W2) 1 1 1 0
Tr(W3) 0 0 0 1

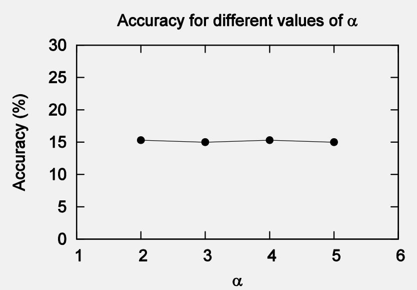
Privacy Definition: (α, 0)-secure Index

There exists a complete partitioning of the rows of an Index such that each partition has at least α rows and all the rows in a partition are exactly similar.

An (2, 0)- secure Index

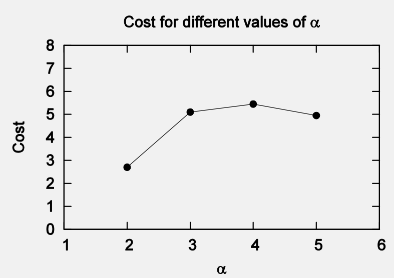


Experiment Results



Parameters

- Keyword Set Size: 1500
- Query Set Size: 150
- Known Query: 15%
- # Documents: 30109



Parameters

- Keyword Set Size: 1500
- Query Set Size: 150
- Known Query: 15%
- # Documents: 30109

Conclusion

- Access Pattern can be exploited to infer sensitive information.
- Simple noise addition based schemes can thwart some of the attacks successfully.

Questions?

Thank You.