Doctoral Defense | Final Oral Examination

Secure and Efficient Query Processing in Outsourced Databases

Range Queries [BKR19; Bog+21], Point Queries [Bog+21], *k*NN Queries [BKOZ22]

Dmytro Bogatov

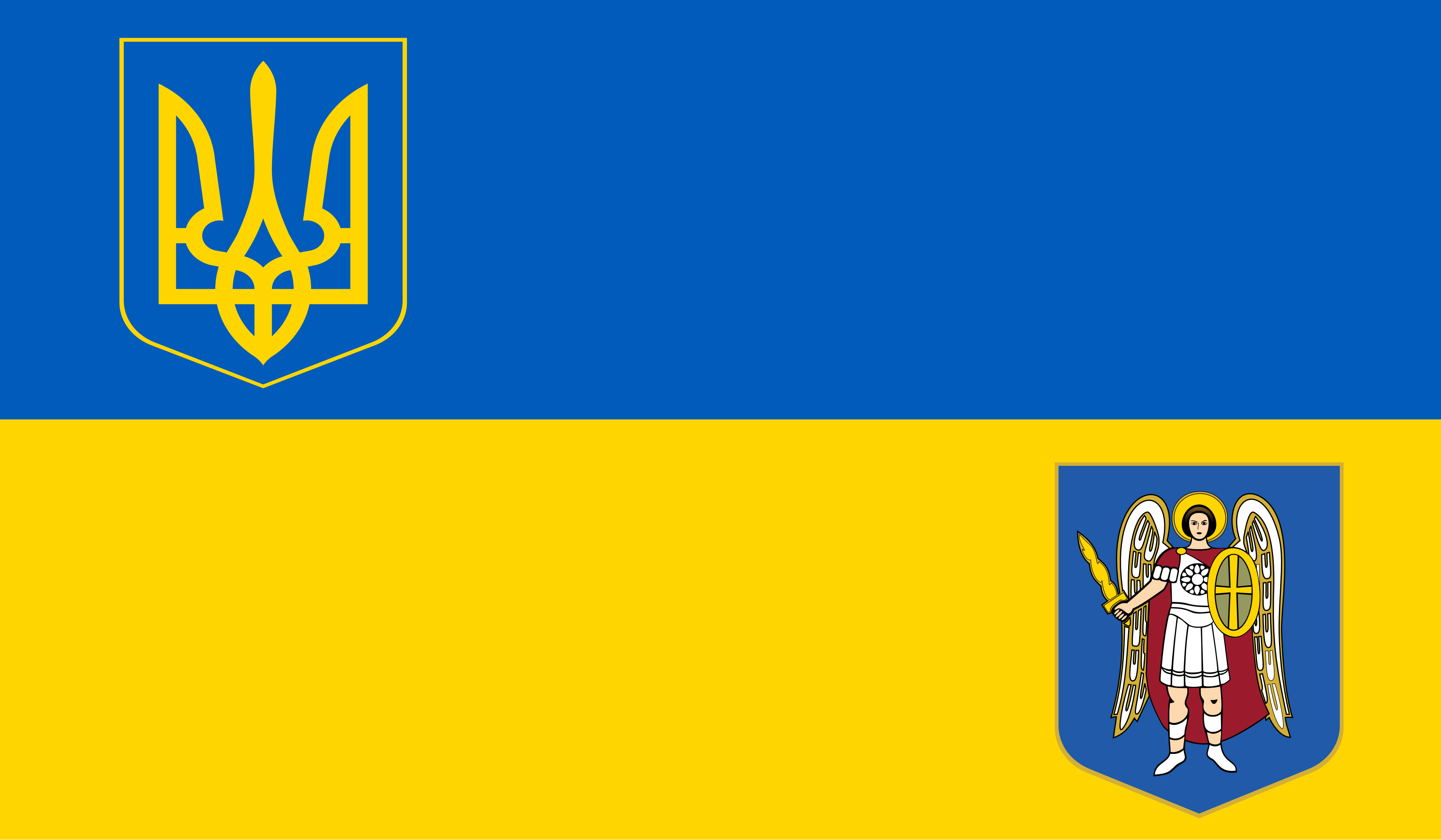
dmytro@bu.edu

Built from *3c92001c* on February 16, 2023

Boston University

Graduate School of Arts and Sciences

Department of Computer Science



Introduction

Motivation and overview

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - *k*NN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

Motivation and overview

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - *k*NN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

Motivation and overview

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - kNN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - *k*NN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - kNN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

- With vast amounts of data, organizations choose to use cloud
- Challenge: solutions must be both **secure** and **efficient**
- Security models for an outsourced database system
  - **Snapshot** adversary: steal the hard drive and RAM snapshot
  - **Persistent** adversary: continuously monitor the entire server
- Query types: `SELECT * FROM t1`
  - Point queries: `WHERE zip = '02215'`
  - Range queries: `WHERE age BETWEEN 18 AND 65`
  - kNN queries: `ORDER BY location <-> '(29.9691,-95.6972)' LIMIT 5`

Works published during the Ph.D. program

[BKR19] **Dmytro Bogatov**, George Kollios, and Leonid Reyzin. “A comparative evaluation of order-revealing encryption schemes and secure range-query protocols”. In: *Proceedings of the VLDB Endowment* 12.8 (2019), pp. 933–947. DOI: 10.14778/3324301.3324309

[Bog+21] **Dmytro Bogatov**, Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O’Neill. “$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy”. In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS ’2021)*. 2021. DOI: 10.1145/3460120.3484786

[BKOZ22] **Dmytro Bogatov**, George Kollios, Adam O’Neill, and Hamed Zamani. “*k-anon*: Secure Similarity Search in Outsourced Databases”. Apr. 2022

[BCET21] **Dmytro Bogatov**, Angelo De Caro, Kaoutar Elkhiyaoui, and Björn Tackmann. “Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric”. In: *International Conference on Cryptology and Network Security*. Springer. 2021. DOI: 10.1007/978-3-030-92547-5_23

Works published during the Ph.D. program

[BKR19] **Dmytro Bogatov**, George Kollios, and Leonid Reyzin. "A comparative evaluation of order-revealing encryption schemes and secure range-query protocols". In: *Proceedings of the VLDB Endowment* 12.8 (2019), pp. 933–947. DOI: 10.14778/3324301.3324309

[Bog+21] **Dmytro Bogatov**, Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. "$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '2021)*. 2021. DOI: 10.1145/3460120.3484786

[BKOZ22] **Dmytro Bogatov**, George Kollios, Adam O'Neill, and Hamed Zamani. "*k-anon*: Secure Similarity Search in Outsourced Databases". Apr. 2022

[BCET21] **Dmytro Bogatov**, Angelo De Caro, Kaoutar Elkhiyaoui, and Björn Tackmann. "Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric". In: *International Conference on Cryptology and Network Security*. Springer. 2021. DOI: 10.1007/978-3-030-92547-5_23

My work

Works published during the Ph.D. program

[BKR19] **Dmytro Bogatov**, George Kollios, and Leonid Reyzin. "A comparative evaluation of order-revealing encryption schemes and secure range-query protocols". In: *Proceedings of the VLDB Endowment* 12.8 (2019), pp. 933–947. DOI: 10.14778/3324301.3324309

[Bog+21] **Dmytro Bogatov**, Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. "$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '2021)*. 2021. DOI: 10.1145/3460120.3484786

[BKOZ22] **Dmytro Bogatov**, George Kollios, Adam O'Neill, and Hamed Zamani. "*k-anon*: Secure Similarity Search in Outsourced Databases". Apr. 2022

[BCET21] **Dmytro Bogatov**, Angelo De Caro, Kaoutar Elkhiyaoui, and Björn Tackmann. "Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric". In: *International Conference on Cryptology and Network Security*. Springer. 2021. DOI: 10.1007/978-3-030-92547-5_23

Works published during the Ph.D. program

[BKR19] **Dmytro Bogatov**, George Kollios, and Leonid Reyzin. "A comparative evaluation of order-revealing encryption schemes and secure range-query protocols". In: *Proceedings of the VLDB Endowment* 12.8 (2019), pp. 933–947. DOI: 10.14778/3324301.3324309

[Bog+21] **Dmytro Bogatov**, Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. "$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '2021)*. 2021. DOI: 10.1145/3460120.3484786

[BKOZ22] **Dmytro Bogatov**, George Kollios, Adam O'Neill, and Hamed Zamani. "*k-anon*: Secure Similarity Search in Outsourced Databases". Apr. 2022

[BCET21] **Dmytro Bogatov**, Angelo De Caro, Kaoutar Elkhiyaoui, and Björn Tackmann. "Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric". In: *International Conference on Cryptology and Network Security*. Springer. 2021. DOI: 10.1007/978-3-030-92547-5_23

A comparative evaluation of order-revealing encryption schemes and secure range-query protocols [BKR19]

Survey of OPE / ORE schemes [BKR19]

The problem

- Model: snapshot, query type: range
- Performance / security tradeoff
- Heterogeneous security definitions and leakage profiles
- **Performance not well-understood**
  - Some schemes are not even implemented
  - Prototype implementation at best
  - Not benchmarked against one another
  - Use different primitive implementations
  - Each claims to be practical and secure

Our solution

- Analyzed security and leakages of the constructions under **a common framework**
- Analyzed theoretically performance of the schemes and protocols
- **Implemented and ran experiments**
  - Implemented 5 OPE / ORE schemes and 5 range query protocols
  - Used same language, framework and primitive implementations
  - Benchmarked primitives execution times
  - Counted **invocations of primitives** and **I/O requests**

ORE table | Protocols table | Plot

Survey of OPE / ORE schemes [BKR19]

The problem

- Model: snapshot, query type: range
- Performance / security tradeoff
- Heterogeneous security definitions and leakage profiles
- **Performance not well-understood**
  - Some schemes are not even implemented
  - Prototype implementation at best
  - Not benchmarked against one another
  - Use different primitive implementations
  - Each claims to be practical and secure

Our solution

- Analyzed security and leakages of the constructions under **a common framework**
- Analyzed theoretically performance of the schemes and protocols
- **Implemented and ran experiments**
  - Implemented 5 OPE / ORE schemes and 5 range query protocols
  - Used same language, framework and primitive implementations
  - Benchmarked primitives execution times
  - Counted **invocations of primitives** and **I/O requests**

ORE table | Protocols table | Plot

$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy [Bog+21]

The problem

- Previous solutions work in the snapshot model (adversary steals the hard drive)
- What about **persistent** adversary (malicious script with `root` permissions)?

  Model: persistent, query type: point and range
- Need to protect access pattern and communication volume
- Using ORAM to hide the access pattern

  Expensive, each request costs $\mathcal{O}(\log n)$ ORAM definition
- Adding fake records (noise) to the answer to hide the result size

  How much noise to add to have a guarantee and the least overhead?

  Adding a constant or a uniformly sampled noise is not an option

  Differential Privacy!

The problem

- Previous solutions work in the snapshot model (adversary steals the hard drive)
- What about **persistent** adversary (malicious script with `root` permissions)?

  Model: persistent, query type: point and range
- Need to protect access pattern and communication volume
- Using ORAM to hide the access pattern

  Expensive, each request costs $\mathcal{O}(\log n)$ ORAM definition
- Adding fake records (noise) to the answer to hide the result size

  How much noise to add to have a guarantee and the least overhead?

  Adding a constant or a uniformly sampled noise is not an option

  Differential Privacy!

The problem

- Previous solutions work in the snapshot model (adversary steals the hard drive)
- What about **persistent** adversary (malicious script with `root` permissions)?

  Model: persistent, query type: point and range
- Need to protect access pattern and communication volume
- Using ORAM to hide the access pattern

  Expensive, each request costs $\mathcal{O}(\log n)$ ORAM definition
- Adding fake records (noise) to the answer to hide the result size

  How much noise to add to have a guarantee and the least overhead?

  Adding a constant or a uniformly sampled noise is not an option

  Differential Privacy!

Definition (Differential Privacy, adapted from [Dwo+06; DMNS06])

A randomized algorithm A is (ϵ, δ)-differentially private if for all $\mathcal{D}_1 \sim \mathcal{D}_2 \in \mathcal{X}^n$, and for all subsets $\mathcal{O}$ of the output space of A,

$$\Pr\left[\mathrm{A}\left(\mathcal{D}_1\right) \in \mathcal{O}\right] \leq \exp(\epsilon) \cdot \Pr\left[\mathrm{A}\left(\mathcal{D}_2\right) \in \mathcal{O}\right] + \delta \ .$$

How to make sense of it?

- Differential Privacy is a property of an algorithm
  What about ϵ and δ?
- How to construct such an algorithm?
  Laplace Perturbation Method!
- What if negative value is sampled?
  Cannot truncate one side, must shift entire distribution

Definition (Differential Privacy, adapted from [Dwo+06; DMNS06])

A randomized algorithm A is (ϵ, δ)-differentially private if for all $\mathcal{D}_1 \sim \mathcal{D}_2 \in \mathcal{X}^n$, and for all subsets $\mathcal{O}$ of the output space of A,

$$\Pr\left[\mathrm{A}\left(\mathcal{D}_1\right) \in \mathcal{O}\right] \leq \exp(\epsilon) \cdot \Pr\left[\mathrm{A}\left(\mathcal{D}_2\right) \in \mathcal{O}\right] + \delta\ .$$

How to make sense of it?

- Differential Privacy is a property of an algorithm

  What about ϵ and δ?

- How to construct such an algorithm?

  Laplace Perturbation Method!

- What if negative value is sampled?

  Cannot truncate one side, must shift entire distribution

Definition (Differential Privacy, adapted from [Dwo+06; DMNS06])

A randomized algorithm A is (ϵ, δ)-differentially private if for all $\mathcal{D}_1 \sim \mathcal{D}_2 \in \mathcal{X}^n$, and for all subsets $\mathcal{O}$ of the output space of A,

$$\Pr\left[\mathrm{A}\left(\mathcal{D}_1\right) \in \mathcal{O}\right] \leq \exp(\epsilon) \cdot \Pr\left[\mathrm{A}\left(\mathcal{D}_2\right) \in \mathcal{O}\right] + \delta \ .$$

How to make sense of it?

- Differential Privacy is a property of an algorithm

  What about ϵ and δ?

- How to construct such an algorithm?

  Laplace Perturbation Method!

- What if negative value is sampled?

  Cannot truncate one side, must shift entire distribution

Definition (Differential Privacy, adapted from [Dwo+06; DMNS06])

A randomized algorithm A is (ϵ, δ)-differentially private if for all $\mathcal{D}_1 \sim \mathcal{D}_2 \in \mathcal{X}^n$, and for all subsets $\mathcal{O}$ of the output space of A,

$$\Pr\left[\mathrm{A}\left(\mathcal{D}_1\right) \in \mathcal{O}\right] \leq \exp(\epsilon) \cdot \Pr\left[\mathrm{A}\left(\mathcal{D}_2\right) \in \mathcal{O}\right] + \delta\ .$$

How to make sense of it?

- Differential Privacy is a property of an algorithm

  What about ϵ and δ?
- How to construct such an algorithm?

  Laplace Perturbation Method!
- What if negative value is sampled?

  Cannot truncate one side, must shift entire distribution

Definition (Differential Privacy, adapted from [Dwo+06; DMNS06])

A randomized algorithm A is (ϵ, δ)-differentially private if for all $\mathcal{D}_1 \sim \mathcal{D}_2 \in \mathcal{X}^n$, and for all subsets $\mathcal{O}$ of the output space of A,

$$\Pr\left[\mathrm{A}\left(\mathcal{D}_1\right) \in \mathcal{O}\right] \leq \exp(\epsilon) \cdot \Pr\left[\mathrm{A}\left(\mathcal{D}_2\right) \in \mathcal{O}\right] + \delta \ .$$

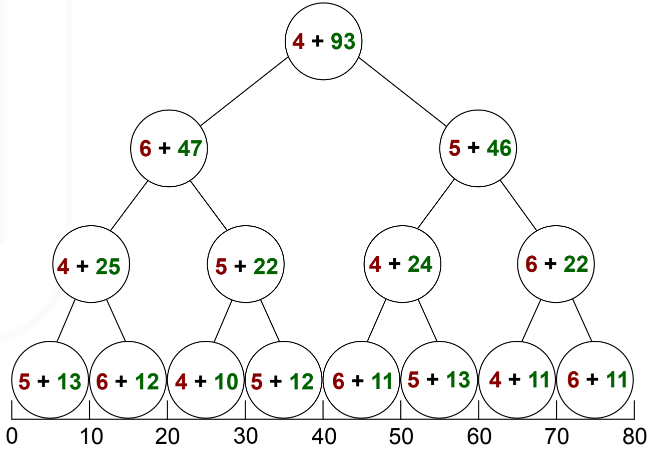
How to make sense of it?

- Differential Privacy is a property of an algorithm

  What about ϵ and δ?
- How to construct such an algorithm?

  Laplace Perturbation Method!
- What if negative value is sampled?

  Cannot truncate one side, must shift entire distribution



Definition (Computationally Differentially Private Outsourced Database System)

We say that an outsourced database system Π is (ϵ, δ)-computationally differentially private (a.k.a. CDP-ODB) if for every polynomial time distinguishing adversary $\mathcal{A}$, for every neighboring databases $\mathcal{D} \sim \mathcal{D}'$, and for every query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ where $m = \text{poly}(\lambda)$,

$$\Pr\left[\mathcal{A}\left(1^\lambda, \textsc{View}_{\Pi,\mathcal{S}}\left(\mathcal{D}, q_1, \ldots, q_m\right)\right) = 1\right] \leq \exp\epsilon \cdot \Pr\left[\mathcal{A}\left(1^\lambda, \textsc{View}_{\Pi,\mathcal{S}}\left(\mathcal{D}', q_1, \ldots, q_m\right)\right) = 1\right] + \delta + \text{negl}(\lambda) \ ,$$

the probability is over the randomness of the distinguishing adversary $\mathcal{A}$ and the protocol Π.

Note:

- Entire view of the adversary is DP-protected
- Implies protection against **communication volume** and **access pattern** leakages
- Query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ is fixed See why
- $\text{negl}(\lambda)$ needed for the computational (as opposed to information-theoretical) DP definition

Definition (Computationally Differentially Private Outsourced Database System)

We say that an outsourced database system Π is (ϵ, δ)-computationally differentially private (a.k.a. CDP-ODB) if for every polynomial time distinguishing adversary $\mathcal{A}$, for every neighboring databases $\mathcal{D} \sim \mathcal{D}'$, and for every query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ where $m = \text{poly}(\lambda)$,

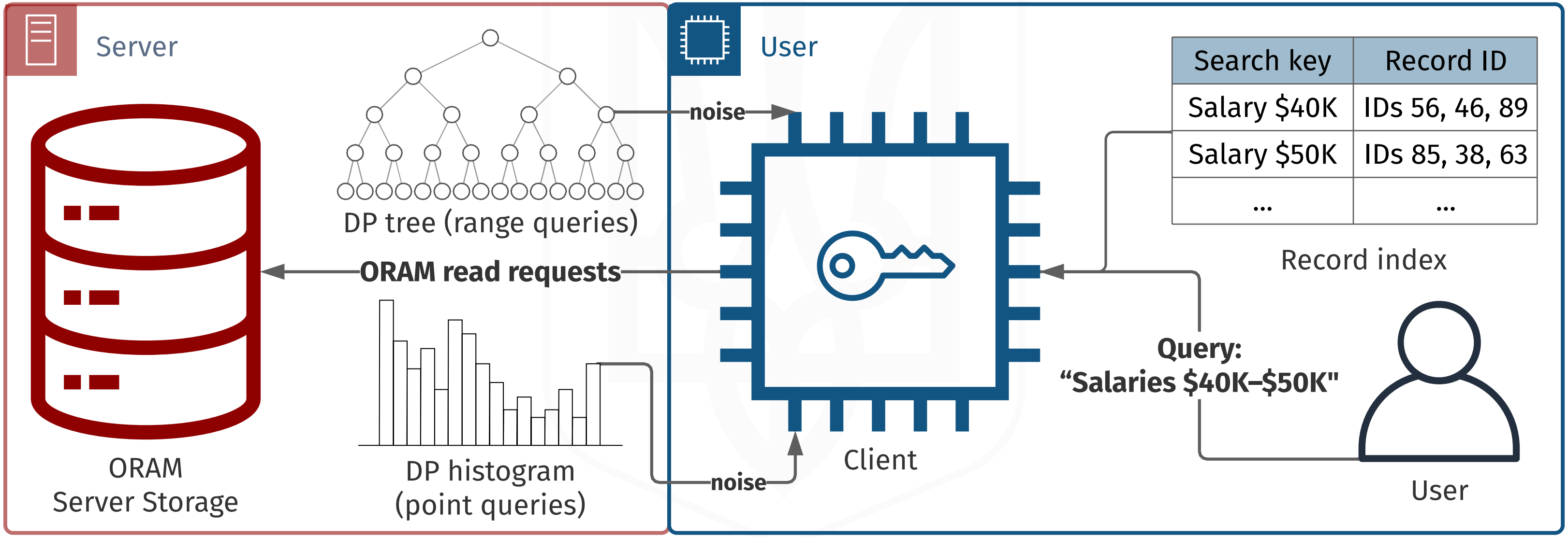
$$\Pr\left[\mathcal{A}\left(1^\lambda, \textsc{View}_{\Pi,\mathcal{S}}\left(\mathcal{D}, q_1, \ldots, q_m\right)\right) = 1\right] \leq \\ \exp \epsilon \cdot \Pr\left[\mathcal{A}\left(1^\lambda, \textsc{View}_{\Pi,\mathcal{S}}\left(\mathcal{D}', q_1, \ldots, q_m\right)\right) = 1\right] + \delta + \text{negl}(\lambda) \,,$$

the probability is over the randomness of the distinguishing adversary $\mathcal{A}$ and the protocol Π.

Note:

- Entire view of the adversary is DP-protected
- Implies protection against **communication volume** and **access pattern** leakages
- Query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ is fixed See why
- $\text{negl}(\lambda)$ needed for the computational (as opposed to information-theoretical) DP definition

Single-Threaded $\mathcal{E}$psolute protocol



- Single-threaded version is prohibitively slow, must parallelize
  Assume single-threaded solution generates r real and f fake records
- Split $\mathcal{U}$ and $\mathcal{S}$ state into m ORAMs, run as separate machines
- Partition records randomly (by ID) into m partitions, generate m record indexes
- What to do about sanitizer $\mathcal{DS}$?

Π_{separate}: separate sanitizer $\mathcal{DS}$ per ORAM

- Each ORAM incurs noise comparable to f
- Win by splitting ORAM work r into m partitions and lose by multiplying noise f times m
- That is, all ORAMs are processing $r + mf$ records in parallel

Π_{shared}: shared sanitizer $\mathcal{DS}$ for all ORAMs

- Same number of total records per ORAM
- Generated noise is larger than f (say, αf), but split among m ORAMs
- That is, all ORAMs are processing $r+\alpha f$ records in parallel

Π_{shared} wins if $\alpha < m$, which it is for almost all values of m ($m \gtrsim 4$)

Parallel $\mathcal{E}$psolute

- Single-threaded version is prohibitively slow, must parallelize

  Assume single-threaded solution generates r real and f fake records
- Split $\mathcal{U}$ and $\mathcal{S}$ state into m ORAMs, run as separate machines
- Partition records randomly (by ID) into m partitions, generate m record indexes
- What to do about sanitizer $\mathcal{DS}$?

Π_{separate}: separate sanitizer $\mathcal{DS}$ per ORAM

- Each ORAM incurs noise comparable to f
- Win by splitting ORAM work r into m partitions and lose by multiplying noise f times m
- That is, all ORAMs are processing $r + mf$ records in parallel

Π_{shared}: shared sanitizer $\mathcal{DS}$ for all ORAMs

- Same number of total records per ORAM
- Generated noise is larger than f (say, αf), but split among m ORAMs
- That is, all ORAMs are processing $r+\alpha f$ records in parallel

Π_{shared} wins if $\alpha < m$, which it is for almost all values of m ($m \gtrsim 4$)

- Single-threaded version is prohibitively slow, must parallelize

  Assume single-threaded solution generates r real and f fake records
- Split $\mathcal{U}$ and $\mathcal{S}$ state into m ORAMs, run as separate machines
- Partition records randomly (by ID) into m partitions, generate m record indexes
- What to do about sanitizer $\mathcal{DS}$?

Π_{separate}: separate sanitizer $\mathcal{DS}$ per ORAM

- Each ORAM incurs noise comparable to f
- Win by splitting ORAM work r into m partitions and lose by multiplying noise f times m
- That is, all ORAMs are processing $r + mf$ records in parallel

Π_{shared}: shared sanitizer $\mathcal{DS}$ for all ORAMs

- Same number of total records per ORAM
- Generated noise is larger than f (say, αf), but split among m ORAMs
- That is, all ORAMs are processing $r+\alpha f$ records in parallel

Π_{shared} wins if $\alpha < m$, which it is for almost all values of m ($m \gtrsim 4$)

Parallel $\mathcal{E}$psolute

- Single-threaded version is prohibitively slow, must parallelize

  Assume single-threaded solution generates r real and f fake records
- Split $\mathcal{U}$ and $\mathcal{S}$ state into m ORAMs, run as separate machines
- Partition records randomly (by ID) into m partitions, generate m record indexes
- What to do about sanitizer $\mathcal{DS}$?

Π_{separate}: separate sanitizer $\mathcal{DS}$ per ORAM

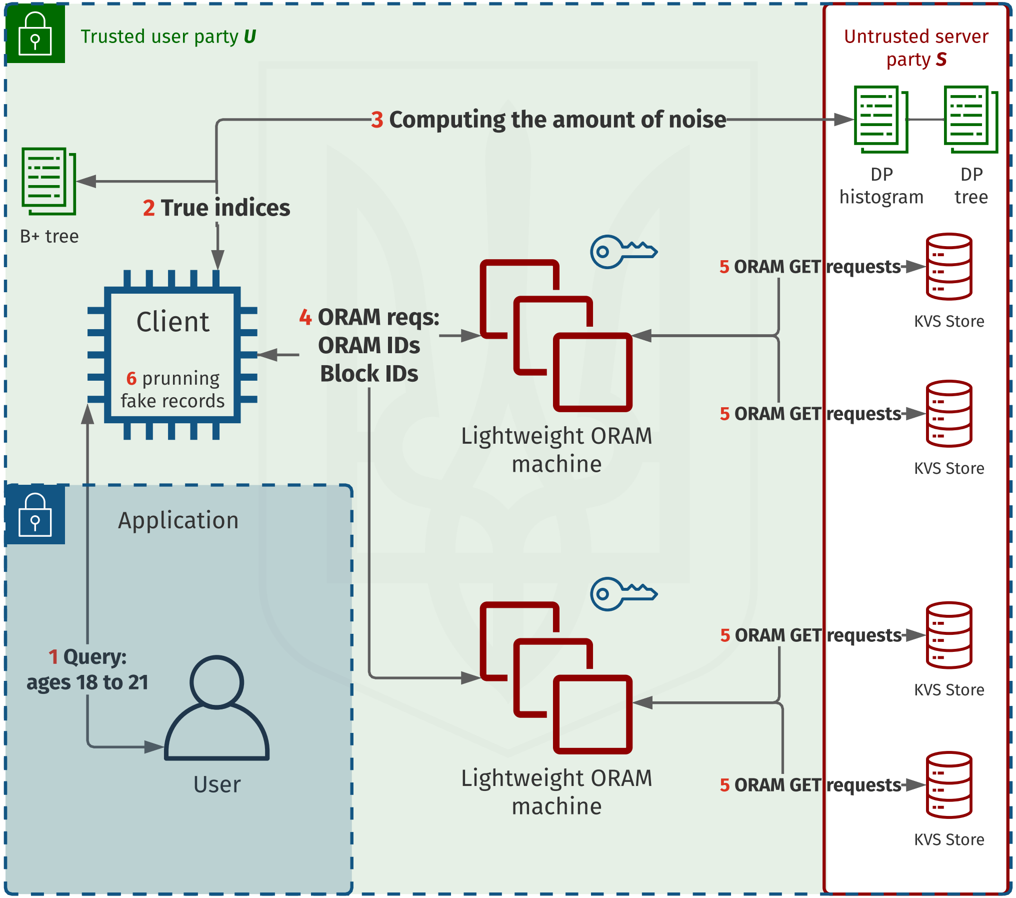
- Each ORAM incurs noise comparable to f
- Win by splitting ORAM work r into m partitions and lose by multiplying noise f times m
- That is, all ORAMs are processing $r + mf$ records in parallel

Π_{shared}: shared sanitizer $\mathcal{DS}$ for all ORAMs

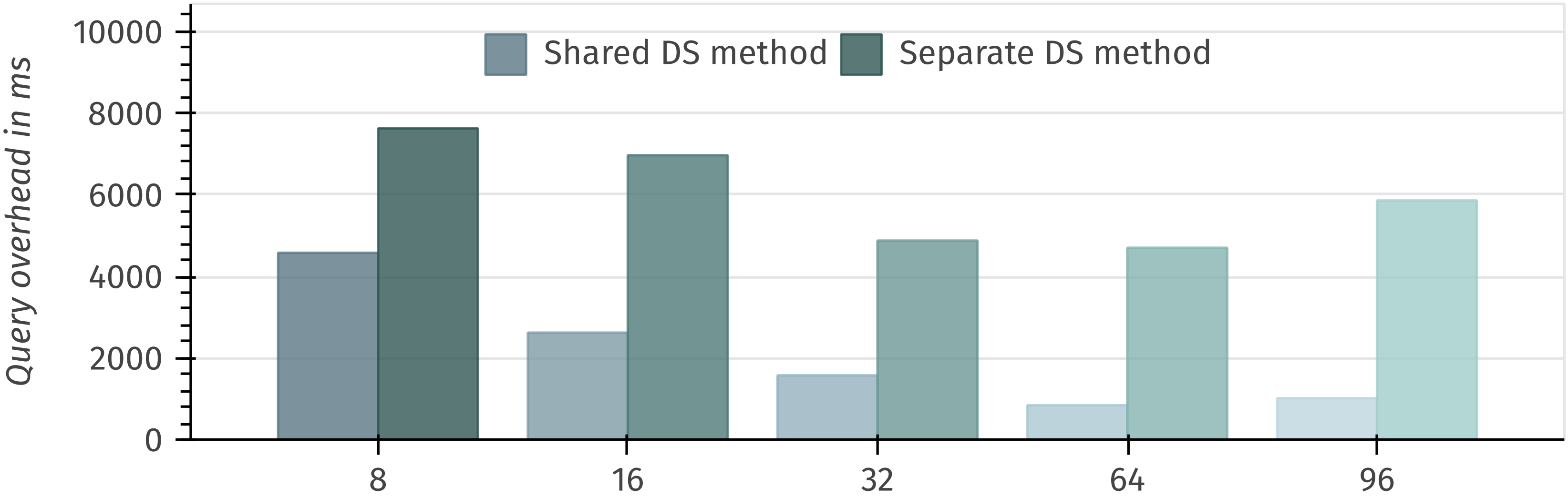
- Same number of total records per ORAM
- Generated noise is larger than f (say, αf), but split among m ORAMs
- That is, all ORAMs are processing $r+\alpha f$ records in parallel

Π_{shared} wins if $\alpha < m$, which it is for almost all values of m ($m \gtrapprox 4$)

Parallel $\mathcal{E}$psolute diagram (with improvements)

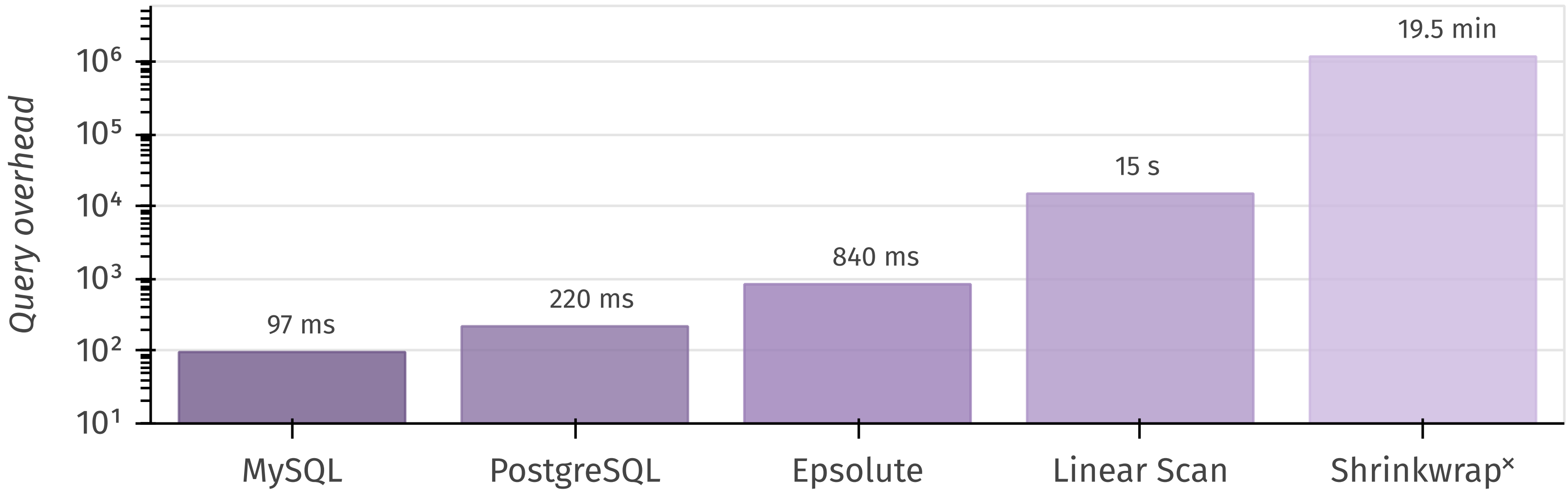


Experiments: scalability



Scalability measurements for Π_{shared} and Π_{separate} ($\mathcal{DS}$ is a DP sanitizer)

Experiments: against other mechanisms



Different range-query mechanisms (log scale).

Default setting: 10^6 4 KiB uniformly-sampled records with the domain 10^4.

k-anon: Secure Similarity Search in Outsourced Databases [BKOZ22]

- Model: snapshot, query type: kNN in arbitrary dimensions
- Nearest-neighbor search needs definitions of *object* and *distance*
  - *Object* can be 2D/3D location, or a document embedding (high-dimensional vector)
  - *Distance* can be a Euclidean distance or inner (dot) product distance
  - *Query* then can be “5 closest restaurants” or “3 most similar documents”
- Our approach is to apply an *approximate property-preserving encryption* on objects
  - Query protocol is then similar to OPE / ORE
  - Existing nearest-neighbor search algorithms then work naturally
- **Study how accuracy of search and efficiency of attacks drop with higher security**

General idea

- Model: snapshot, query type: *k*NN in arbitrary dimensions
- Nearest-neighbor search needs definitions of *object* and *distance*
  - *Object* can be 2D/3D location, or a document embedding (high-dimensional vector)
  - *Distance* can be a Euclidean distance or inner (dot) product distance
  - *Query* then can be "5 closest restaurants" or "3 most similar documents"
- Our approach is to apply an *approximate property-preserving encryption* on objects
  - Query protocol is then similar to OPE / ORE
  - Existing nearest-neighbor search algorithms then work naturally
- Study how accuracy of search and efficiency of attacks drop with higher security

General idea

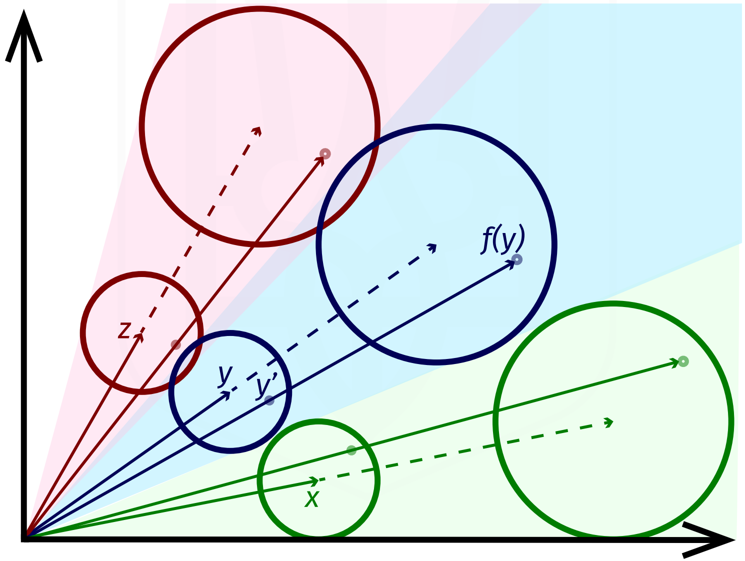
- Model: snapshot, query type: kNN in arbitrary dimensions
- Nearest-neighbor search needs definitions of *object* and *distance*
  - *Object* can be 2D/3D location, or a document embedding (high-dimensional vector)
  - *Distance* can be a Euclidean distance or inner (dot) product distance
  - *Query* then can be "5 closest restaurants" or "3 most similar documents"
- Our approach is to apply an *approximate property-preserving encryption* on objects
  - Query protocol is then similar to OPE / ORE
  - Existing nearest-neighbor search algorithms then work naturally
- Study how accuracy of search and efficiency of attacks drop with higher security

General idea

- Model: snapshot, query type: kNN in arbitrary dimensions
- Nearest-neighbor search needs definitions of *object* and *distance*
  - *Object* can be 2D/3D location, or a document embedding (high-dimensional vector)
  - *Distance* can be a Euclidean distance or inner (dot) product distance
  - *Query* then can be "5 closest restaurants" or "3 most similar documents"
- Our approach is to apply an *approximate property-preserving encryption* on objects
  - Query protocol is then similar to OPE / ORE
  - Existing nearest-neighbor search algorithms then work naturally
- **Study how accuracy of search and efficiency of attacks drop with higher security**

$$\forall x, y, z \in \mathbb{X} : \textsc{Dist}(x, y) < \textsc{Dist}(x, z) - \beta \implies \textsc{Dist}(f(x), f(y)) < \textsc{Dist}(f(x), f(z))$$

If distance between x and z is larger than the distance between x and y by more than β, then the encryption of z will be further than the encryption of y from the encryption of x.



Distance Comparison Preserving Encryption scheme [FGHO21]

DCPE algorithms

Search accuracy and TREC dataset

- Setup and query protocols: for given β
  - Generate encryption key
  - Encrypt dataset and queries set with β
  - Run queries using conventional nearest-neighbor search (e.g., FAISS)
  - Report search accuracy metrics
- TREC 2020 dataset is 8.8M documents embedded with fine-tuned BERT (768 dimensions)

  Thanks Hamed Zamani for the dataset
- Query is a 768-dimensional embedding asking for $k = 1000$ closest documents

  TREC has a set of documents, a set of topics (questions), and relevance judgments (right answers)
- We report result set distance and difference, and ranking quality Recall, MRR and nDCG

  Set distance and difference measure pure kNN accuracy

  Recall, MRR and nDCG report ranking quality using relevance, common in information retrieval literature

Search accuracy and TREC dataset

- Setup and query protocols: for given β
  - Generate encryption key
  - Encrypt dataset and queries set with β
  - Run queries using conventional nearest-neighbor search (e.g., FAISS)
  - Report search accuracy metrics
- TREC 2020 dataset is 8.8M documents embedded with fine-tuned BERT (768 dimensions)

  Thanks Hamed Zamani for the dataset
- Query is a 768-dimensional embedding asking for $k = 1000$ closest documents

  TREC has a set of documents, a set of topics (questions), and relevance judgments (right answers)
- We report result set distance and difference, and ranking quality Recall, MRR and nDCG

  Set distance and difference measure pure kNN accuracy

  Recall, MRR and nDCG report ranking quality using relevance, common in information retrieval literature

Search accuracy and TREC dataset

- Setup and query protocols: for given β
  - Generate encryption key
  - Encrypt dataset and queries set with β
  - Run queries using conventional nearest-neighbor search (e.g., FAISS)
  - Report search accuracy metrics
- TREC 2020 dataset is 8.8M documents embedded with fine-tuned BERT (768 dimensions)

  Thanks Hamed Zamani for the dataset
- Query is a 768-dimensional embedding asking for $k = 1000$ closest documents

  TREC has a set of documents, a set of topics (questions), and relevance judgments (right answers)
- We report result set distance and difference, and ranking quality Recall, MRR and nDCG

  Set distance and difference measure pure kNN accuracy

  Recall, MRR and nDCG report ranking quality using relevance, common in information retrieval literature

Search accuracy and TREC dataset

- Setup and query protocols: for given β
  - Generate encryption key
  - Encrypt dataset and queries set with β
  - Run queries using conventional nearest-neighbor search (e.g., FAISS)
  - Report search accuracy metrics
- TREC 2020 dataset is 8.8M documents embedded with fine-tuned BERT (768 dimensions)

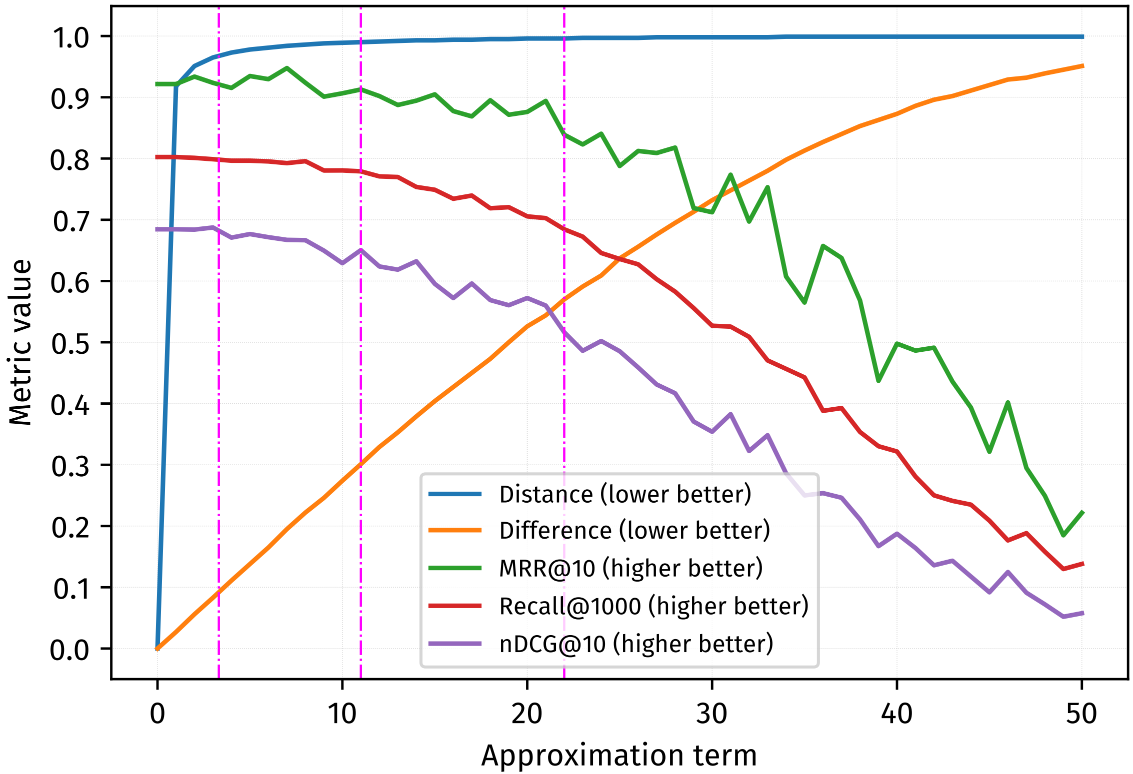
  Thanks Hamed Zamani for the dataset
- Query is a 768-dimensional embedding asking for $k = 1000$ closest documents

  TREC has a set of documents, a set of topics (questions), and relevance judgments (right answers)
- We report result set distance and difference, and ranking quality Recall, MRR and nDCG

  Set distance and difference measure pure kNN accuracy

  Recall, MRR and nDCG report ranking quality using relevance, common in information retrieval literature

Search accuracy results



Rank quality metrics, result set distance and difference for $\beta \in \{0, 1, \ldots, 50\}$

Black-box inversion attack [SR20]

- ML model trains on document-embedding pairs and predicts a *set of words* from embedding

  Model is an LSTM trained for 30 epochs

  Original attack used BookCorpus [Zhu+15] dataset, but we will use TREC
- We evaluate the attack on encrypted embeddings
  - We also add plaintext and random embeddings for the baselines
  - **Public model**: adversary can use *the embedding model*, therefore, trains on plaintexts
  - **Private model**: adversary can only use *the entire system*, therefore, trains on ciphertexts
  - In both cases the model predicts the words from the *encrypted* embedding
- We measure precision, recall and F_1 score along with *the percent of stop-words*

  Stop-words are common words like “a”, “the”, pronouns, even punctuation and digits

Black-box inversion attack [SR20]

- ML model trains on document-embedding pairs and predicts a *set of words* from embedding
  Model is an LSTM trained for 30 epochs
  Original attack used BookCorpus [Zhu+15] dataset, but we will use TREC
- We evaluate the attack on encrypted embeddings
  - We also add plaintext and random embeddings for the baselines
  - **Public model**: adversary can use *the embedding model*, therefore, trains on plaintexts
  - **Private model**: adversary can only use *the entire system*, therefore, trains on ciphertexts
  - In both cases the model predicts the words from the *encrypted* embedding
- We measure precision, recall and F_1 score along with *the percent of stop-words*
  Stop-words are common words like “a”, “the”, pronouns, even punctuation and digits

Black-box inversion attack [SR20]

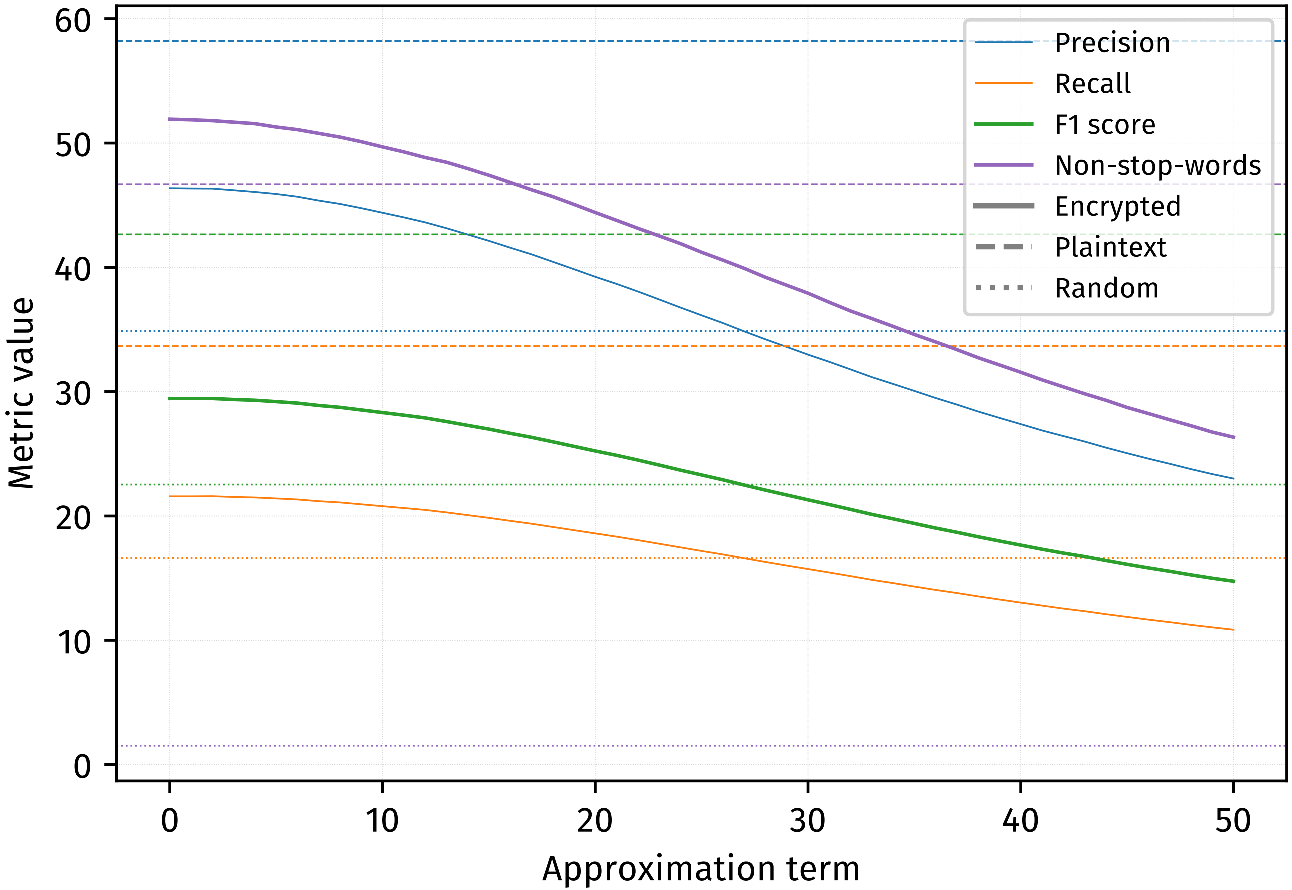
- ML model trains on document-embedding pairs and predicts a *set of words* from embedding

  Model is an LSTM trained for 30 epochs

  Original attack used BookCorpus [Zhu+15] dataset, but we will use TREC
- We evaluate the attack on encrypted embeddings
  - We also add plaintext and random embeddings for the baselines
  - **Public model**: adversary can use *the embedding model*, therefore, trains on plaintexts
  - **Private model**: adversary can only use *the entire system*, therefore, trains on ciphertexts
  - In both cases the model predicts the words from the *encrypted* embedding
- We measure precision, recall and F_1 score along with *the percent of stop-words*

  Stop-words are common words like “a”, “the”, pronouns, even punctuation and digits

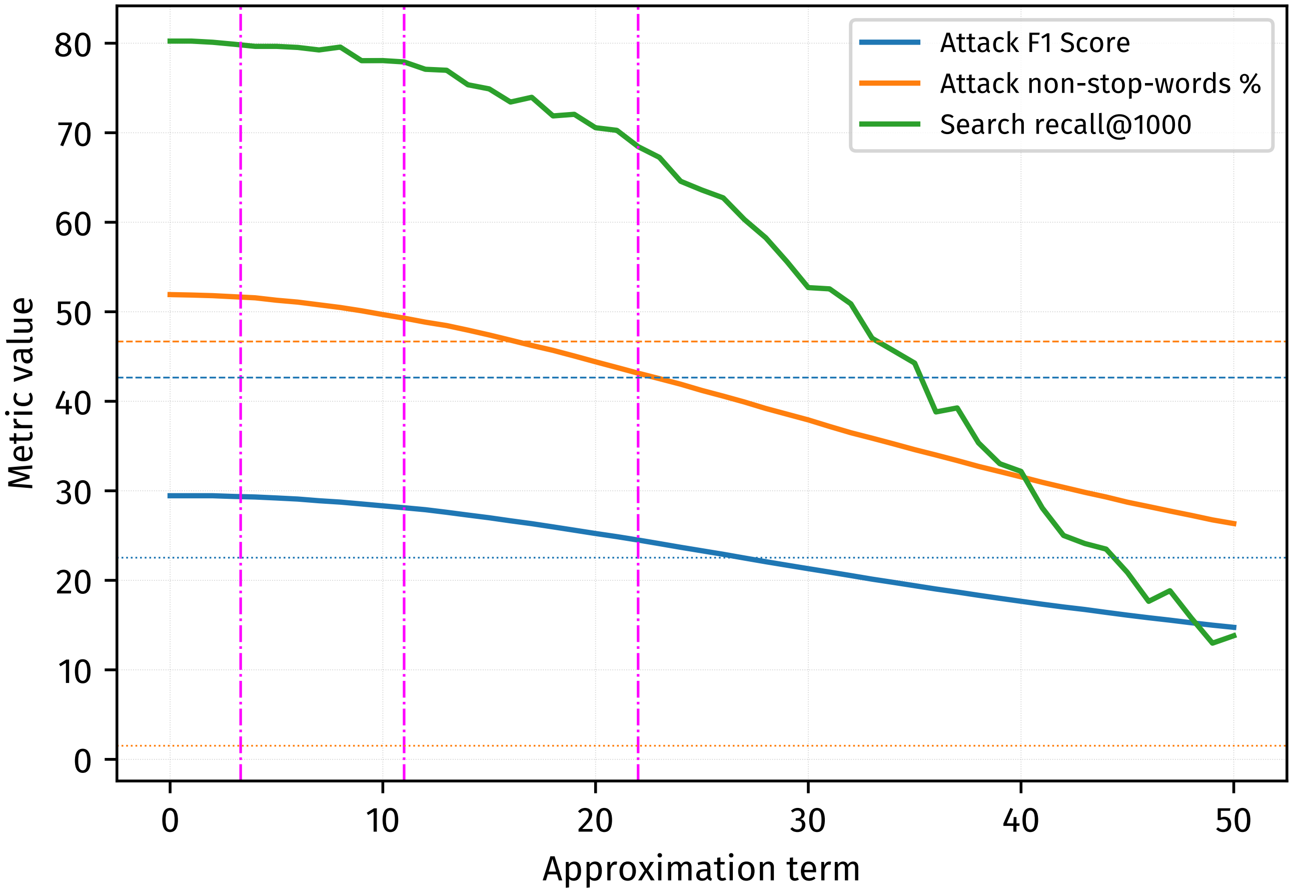
Attack performance results: public model (trained on plaintext)



Attack performance results: private model (trained on ciphertext)

Dataset	Precision	Recall	**F_1 score**	% of non-stop-words
Encrypted with $\beta = 0.0$	38.84	27.64	**32.30**	2.31
Encrypted with $\beta = 4.0$	36.28	26.61	**30.70**	3.21
Random embeddings	36.07	26.61	**30.62**	0.00

Tradeoff between search accuracy and attack performance (public model, trained on plaintext)



Conclusions

Future work directions

- Focus on practicality and reproducibility!
- Property-preserving encryption is practical [BKR19; BKOZ22]
  - May not be ideally-secure, and does not have to be
  - Benchmark the scheme and quantify the leakage
- Hardware gets cheaper, consider "heavy" primitives and protocols [Bog+21]
  - ORAM, homomorphic encryption, garbled circuits, zero-knowledge proofs, etc
  - Performance may be acceptable with optimizations, specialized hardware and parallelization
- More database query types in outsourced model
  - `JOIN`, `GROUP BY`, `AGGREGATE`, custom predicates, etc

Future work directions

- **Focus on practicality and reproducibility!**
- Property-preserving encryption is practical [BKR19; BKOZ22]
  - May not be ideally-secure, and does not have to be
  - Benchmark the scheme and quantify the leakage
- Hardware gets cheaper, consider “heavy” primitives and protocols [Bog+21]
  - ORAM, homomorphic encryption, garbled circuits, zero-knowledge proofs, etc
  - Performance may be acceptable with optimizations, specialized hardware and parallelization
- More database query types in outsourced model
  - `JOIN`, `GROUP BY`, `AGGREGATE`, custom predicates, etc

Future work directions

- **Focus on practicality and reproducibility!**
- Property-preserving encryption is practical [BKR19; BKOZ22]
  - May not be ideally-secure, and does not have to be
  - Benchmark the scheme and quantify the leakage
- Hardware gets cheaper, consider "heavy" primitives and protocols [Bog+21]
  - ORAM, homomorphic encryption, garbled circuits, zero-knowledge proofs, etc
  - Performance may be acceptable with optimizations, specialized hardware and parallelization
- More database query types in outsourced model
  - `JOIN`, `GROUP BY`, `AGGREGATE`, custom predicates, etc

- **Focus on practicality and reproducibility!**
- Property-preserving encryption is practical [BKR19; BKOZ22]
  - May not be ideally-secure, and does not have to be
  - Benchmark the scheme and quantify the leakage
- Hardware gets cheaper, consider "heavy" primitives and protocols [Bog+21]
  - ORAM, homomorphic encryption, garbled circuits, zero-knowledge proofs, etc
  - Performance may be acceptable with optimizations, specialized hardware and parallelization
- More database query types in outsourced model
  - `JOIN`, `GROUP BY`, `AGGREGATE`, custom predicates, etc

Acknowledgements

Doctoral Defense | Final Oral Examination

Secure and Efficient Query Processing in Outsourced Databases

Range Queries [BKR19; Bog+21], Point Queries [Bog+21], *k*NN Queries [BKOZ22]

Dmytro Bogatov
dmytro@bu.edu

Built from *3c92001c* on February 16, 2023

Boston University
Graduate School of Arts and Sciences
Department of Computer Science

References

[BKOZ22] **Dmytro Bogatov**, George Kollios, Adam O'Neill, and Hamed Zamani. "*k-anon*: Secure Similarity Search in Outsourced Databases". Apr. 2022.

[BCET21] **Dmytro Bogatov**, Angelo De Caro, Kaoutar Elkhiyaoui, and Björn Tackmann. "Anonymous Transactions with Revocation and Auditing in Hyperledger Fabric". In: *International Conference on Cryptology and Network Security*. Springer. 2021. DOI: 10.1007/978-3-030-92547-5_23.

[Bog+21] **Dmytro Bogatov**, Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. "$\mathcal{E}$psolute: Efficiently Querying Databases While Providing Differential Privacy". In: *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS '2021)*. 2021. DOI: 10.1145/3460120.3484786.

[FGHO21] Georg Fuchsbauer, Riddhi Ghosal, Nathan Hauke, and Adam O'Neill. *Approximate Distance-Comparison-Preserving Symmetric Encryption*. Cryptology ePrint Archive, Report 2021/1666. https://ia.cr/2021/1666. 2021.

[SR20] Congzheng Song and Ananth Raghunathan. "Information Leakage in Embedding Models". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. Association for Computing Machinery, 2020, pp. 377–390. DOI: 10.1145/3372297.3417270.

[BKR19] **Dmytro Bogatov**, George Kollios, and Leonid Reyzin. "A comparative evaluation of order-revealing encryption schemes and secure range-query protocols". In: *Proceedings of the VLDB Endowment* 12.8 (2019), pp. 933–947. DOI: 10.14778/3324301.3324309.

[KT19] Florian Kerschbaum and Anselme Tueno. "An Efficiently Searchable Encrypted Data Structure for Range Queries". In: *Computer Security – ESORICS 2019*. Springer International Publishing, 2019, pp. 344–364.

[Cas+18] David Cash, Feng-Hao Liu, Adam O'Neill, Mark Zhandry, and Cong Zhang. "Parameter-Hiding Order Revealing Encryption". In: *Advances in Cryptology – ASIACRYPT 2018*. Springer International Publishing, 2018, pp. 181–210.

[CLWW16] Nathan Chenette, Kevin Lewi, Stephen A. Weis, and David J. Wu. "Practical Order-Revealing Encryption with Limited Leakage". In: *Fast Software Encryption*. Springer Berlin Heidelberg, 2016, pp. 474–493.

[Dem+16] Ioannis Demertzis, Stavros Papadopoulos, Odysseas Papapetrou, Antonios Deligiannakis, and Minos Garofalakis. "Practical private range search revisited". In: *Proceedings of the 2016 International Conference on Management of Data*. 2016, pp. 185–198. DOI: 10.1145/2882903.2882911.

[LW16] Kevin Lewi and David J. Wu. "Order-Revealing Encryption: New Constructions, Applications, and Lower Bounds". In: ACM, 2016, pp. 1167–1178.

[RACY16] Daniel S. Roche, Daniel Apon, Seung Geol Choi, and Arkady Yerukhimovich. "POPE: Partial Order Preserving Encoding". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1131–1142.

[Sah+16] Cetin Sahin, Victor Zakhary, Amr El Abbadi, Huijia Lin, and Stefano Tessaro. "Taostore: Overcoming asynchronicity in oblivious data storage". In: *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2016, pp. 198–217.

[Xie+16] Dong Xie, Guanru Li, Bin Yao, Xuan Wei, Xiaokui Xiao, Yunjun Gao, and Minyi Guo. "Practical private shortest path computation based on oblivious storage". In: *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE. 2016, pp. 361–372.

[Ker15] Florian Kerschbaum. "Frequency-Hiding Order-Preserving Encryption". In: *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 656–667.

[Zhu+15] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Dec. 2015, pp. 19–27. DOI: `10.1109/ICCV.2015.11`.

[Ste+13] Emil Stefanov, Marten van Dijk, Elaine Shi, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. "Path ORAM: An Extremely Simple Oblivious RAM Protocol". In: *Proceedings of the 2013 ACM SIGSAC Conference on Computer Communications Security*. ACM, 2013, pp. 299–310.

[SSS12] Emil Stefanov, Elaine Shi, and Dawn Xiaodong Song. "Towards Practical Oblivious RAM". In: *Network and Distributed System Security Symposium (NDSS)*. 2012.

[SCSL11] Elaine Shi, T-H Hubert Chan, Emil Stefanov, and Mingfei Li. “Oblivious RAM with $O(\log^3 N)$ worst-case cost”. In: *International Conference on The Theory and Application of Cryptology and Information Security*. Springer. 2011, pp. 197–214. DOI: 10.1007/978-3-642-25385-0_11.

[BCLO09] Alexandra Boldyreva, Nathan Chenette, Younho Lee, and Adam O’Neill. “Order-Preserving Symmetric Encryption”. In: *Advances in Cryptology - EUROCRYPT 2009*. Springer Berlin Heidelberg, 2009, pp. 224–241.

[Dwo+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. “Our data, ourselves: Privacy via distributed noise generation”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2006, pp. 486–503. DOI: 10.1007/11761679_29.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. “Calibrating noise to sensitivity in private data analysis”. In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284. DOI: 10.1007/11681878_14.

[GO96] Oded Goldreich and Rafail Ostrovsky. "Software protection and simulation on oblivious RAMs". In: *Journal of the ACM (JACM)* 43.3 (1996), pp. 431–473. DOI: 10.1145/233551.233553.

[Gol87] Oded Goldreich. "Towards a theory of software protection and simulation by oblivious RAMs". In: *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987, pp. 182–194. DOI: 10.1145/28395.28416.

Appendix

OPE / ORE schemes

Scheme	Primitive usage		Ciphertext size, or state size	Leakage (in addition to inherent total order)
	Encryption	Comparison		
BCLO [BCLO09]	**n HG**	none	$2n$	**$\approx$ Top half of the bits**
CLWW [CLWW16]	n PRF	none	$2n$	**Most-significant differing bit**
Lewi-Wu [LW16]	**$2n/d$ PRP** $2\frac{n}{d}\left(2^d+1\right)$ PRF $\frac{n}{d}2^d$ Hash	$\frac{n}{2d}$ Hash	$\frac{n}{d}\left(\lambda+n+2^{d+1}\right)+\lambda$	Most-significant differing block
CLOZ [Cas+18]	n PRF n PPH 1 PRP	**n^2 PPH**	$n \cdot h$	Equality pattern of most-significant differing bit
FH-OPE [Ker15]	1 Traversal	3 Traversals	**$3 \cdot n \cdot N$**	Insertion order

Table 1: [BKR19, Table 1]. Primitive usage by OPE / ORE schemes. Ordered by security rank — most secure below. n is the input length in bits, d is a block size for Lewi-Wu [LW16] scheme, λ is a PRF output size, N is a total data size, **HG** is a hyper-geometric distribution sampler, **PPH** is a property-preserving hash with h-bit outputs built with bilinear maps and **bolded** are weak points of the schemes.

Back to ORE

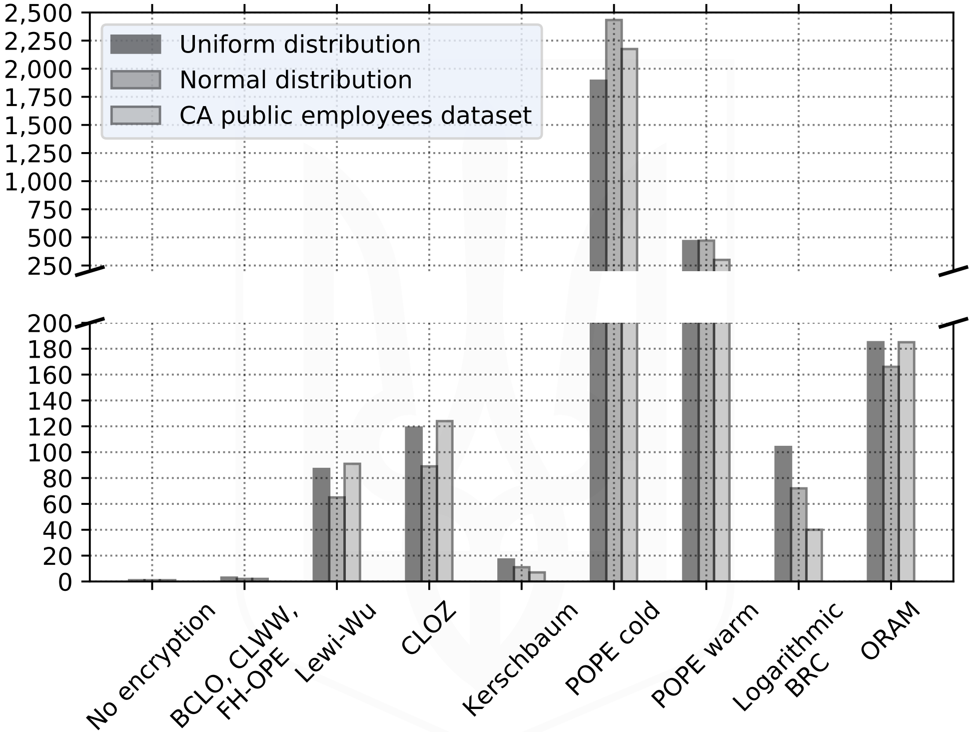
Range query protocols

Protocol	I/O requests		Leakage	Communication (result excluded)	
	Construction	Query		Construction	Query
B+ tree with ORE	$\log_B \frac{N}{B}$	$\log_B \frac{N}{B} + \frac{r}{B}$	**Same as ORE**	1	1
Kerschbaum [KT19]	$\boldsymbol{\frac{N}{B}}$	$\log_2 \frac{N}{B} + \frac{r}{B}$	**Total order**	$\log_2 N$	$\log_2 N$
POPE [RACY16] warm POPE [RACY16] cold	1	$\log_L \frac{N}{B} + \frac{r}{B}$ $\boldsymbol{{}^{N}/_{B}}$	**Partial order** Fully hiding	1	$\log_L N$ $\boldsymbol{N}$
Logarithmic-BRC [Dem+16]	—	$\boldsymbol{r}$	Same as SSE	—	$\log_2 N$
ORAM	$\log^2 \boldsymbol{\frac{N}{B}}$	$\log_2 \boldsymbol{\frac{N}{B}} \left(\log_B \boldsymbol{\frac{N}{B}} + \boldsymbol{\frac{r}{B}}\right)$	Fully hiding (access pattern)	$\log^2 \boldsymbol{\frac{N}{B}}$	$\log^2 \boldsymbol{\frac{N}{B}}$

Table 2: [BKR19, Tables 2]. Performance of the range query protocols. Ordered by security rank — most secure below. N is a total data size, B is an I/O page size, L is a POPE tree branching factor, r is the result size in records and **bolded** are weak points of the protocols.

Back to ORE

One of the experimental results



Query stage number of I/O requests

Back to ORE

Access pattern and ORAM

Access pattern is a sequence of memory accesses $\mathbf{y}$, where each access consists of the memory *location o*, read $\mathbf{r}$ or write $\mathbf{w}$ *operation* and the *data d* to be written.

Oblivious RAM (ORAM) is a mechanism that hides the accesses pattern. More formally, ORAM is a protocol between the client $\mathcal{C}$ (who accesses) and the server $\mathcal{S}$ (who stores), with a guarantee that the view of the server is indistinguishable for any two sequences of the same lengths.

$$|\mathbf{y}_1| = |\mathbf{y}_2|$$
$$\textsc{View}_{\mathcal{S}}(\mathbf{y}_1) \overset{c}{\approx} \textsc{View}_{\mathcal{S}}(\mathbf{y}_2)$$

ORAM protocol

1 :	**Client** $\mathcal{C}$		**Server** $\mathcal{S}$
2 :	$\mathbf{y} = (\mathbf{r}, i, \perp)\rvert_{i=1}^{5}$		
3 :	(client state)	$\xleftrightarrow{\text{ORAM}(\mathbf{y})}$	(server state)
4 :	$\{d_1, d_2, d_3, d_4, d_5\}$		

Square Root ORAM [Gol87], Hierarchical ORAM [GO96], Binary-Tree ORAM [SCSL11], Interleave Buffer Shuffle Square Root ORAM [Xie+16], TP-ORAM [SSS12], **PathORAM** [Ste+13] and TaORAM [Sah+16]. ORAM incurs at least logarithmic overhead in the number of stored records. [GO96]

Back to $\mathcal{E}$psolute

On impossibility of adaptive queries

Why is the query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ fixed?

- Suppose neighboring medical databases differ in one record with a rare diagnosis "Alzheimer's disease"
- A medical professional, who is **a user (and not an adversary)** queries the database
  - for that diagnosis first

```
SELECT name FROM patients WHERE condition = 'ALZ'
```

  - if there is a record, she queries the senior patients next

```
SELECT name FROM patients WHERE age >= 65
```

  - otherwise she queries the general population, resulting in many more records

```
SELECT name FROM patients
```

- Adversary can know the answer to the first query by observing result size of the second
- Efficient system cannot return nearly the same number of records in both cases, thus, the adversary can distinguish

Back to $\mathcal{E}$psolute

On impossibility of adaptive queries

Why is the query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ fixed?

- Suppose neighboring medical databases differ in one record with a rare diagnosis "Alzheimer's disease"
- A medical professional, who is **a user (and not an adversary)** queries the database
  - for that diagnosis first
    `SELECT name FROM patients WHERE condition = 'ALZ'`
  - if there is a record, she queries the senior patients next
    `SELECT name FROM patients WHERE age >= 65`
  - otherwise she queries the general population, resulting in many more records
    `SELECT name FROM patients`
- Adversary can know the answer to the first query by observing result size of the second
- Efficient system cannot return nearly the same number of records in both cases, thus, the adversary can distinguish

Back to $\mathcal{E}$psolute

On impossibility of adaptive queries

Why is the query sequence $q_1, \ldots, q_m \in \mathcal{Q}^m$ fixed?

- Suppose neighboring medical databases differ in one record with a rare diagnosis "Alzheimer's disease"
- A medical professional, who is **a user (and not an adversary)** queries the database
  - for that diagnosis first
    `SELECT name FROM patients WHERE condition = 'ALZ'`
  - if there is a record, she queries the senior patients next
    `SELECT name FROM patients WHERE age >= 65`
  - otherwise she queries the general population, resulting in many more records
    `SELECT name FROM patients`
- Adversary can know the answer to the first query by observing result size of the second
- Efficient system cannot return nearly the same number of records in both cases, thus, the adversary can distinguish

Back to $\mathcal{E}$psolute

Distance Comparison Preserving Encryption algorithms [FGHO21]

Algorithm 1 Distance Comparison Preserving Encryption, adapted from [FGHO21, Algorithm 2]

$\textsc{KeyGen}(1^\lambda, \mathbb{S})$

$s \leftarrow_\$ \mathbb{S}$

$\mathsf{k} \leftarrow_\$ \{0,1\}^\lambda$

return (s, k)

$\textsc{Enc}((s, \mathsf{k}), \vec{m})$

$n \leftarrow_\$ \{0,1\}^\lambda$

$\mathsf{coins}_n || \mathsf{coins}_u \leftarrow \mathsf{PRF}(\mathsf{k}, n)$

$\vec{n} \leftarrow_\$ \textsc{Normal}(0, I_d; \mathsf{coins}_n)$

$u \leftarrow_\$ \textsc{Uniform}(0, 1; \mathsf{coins}_u)$

$x \leftarrow \frac{s\beta}{4} \cdot \sqrt[d]{u}$

$\vec{\delta} \leftarrow \frac{\vec{n}}{\|\vec{n}\|} \cdot x$

$\vec{c} \leftarrow s \cdot \vec{m} + \vec{\delta}$

return $\vec{c}$

$\textsc{Dec}((s, \mathsf{k}), (\vec{c}, n))$

$\mathsf{coins}_n || \mathsf{coins}_u \leftarrow \mathsf{PRF}(\mathsf{k}, n)$

$\vec{n} \leftarrow_\$ \textsc{Normal}(0, I_d; \mathsf{coins}_n)$

$u \leftarrow_\$ \textsc{Uniform}(0, 1; \mathsf{coins}_u)$

$x \leftarrow \frac{s\beta}{4} \cdot \sqrt[d]{u}$

$\vec{\delta} \leftarrow \frac{\vec{n}}{\|\vec{n}\|} \cdot x$

$\vec{m} \leftarrow \frac{\vec{c} - \vec{\delta}}{s}$

return $\vec{m}$

Back to Private kNN