Oral Exam

Private and Secure Query Processing in Outsourced Databases

Property-Revealing Encryption, Oblivious Execution, Differential Privacy, $\mathcal{E}$psolute [73]

Dmytro Bogatov

dmytro@bu.edu

Built from *134854bf* on November 8, 2021

Boston University

Graduate School of Arts and Sciences

Department of Computer Science

Background

Motivation and overview

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

Motivation and overview

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

Motivation and overview

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

Motivation and overview

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

Motivation and overview

- With vast amounts of data, organizations choose to use cloud solutions
- These solutions need to be both efficient and secure
- Recent attacks on access pattern (AP) [19, 27, 30, 33, 40, 51, 48, 20, 55] and communication volume (CV) [40, 70, 55, 54, 63]
- Existing solutions may be insufficient:
  - protection against snapshot adversary does not account for AP and CV
    CryptDB [17], Arx [66], Seabed [43] and SisoSPIR [39]
  - enclaves like SGX are still uncommon and limited in memory
    Cipherbase [23], HardIDX [47], StealthDB [67], EnclaveDB [57], ObliDB [62], Opaque [49] and Oblix [56]
  - other solutions protect either from one of AP or CV, or use linear scan and full padding
    Cryptϵ [72], Shrinkwrap [50], SEAL [69] and PINED-RQ [58]
- $\mathcal{E}$psolute [73]: most secure and practical range- and point-query engine in the outsourced database model, that protects both AP and CV using Differential Privacy, while not relying on TEE, linear scan or full padding

Symmetric Encryption Scheme

Key generation $k \leftarrow_\$ \text{E.KeyGen}()$

Encrypt $c \leftarrow_\$ \text{E.Enc}(x, k)$

Decrypt $x \leftarrow \text{E.Dec}(c, k)$

For example, AES [5] in CBC mode + IV [4].

Order-revealing encryption scheme

Key generation $k \leftarrow_\$ \text{ORE.KeyGen}()$

Encrypt $c \leftarrow_\$ \text{ORE.Enc}(x, k)$

Decrypt $x \leftarrow \text{ORE.Dec}(c, k)$

Compare $c_1 \textbf{ op } c_2 \equiv x_1 \textbf{ op } x_2$

$\textbf{op} \in \{<, \leq, =, \geq, >\}$

For example, BCLO [13], CLWW [35], Lewi-Wu [41], CLOZ [52] and FH-OPE [32].

Cryptographic primitives

Symmetric Encryption Scheme

Key generation $k \leftarrow_{\$} \text{E.K}\textsc{ey}\text{G}\textsc{en}\,()$

Encrypt $c \leftarrow_{\$} \text{E.E}\textsc{nc}\,(x, k)$

Decrypt $x \leftarrow \text{E.D}\textsc{ec}\,(c, k)$

For example, AES [5] in CBC mode + IV [4].

Order-revealing encryption scheme

Key generation $k \leftarrow_{\$} \text{ORE.K}\textsc{ey}\text{G}\textsc{en}\,()$

Encrypt $c \leftarrow_{\$} \text{ORE.E}\textsc{nc}\,(x, k)$

Decrypt $x \leftarrow \text{ORE.D}\textsc{ec}\,(c, k)$

Compare $c_1 \textbf{ op } c_2 \equiv x_1 \textbf{ op } x_2$

$\textbf{op} \in \{<, \leq, =, \geq, >\}$

For example, BCLO [13], CLWW [35], Lewi-Wu [41], CLOZ [52] and FH-OPE [32].

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

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

For example: Square Root ORAM [1], Hierarchical ORAM [2], Binary-Tree ORAM [18], Interleave Buffer Shuffle Square Root ORAM [46], TP-ORAM [21], **Path-ORAM** [26] and TaORAM [45]. ORAM incurs at least logarithmic overhead in the number of stored records. [2]

k-anonymity [6]

Every tuple in the released table must be indistinguishably related to no fewer than k respondents (i.e., similar to at lest $k-1$ other tuples).

- only with respect to quasi-identifiers
- attacks using background knowledge and lack of diversity
- a property of a table, not a mechanism (other works with anonymization techniques exist)

ℓ-diversity [12]

A block is ℓ-diverse if it contains at least ℓ "well-represented" values for the sensitive attribute S. A table is ℓ-diverse if every block is ℓ-diverse.

Can choose definition of "well-represented". For example, in *entropy ℓ-diversity*, every block has at least ℓ distinct values for the sensitive attribute. In *recursive ℓ-diversity*, most common value does not appear too often, less common — not too infrequently.

k-anonymity [6]

Every tuple in the released table must be indistinguishably related to no fewer than k respondents (i.e., similar to at lest $k-1$ other tuples).

- only with respect to quasi-identifiers
- attacks using background knowledge and lack of diversity
- a property of a table, not a mechanism (other works with anonymization techniques exist)

ℓ-diversity [12]

A block is ℓ-diverse if it contains at least ℓ “well-represented” values for the sensitive attribute S. A table is ℓ-diverse if every block is ℓ-diverse.

Can choose definition of “well-represented”. For example, in *entropy ℓ-diversity*, every block has at least ℓ distinct values for the sensitive attribute. In *recursive ℓ-diversity*, most common value does not appear too often, less common — not too infrequently.

t-closeness [11]

A block exhibits t-closeness if the distance between the distributions of a sensitive attribute in this block and in the whole table is no more than a threshold t. A table exhibits t-closeness if every block does. The metric used is the Earth Mover's Distance [3].

Differential Privacy, adapted from [10, 9]

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,

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

- Laplace Perturbation Algorithm (LPA) [9, Theorem 1]
- Differentially Private Sanitizer
- Composition Theorem (disjoint and non-disjoint sets)

t-closeness [11]

A block exhibits t-closeness if the distance between the distributions of a sensitive attribute in this block and in the whole table is no more than a threshold t. A table exhibits t-closeness if every block does. The metric used is the Earth Mover's Distance [3].

Differential Privacy, adapted from [10, 9]

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,

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

- Laplace Perturbation Algorithm (LPA) [9, Theorem 1]
- Differentially Private Sanitizer
- Composition Theorem (disjoint and non-disjoint sets)

Encalves, SGX and ZeroTrace

Software Guard Extensions (SGX) [22, 24, 25, 31, 37]

Features:

- Set of new x86 instructions
- Virtual isolation within “enclaves”
- The entire non-enclave stack is untrusted
- Can swap/re-encrypt pages from RAM
- Application declares enclave and non-enclave parts
- Enclave should manipulate sensitive data, e.g., keys

Issues:

- Small $\approx$ 96 MB of “trusted” memory
- Enclave code is significantly slower
- No direct I/O or syscalls
- Leaks access pattern

ZeroTrace [59]

PathORAM [26] or CircuitORAM [34] in SGX, given that the enclave code leaks access pattern. Uses oblivious operations.

Model

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily “full” reconstruction
- Lots of attacks [33, 51, 48, 65]

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily “full” reconstruction
- Lots of attacks [33, 51, 48, 65]

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily "full" reconstruction
- Lots of attacks [33, 51, 48, 65]

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily “full” reconstruction
- Lots of attacks [33, 51, 48, 65]

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily “full” reconstruction
- Lots of attacks [33, 51, 48, 65]

Adversary steals the hard drive

- Cannot query fully encrypted blob
  cannot outsource key
- Download-decrypt-query is inefficient
- Relaxing from absolute (semantic) security
- Searchable symmetric encryption (SSE) [8]
- Fully-homomorphic encryption (FHE) [15]
- Functional Encryption [16]
- Property-preserving encryption (PPE) [7, 29]

Attacks

- Usually require auxillary knowledge
  e.g., distribution
- Not necessarily “full” reconstruction
- Lots of attacks [33, 51, 48, 65]

Persistent adversary / AP and CV leakage

Access Pattern

- Which query "touches" which records
- Applicable to all types of queries
- Usually mitigated with ORAM
- Attacks [19, 27, 30, 33, 40, 51, 48, 20, 55]

Communication Volume

- The size of the answer (in bytes or records)
- More often applicable to range queries
- Usually mitigated with padding / noise
- Attacks [40, 70, 55, 54, 63]

Can we put forth a definition that would imply protection against all these attacks?

Access Pattern

- Which query “touches” which records
- Applicable to all types of queries
- Usually mitigated with ORAM
- Attacks [19, 27, 30, 33, 40, 51, 48, 20, 55]

Communication Volume

- The size of the answer (in bytes or records)
- More often applicable to range queries
- Usually mitigated with padding / noise
- Attacks [40, 70, 55, 54, 63]

Can we put forth a definition that would imply protection against all these attacks?

Persistent adversary / AP and CV leakage

Access Pattern

- Which query “touches” which records
- Applicable to all types of queries
- Usually mitigated with ORAM
- Attacks [19, 27, 30, 33, 40, 51, 48, 20, 55]

Communication Volume

- The size of the answer (in bytes or records)
- More often applicable to range queries
- Usually mitigated with padding / noise
- Attacks [40, 70, 55, 54, 63]

Can we put forth a definition that would imply protection against all these attacks?

Definition (Computationally Differentially Private Outsourced Database System (CDP-ODB))

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
- $\text{negl}(\lambda)$ accounts for the computational (as opposed to theoretical) DP definition

Definition (Computationally Differentially Private Outsourced Database System (CDP-ODB))

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
- $\text{negl}(\lambda)$ accounts for the computational (as opposed to theoretical) DP definition

Work in the Area

Work in the Area

Property-Preserving Encryption

Dmytro Bogatov
Boston University

Survey of OPE/ORE schemes [60]

The problem

- Many different solutions
- Performance / security tradeoff
- Heterogeneous security definitions and leakage profiles
- **Performance of the schemes not well-understood**
  - Some were not even implemented
  - Prototype implementation at best
  - Not benchmarked against one another
  - Use different primitive implementations

Our solution

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

Survey of OPE/ORE schemes [60]

The problem

- Many different solutions
- Performance / security tradeoff
- Heterogeneous security definitions and leakage profiles
- **Performance of the schemes not well-understood**
  - Some were not even implemented
  - Prototype implementation at best
  - Not benchmarked against one another
  - Use different primitive implementations

Our solution

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

Survey of OPE/ORE schemes [60]

OPE/ORE schemes + leakage

- BCLO [14]
  ≈ top half of the bits
- CLWW [36]
  most-significant differing bit
- Lewi-Wu [42]
  most-significant differing block
- CLOZ [53]
  equality pattern of most-significant differing bit
- FH-OPE [32]
  insertion order

Range query protocols + leakage (on top of AP and CV)

- B^+ tree with ORE
  same as underlying ORE
- Kerschbaum [64]
  total order
- POPE [44]
  partial order
- Logarithmic-BRC [38]
  same as underlying SSE
- ORAM with B^+ tree
  fully hiding

CryptDB [17]

CryptDB design and contributions

- Regular SQL API for applications
- Proxy between app and server rewrites queries
- Encryption depending on operations
  - OPE for comparison
  - DET for equality
  - HE for aggregates
  - RND if value is never used
- Records encrypted in onion layers
- Column key derives from user password

Issues

- Once onion level is removed, security degradation is permanent
- Leakage
  - Order and histogram
  - Access pattern
  - Communication volume

Arx [66], PPQED [28] and SisoSPIR [39]

Arx [66]

- A proxy between app and MongoDB
- Uses only semantically secure encryption
- Innovative range index with garbled circuits
  has to "rebuild" circuits
- Equality index inspired by SSE
- Almost no leakage for snapshot adversary
- Requires schema and queries in advance
- **Leaks AP and CV**

PPQED [28]

- Securely evaluate DNF of predicates
- Two non-colluding servers, one has keys
- Uses garbled circuits or HE
- Slow, leaks CV and (apparently) AP

SisoSPIR [39]

- Three parties, at most one corrupted
- B+ tree stored in ORAM layer-by-layer
- Neither party sees the exact search path
- Claim protection against CV and AP

Arx [66], PPQED [28] and SisoSPIR [39]

Arx [66]

- A proxy between app and MongoDB
- Uses only semantically secure encryption
- Innovative range index with garbled circuits
  has to "rebuild" circuits
- Equality index inspired by SSE
- Almost no leakage for snapshot adversary
- Requires schema and queries in advance
- **Leaks AP and CV**

PPQED [28]

- Securely evaluate DNF of predicates
- Two non-colluding servers, one has keys
- Uses garbled circuits or HE
- Slow, leaks CV and (apparently) AP

SisoSPIR [39]

- Three parties, at most one corrupted
- B^+ tree stored in ORAM layer-by-layer
- Neither party sees the exact search path
- Claim protection against CV and AP

Arx [66], PPQED [28] and SisoSPIR [39]

Arx [66]

- A proxy between app and MongoDB
- Uses only semantically secure encryption
- Innovative range index with garbled circuits

  has to "rebuild" circuits
- Equality index inspired by SSE
- Almost no leakage for snapshot adversary
- Requires schema and queries in advance
- **Leaks AP and CV**

PPQED [28]

- Securely evaluate DNF of predicates
- Two non-colluding servers, one has keys
- Uses garbled circuits or HE
- Slow, leaks CV and (apparently) AP

SisoSPIR [39]

- Three parties, at most one corrupted
- B^+ tree stored in ORAM layer-by-layer
- Neither party sees the exact search path
- Claim protection against CV and AP

Work in the Area

Access Pattern and/or Communication Volume

Cryptϵ [72]

Cryptϵ [72] design and contributions

- Executes entire "DP programs"
  transformations followed by the measurement
- Non-colluding *analyst* and *crypto server*
- Crypto server
  - decrypts
  - keeps privacy budget ϵ
  - adds Laplacian noise
- Analyst processes transformations
  - project
  - cross product
  - filter
- Experiments cover 7 "heavy" programs

Issues

- Adversary can observe both servers
- Malicious server defense requires TEE
- Not clear about the privacy of an individual
- Cannot output the result of transformation, program must end with a measurement
  `COUNT` or `CDF` is OK, range query is not
- Very slow
  typical program runs 3.6 hours even without network

Shrinkwrap [50]

Shrinkwrap [50] design and contributions

- *Federated* SQL queries (m owners)
- Pad and obliviously sort in circuit model
  hides both AP and CV
- "Shrink" to DP-sized chunk
- Optimal privacy budget allocation
- Much faster than fully oblivious

Issues

- $\mathcal{O}(n \log n)$ for n fully padded
- Pad and obliviously sort in circuit model
  naïve and performance is subpar
- Cannot run for $m > 2$
  union-join across m owners is infeasible
- Takes hours per query on a local network

SEAL [69]

- **SE A**djustable **L**eakage to the bit-level
- SE is based on Logarithmic-SRC-i [38]
- Adjustable ORAM hides α bits of AP
  partition data into $\frac{n}{2^\alpha}$ ORAMs
- Adjustable padding hides x bits of CV
  pad to the closest power of x
- New query protocols use SEAL as black-box
- Faster than scan, very slow in practice
  even though no I/Os, only RAM

PINED-RQ [58]

- B+ tree with already noisy records
- May end up dropping real records
- Updates are limited and expensive
- Experimental evaluation is misleading

Foundations of Differentially Oblivious Algorithms [61]

- New definition, algorithms and bounds
- AP itself is a DP-protected statistics
- Weaker than full obliviousness

SEAL [69], PINED-RQ [58] and Foundations of Differentially Oblivious Algorithms [61]

SEAL [69]

- **SE A**djustable **L**eakage to the bit-level
- SE is based on Logarithmic-SRC-i [38]
- Adjustable ORAM hides α bits of AP
  partition data into $\frac{n}{2^\alpha}$ ORAMs
- Adjustable padding hides x bits of CV
  pad to the closest power of x
- New query protocols use SEAL as black-box
- Faster than scan, very slow in practice
  even though no I/Os, only RAM

PINED-RQ [58]

- B^+ tree with already noisy records
- May end up dropping real records
- Updates are limited and expensive
- Experimental evaluation is misleading

Foundations of Differentially Oblivious Algorithms [61]

- New definition, algorithms and bounds
- AP itself is a DP-protected statistics
- Weaker than full obliviousness

SEAL [69]

- **SE A**djustable **L**eakage to the bit-level
- SE is based on Logarithmic-SRC-i [38]
- Adjustable ORAM hides α bits of AP
  partition data into $\frac{n}{2^\alpha}$ ORAMs
- Adjustable padding hides x bits of CV
  pad to the closest power of x
- New query protocols use SEAL as black-box
- Faster than scan, very slow in practice
  even though no I/Os, only RAM

PINED-RQ [58]

- B^+ tree with already noisy records
- May end up dropping real records
- Updates are limited and expensive
- Experimental evaluation is misleading

Foundations of Differentially Oblivious Algorithms [61]

- New definition, algorithms and bounds
- AP itself is a DP-protected statistics
- Weaker than full obliviousness

Work in the Area

Trusted Execution Environment / Enclaves / SGX

Opaque [49] and ObliDB [62]

Opaque [49]

- Distributed analytics on top of Spark
  supports filter, join, aggregation
- Requires truly oblivious memory
  does not exist
- *Encryption* mode: security and integrity
  most of it "for free" with SGX
- *Oblivious* mode: hiding AP
  sort with bitonic, linear scan
- *Padding* mode: not filter out dummies
  impractical

ObliDB [62]

- Requires truly oblivious memory
- Choice of *flat* and *indexed* storage
- SELECT
  - *naïve*: ORAM over two tables
  - *small*: load to oblivious buffer
  - *large*: duplicates table, scans obliviously
  - *continuous*: assumes table is sorted
  - *hash*: put row into $H(i)$ position
- AGGREGATE: running value in enclave
- JOIN
  - *hash join*: put hashes in enclave
  - *sort-merge join*: sort chunk in SGX, merge chunks with bitonic, filter with linear scan

Opaque [49] and ObliDB [62]

Opaque [49]

- Distributed analytics on top of Spark
  supports filter, join, aggregation
- Requires truly oblivious memory
  does not exist
- *Encryption* mode: security and integrity
  most of it "for free" with SGX
- *Oblivious* mode: hiding AP
  sort with bitonic, linear scan
- *Padding* mode: not filter out dummies
  impractical

ObliDB [62]

- Requires truly oblivious memory
- Choice of *flat* and *indexed* storage
- SELECT
  - *naïve*: ORAM over two tables
  - *small*: load to oblivious buffer
  - *large*: duplicates table, scans obliviously
  - *continuous*: assumes table is sorted
  - *hash*: put row into $H(i)$ position
- AGGREGATE: running value in enclave
- JOIN
  - *hash join*: put hashes in enclave
  - *sort-merge join*: sort chunk in SGX, merge chunks with bitonic, filter with linear scan

Cipherbase [23]

- Pre-SGX era, FPGA "trusted machine"
- Primitive operators executed in TEE
- No experiments or analysis

StealthDB [67]

- SGX extension over PostgreSQL
- Bring components to SGX on-demand
- Implementation is great: loadable module

EnclaveDB [57]

- Mostly integrity and freshness guarantees
- No AP or CV protection, side-channels out of scope
- Assumes 192 GB of truly oblivious memory
- Ideas:
  - put entire mini-OS in enclave
  - compile queries to binaries
- No SGX in experiments, poor simulation

Hermetic [68]

- Set of primitives against side-channels
- Oblivious execution environment
  - always runs to completion
  - uses only cache lines
  - data-oblivious
  - no side-effects
- Assumes only software attacks
- Not published yet

Cipherbase [23]

- Pre-SGX era, FPGA "trusted machine"
- Primitive operators executed in TEE
- No experiments or analysis

StealthDB [67]

- SGX extension over PostgreSQL
- Bring components to SGX on-demand
- Implementation is great: loadable module

EnclaveDB [57]

- Mostly integrity and freshness guarantees
- No AP or CV protection, side-channels out of scope
- Assumes 192 GB of truly oblivious memory
- Ideas:
  - put entire mini-OS in enclave
  - compile queries to binaries
- No SGX in experiments, poor simulation

Hermetic [68]

- Set of primitives against side-channels
- Oblivious execution environment
  - always runs to completion
  - uses only cache lines
  - data-oblivious
  - no side-effects
- Assumes only software attacks
- Not published yet

Cipherbase [23]

- Pre-SGX era, FPGA "trusted machine"
- Primitive operators executed in TEE
- No experiments or analysis

StealthDB [67]

- SGX extension over PostgreSQL
- Bring components to SGX on-demand
- Implementation is great: loadable module

EnclaveDB [57]

- Mostly integrity and freshness guarantees
- No AP or CV protection, side-channels out of scope
- Assumes 192 GB of truly oblivious memory
- Ideas:
  - put entire mini-OS in enclave
  - compile queries to binaries
- No SGX in experiments, poor simulation

Hermetic [68]

- Set of primitives against side-channels
- Oblivious execution environment
  - always runs to completion
  - uses only cache lines
  - data-oblivious
  - no side-effects
- Assumes only software attacks
- Not published yet

Cipherbase [23]

- Pre-SGX era, FPGA “trusted machine”
- Primitive operators executed in TEE
- No experiments or analysis

StealthDB [67]

- SGX extension over PostgreSQL
- Bring components to SGX on-demand
- Implementation is great: loadable module

EnclaveDB [57]

- Mostly integrity and freshness guarantees
- No AP or CV protection, side-channels out of scope
- Assumes 192 GB of truly oblivious memory
- Ideas:
  - put entire mini-OS in enclave
  - compile queries to binaries
- No SGX in experiments, poor simulation

Hermetic [68]

- Set of primitives against side-channels
- Oblivious execution environment
  - always runs to completion
  - uses only cache lines
  - data-oblivious
  - no side-effects
- Assumes only software attacks
- Not published yet

Oblix [69]

- "Doubly-oblivious" data structures
- Doubly-oblivious sorted multimap
  r top values to hide CV
- Doubly-oblivious PathORAM
  somewhat better than ZeroTrace [59]
- Way to make "tree-like" structure oblivious
- Experiments only "estimate" performance of doubly-oblivious ORAM

HybrIDX [71]

- Range query index obfuscates CV and AP
- Does not consider AP leakage inside SGX
- CV is obfuscated with bucketization
- AP is obfuscated using cache

HardIDX [47]

- B^+ tree put directly in enclave
- AP and CV are not even considered

Oblix [69]

- "Doubly-oblivious" data structures
- Doubly-oblivious sorted multimap
  r top values to hide CV
- Doubly-oblivious PathORAM
  somewhat better than ZeroTrace [59]
- Way to make "tree-like" structure oblivious
- Experiments only "estimate" performance of doubly-oblivious ORAM

HybrIDX [71]

- Range query index obfuscates CV and AP
- Does not consider AP leakage inside SGX
- CV is obfuscated with bucketization
- AP is obfuscated using cache

HardIDX [47]

- B^+ tree put directly in enclave
- AP and CV are not even considered

Oblix [69], HybrIDX [71] and HardIDX [47]

Oblix [69]

- "Doubly-oblivious" data structures
- Doubly-oblivious sorted multimap

  r top values to hide CV
- Doubly-oblivious PathORAM

  somewhat better than ZeroTrace [59]
- Way to make "tree-like" structure oblivious
- Experiments only "estimate" performance of doubly-oblivious ORAM

HybrIDX [71]

- Range query index obfuscates CV and AP
- Does not consider AP leakage inside SGX
- CV is obfuscated with bucketization
- AP is obfuscated using cache

HardIDX [47]

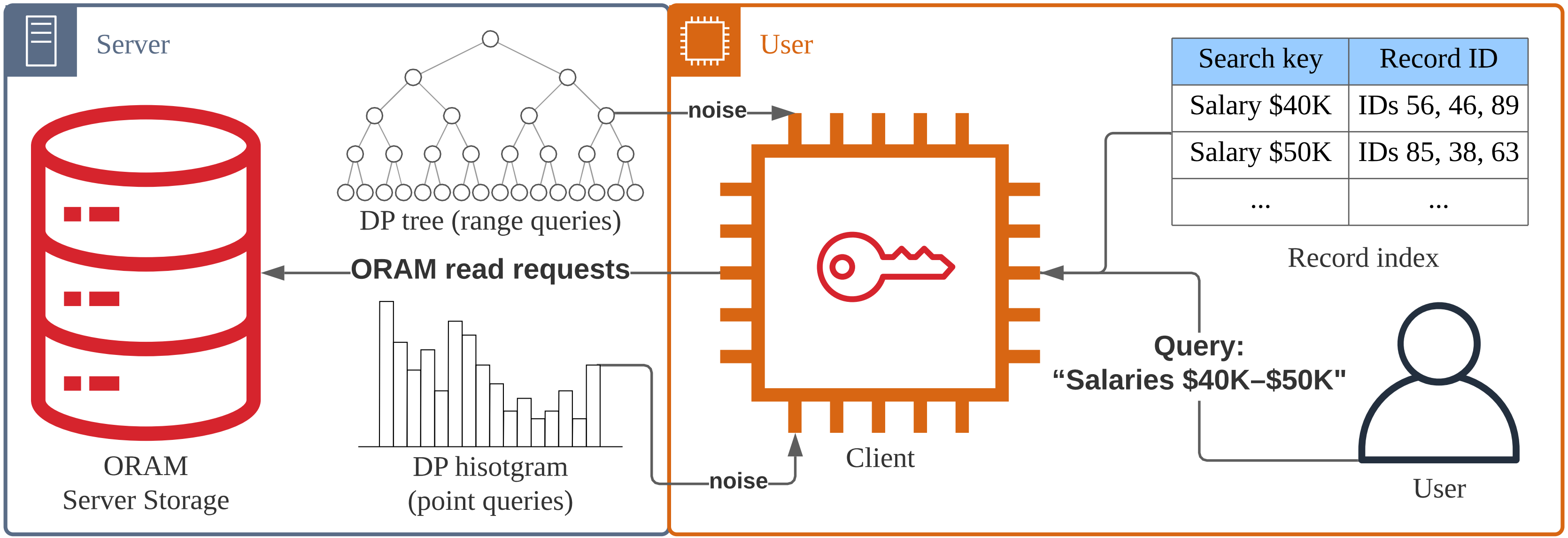
- B^+ tree put directly in enclave
- AP and CV are not even considered

EPSOLUTE

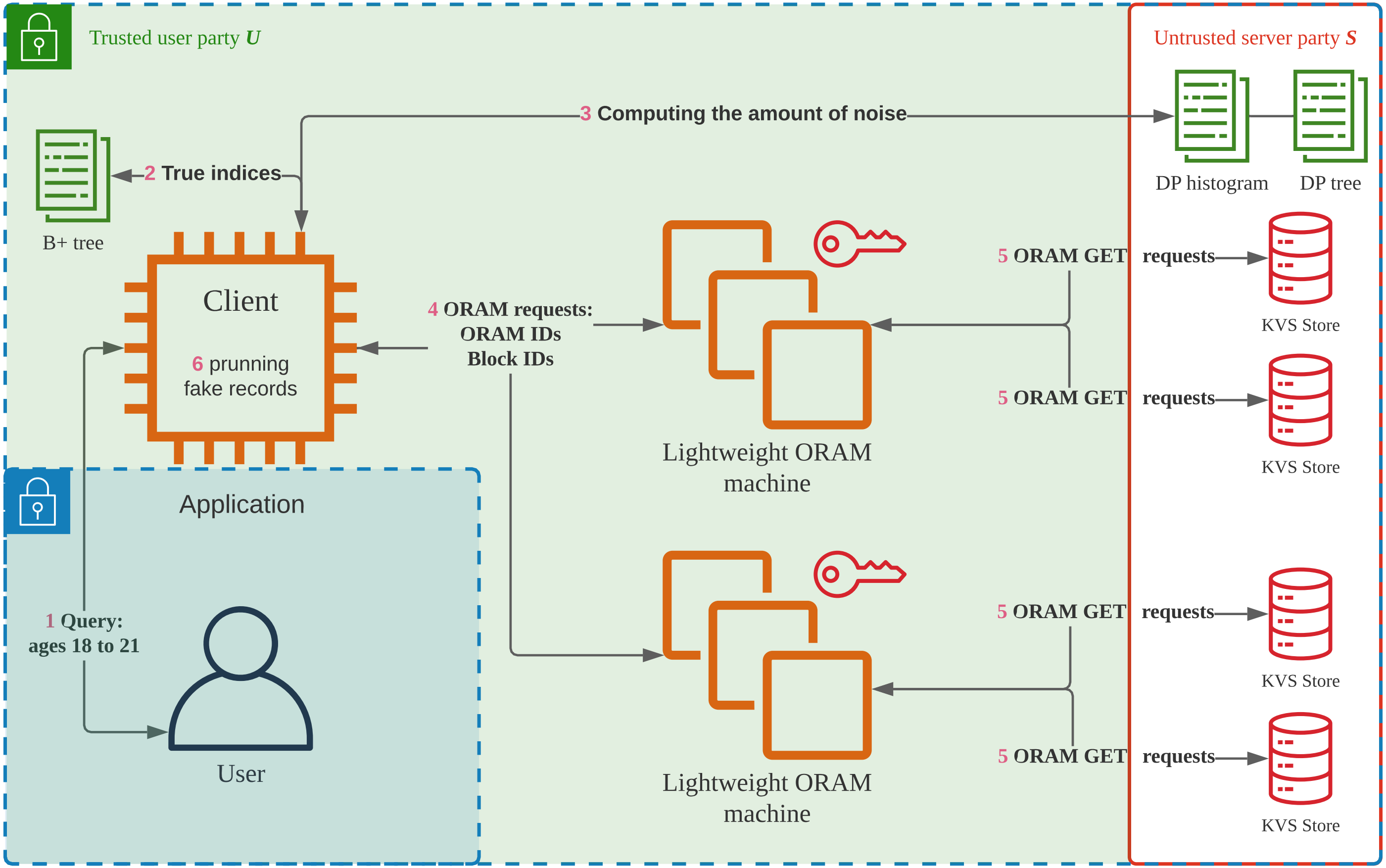
Dmytro Bogatov

Boston University

Single-Threaded $\mathcal{E}$psolute



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



Oral Exam

Private and Secure Query Processing in Outsourced Databases

Property-Revealing Encryption, Oblivious Execution, Differential Privacy, $\mathcal{E}$psolute [73]

Dmytro Bogatov

dmytro@bu.edu

Built from *134854bf* on November 8, 2021

Boston University

Graduate School of Arts and Sciences

Department of Computer Science

References

[1] 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.

[2] 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.

[3] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. "The earth mover's distance as a metric for image retrieval". In: *International journal of computer vision* 40.2 (2000), pp. 99–121.

[4] Morris Dworkin. *Recommendation for Block Cipher Modes of Operation: Methods and Techniques*. en. 2001. DOI: 10.6028/NIST.SP.800-38A.

[5] Morris Dworkin *et al.* *Advanced Encryption Standard (AES)*. en. 2001. DOI: 10.6028/NIST.FIPS.197.

[6] Latanya Sweeney. "k-anonymity: A model for protecting privacy". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10.05 (2002), pp. 557–570.

[7] Rakesh Agrawal *et al.* "Order preserving encryption for numeric data". In: *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*. 2004, pp. 563–574. DOI: 10.1145/1007568.1007632.

[8] Reza Curtmola *et al.* "Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions". In: *Proceedings of the 13th ACM Conference on Computer and Communications Security*. CCS '06. Association for Computing Machinery, 2006, pp. 79–88. DOI: 10.1145/1180405.1180417. URL: https://doi.org/10.1145/1180405.1180417.

[9] Cynthia Dwork *et al.* "Calibrating noise to sensitivity in private data analysis". In: *Theory of cryptography conference*. Springer. 2006, pp. 265–284. DOI: 10.1007/11681878_14.

[10] Cynthia Dwork *et al.* “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.

[11] Ninghui Li, Tiancheng Li, and Suresh Venkatasubramanian. “t-closeness: Privacy beyond k-anonymity and l-diversity”. In: *2007 IEEE 23rd International Conference on Data Engineering*. IEEE. 2007, pp. 106–115.

[12] Ashwin Machanavajjhala *et al.* “l-diversity: Privacy beyond k-anonymity”. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.1 (2007), 3–es.

[13] Alexandra Boldyreva *et al.* “Order-Preserving Symmetric Encryption”. In: *Advances in Cryptology - EUROCRYPT 2009*. Springer Berlin Heidelberg, 2009, pp. 224–241.

[14] Alexandra Boldyreva *et al.* “Order-Preserving Symmetric Encryption”. In: *Advances in Cryptology - EUROCRYPT 2009*. Springer Berlin Heidelberg, 2009, pp. 224–241.

[15] Craig Gentry. “Fully Homomorphic Encryption Using Ideal Lattices”. In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC ’09. Association for Computing Machinery, 2009, pp. 169–178. DOI: 10.1145/1536414.1536440.

[16] Dan Boneh, Amit Sahai, and Brent Waters. “Functional Encryption: Definitions and Challenges”. In: *Theory of Cryptography*. Springer Berlin Heidelberg, 2011, pp. 253–273.

[17] Raluca Ada Popa *et al.* “CryptDB: Protecting confidentiality with encrypted query processing”. In: *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*. 2011, pp. 85–100.

[18] Elaine Shi *et al.* “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.

[19] Bijit Hore *et al.* “Secure multidimensional range queries over outsourced data”. In: *VLDBJ* 21.3 (2012), pp. 333–358. DOI: 10.1007/s00778-011-0245-7.

References v

[20] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. “Access Pattern disclosure on Searchable Encryption: Ramification, Attack and Mitigation”. In: *19th Annual Network and Distributed System Security Symposium, NDSS 2012, San Diego, California, USA, February 5-8, 2012*. The Internet Society, 2012.

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

[22] Ittai Anati *et al.* “Innovative technology for CPU based attestation and sealing”. In: *Proceedings of the 2nd international workshop on hardware and architectural support for security and privacy*. Vol. 13. Citeseer. 2013, p. 7.

[23] Arvind Arasu *et al.* “Orthogonal Security With Cipherbase”. In: *6th Biennial Conference on Innovative Data Systems Research (CIDR’13)*. 2013.

[24] Matthew Hoekstra *et al.* “Using innovative instructions to create trustworthy software solutions.”. In: *HASP@ ISCA* 11.10.1145 (2013), pp. 2487726–2488370.

[25] Frank McKeen *et al.* "Innovative instructions and software model for isolated execution.". In: *Hasp@ isca* 10.1 (2013).

[26] Emil Stefanov *et al.* "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.

[27] Mohammad Saiful Islam, Mehmet Kuzu, and Murat Kantarcioglu. "Inference attack against encrypted range queries on outsourced databases". In: *Proceedings of the 4th ACM conference on Data and application security and privacy*. 2014, pp. 235–246. DOI: 10.1145/2557547.2557561.

[28] Bharath Kumar Samanthula, Wei Jiang, and Elisa Bertino. "Privacy-preserving complex query evaluation over semantically secure encrypted data". In: *European Symposium on Research in Computer Security*. Springer. 2014, pp. 400–418. DOI: 10.1007/978-3-319-11203-9_23.

[29] Dan Boneh *et al.* “Semantically secure order-revealing encryption: Multi-input functional encryption without obfuscation”. In: *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer. 2015, pp. 563–594. DOI: 10.1007/978-3-662-46803-6_19.

[30] David Cash *et al.* “Leakage-abuse attacks against searchable encryption”. In: *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*. 2015, pp. 668–679. DOI: 10.1145/2810103.2813700.

[31] Intel Corporation. *Intel® 64 and IA-32 Architectures Software Developer’s Manual*. Volume 3D: System Programming Guide, Part 4. 2015.

[32] 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.

[33] Muhammad Naveed, Seny Kamara, and Charles V Wright. "Inference attacks on property-preserving encrypted databases". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, pp. 644–655. DOI: 10.1145/2810103.2813651.

[34] Xiao Wang, Hubert Chan, and Elaine Shi. "Circuit ORAM: On Tightness of the Goldreich-Ostrovsky Lower Bound". In: *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. CCS '15. Association for Computing Machinery, 2015, pp. 850–861. DOI: 10.1145/2810103.2813634.

[35] Nathan Chenette *et al.* "Practical Order-Revealing Encryption with Limited Leakage". In: *Fast Software Encryption*. Springer Berlin Heidelberg, 2016, pp. 474–493.

[36] Nathan Chenette *et al.* "Practical Order-Revealing Encryption with Limited Leakage". In: *Fast Software Encryption*. Springer Berlin Heidelberg, 2016, pp. 474–493.

[37] Victor Costan and Srinivas Devadas. "Intel sgx explained.". In: *IACR Cryptol. ePrint Arch.* 2016.86 (2016), pp. 1–118.

[38] Ioannis Demertzis *et al.* “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.

[39] Yuval Ishai *et al.* “Private large-scale databases with distributed searchable symmetric encryption”. In: *Cryptographers’ Track at the RSA Conference*. Springer. 2016, pp. 90–107. DOI: 10.1007/978-3-319-29485-8_6.

[40] Georgios Kellaris *et al.* “Generic attacks on secure outsourced databases”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 1329–1340. DOI: 10.1145/2976749.2978386.

[41] Kevin Lewi and David J Wu. “Order-revealing encryption: New constructions, applications, and lower bounds”. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. 2016, pp. 1167–1178. DOI: 10.1145/2976749.2978376.

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

[43] Antonis Papadimitriou *et al.* "Big Data Analytics over Encrypted Datasets with Seabed". In: *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. USENIX Association, 2016, pp. 587–602.

[44] Daniel S. Roche *et al.* "POPE: Partial Order Preserving Encoding". In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016, pp. 1131–1142.

[45] Cetin Sahin *et al.* "Taostore: Overcoming asynchronicity in oblivious data storage". In: *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2016, pp. 198–217.

[46] Dong Xie *et al.* "Practical private shortest path computation based on oblivious storage". In: *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE. 2016, pp. 361–372.

[47] Benny Fuhry *et al.* "HardIDX: Practical and secure index with SGX". In: *IFIP Annual Conference on Data and Applications Security and Privacy*. Springer. 2017, pp. 386–408. DOI: 10.1007/978-3-319-61176-1_22.

[48] Paul Grubbs, Thomas Ristenpart, and Vitaly Shmatikov. "Why Your Encrypted Database Is Not Secure". In: *Proceedings of the 16th Workshop on Hot Topics in Operating Systems*. ACM, 2017, pp. 162–168. DOI: 10.1145/3102980.3103007.

[49] Wenting Zheng *et al.* "Opaque: An oblivious and encrypted distributed analytics platform". In: *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI '17)*. 2017, pp. 283–298.

[50] Johes Bater *et al.* "Shrinkwrap: efficient sql query processing in differentially private data federations". In: *Proceedings of the VLDB Endowment* 12.3 (2018), pp. 307–320. DOI: 10.14778/3291264.3291274.

[51] Vincent Bindschaedler *et al.* "The Tao of Inference in Privacy-Protected Databases". In: *PVLDB* 11.11 (2018), pp. 1715–1728. DOI: 10.14778/3236187.3236217.

[52] David Cash *et al.* "Parameter-Hiding Order Revealing Encryption". In: *Advances in Cryptology – ASIACRYPT 2018*. 2018, pp. 181–210.

[53] David Cash *et al.* “Parameter-Hiding Order Revealing Encryption”. In: *Advances in Cryptology – ASIACRYPT 2018*. Springer International Publishing, 2018, pp. 181–210.

[54] Paul Grubbs *et al.* “Pump up the volume: Practical database reconstruction from volume leakage on range queries”. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. 2018, pp. 315–331. DOI: 10.1145/3243734.3243864.

[55] Marie-Sarah Lacharité, Brice Minaud, and Kenneth G Paterson. “Improved reconstruction attacks on encrypted data using range query leakage”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 297–314. DOI: 10.1109/SP.2018.00002.

[56] Pratyush Mishra *et al.* “Oblix: An efficient oblivious search index”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 279–296. DOI: 10.1109/SP.2018.00045.

[57] Christian Priebe, Kapil Vaswani, and Manuel Costa. “EnclaveDB: A secure database using SGX”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2018, pp. 264–278. DOI: 10.1109/SP.2018.00025.

[58] Cetin Sahin *et al.* “A Differentially Private Index for Range Query Processing in Clouds”. In: *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. 2018, pp. 857–868. DOI: 10.1109/ICDE.2018.00082.

[59] Sajin Sasy, Sergey Gorbunov, and Christopher Fletcher. “ZeroTrace : Oblivious Memory Primitives from Intel SGX”. In: Jan. 2018. DOI: 10.14722/ndss.2018.23243.

[60] 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.

[61] T-H. Hubert Chan *et al.* “Foundations of Differentially Oblivious Algorithms”. In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA ’19. San Diego, California: Society for Industrial and Applied Mathematics, 2019, pp. 2448–2467.

[62] Saba Eskandarian and Matei Zaharia. “ObliDB: Oblivious query processing for secure databases”. In: *PVLDB* 13.2 (2019), pp. 169–183. DOI: 10.14778/3364324.3364331.

[63] Zichen Gui, Oliver Johnson, and Bogdan Warinschi. “Encrypted databases: New volume attacks against range queries”. In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 361–378. DOI: 10.1145/3319535.3363210.

[64] 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.

[65] Evgenios M. Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. “Data Recovery on Encrypted Databases with k-Nearest Neighbor Query Leakage”. In: *2019 IEEE Symposium on Security and Privacy (SP)*. 2019, pp. 1033–1050. DOI: 10.1109/SP.2019.00015.

[66] Rishabh Poddar, Tobias Boelter, and Raluca Ada Popa. “Arx: an encrypted database using semantically secure encryption”. In: *Proceedings of the VLDB Endowment* 12.11 (2019), pp. 1664–1678. DOI: 10.14778/3342263.3342641.

[67] Dhinakaran Vinayagamurthy, Alexey Gribov, and Sergey Gorbunov. “StealthDB: a scalable encrypted database with full SQL query support”. In: *Proceedings on Privacy Enhancing Technologies* 2019.3 (2019), pp. 370–388. DOI: 10.2478/popets-2019-0052.

[68] Min Xu *et al.* “Hermetic: Privacy-preserving distributed analytics without (most) side channels”. In: (2019).

[69] Ioannis Demertzis *et al.* “SEAL: Attack Mitigation for Encrypted Databases via Adjustable Leakage”. In: *29th USENIX Security Symposium (USENIX Security 20)*. USENIX Association, 2020, pp. 2433–2450. ISBN: 978-1-939133-17-5. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/demertzis.

[70] Evgenios M Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. “The state of the uniform: attacks on encrypted databases beyond the uniform query distribution”. In: *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE. 2020, pp. 1223–1240. DOI: 10.1109/SP40000.2020.00029.

[71] Kui Ren *et al.* "HybrIDX: New Hybrid Index for Volume-hiding Range Queries in Data Outsourcing Services". In: *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*. 2020, pp. 23–33. DOI: 10.1109/ICDCS47774.2020.00014.

[72] Amrita Roy Chowdhury *et al.* "Cryptϵ: Crypto-assisted differential privacy on untrusted servers". In: *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 2020, pp. 603–619. DOI: 10.1145/3318464.3380596.

[73] Dmytro Bogatov *et al.* "$\mathcal{E}$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.