ARMOR: Secure Querying of Massive Scientific Datasets

Hoda Maleki*, Ben Fuller€, Gagan Agrawal‡

*Augusta University, € University of Connecticut, ‡ University of Georgia

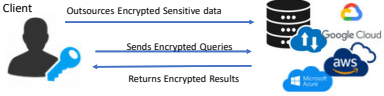
Motivation and Background

- **Scientific Data Examples**
  - Genomics
  - Astronomy
  - Climate Science
  - User locations
  - …
- **Data Outsourcing as a solution for data owners**

Companies using cloud 

- **Privacy Issues of Sensitive Data Outsourcing: Outsourcing data to third-party servers requires trusting them not to inspect or leak it, a guarantee they often cannot provide.**

Searchable Encryption



Searchable Encryption: searching on encrypted data without leaking the database contents and the query to the server

Access Pattern Attack Example

- Server sees encrypted block B accessed → B contains cancer records
- Result size → number of cancer patients

Oblivious RAM(ORAM): Hides Access patterns

A compiler that Re-randomize data locations after every operation

Organizing Records for Retrieval in Multi-Dimensional Range Searchable Encryption[1]

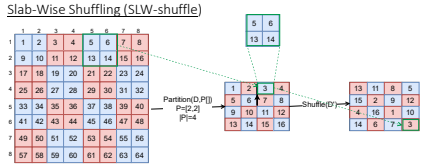
In this research we Focus on **Multi-dimensional Range queries** and **Security and efficiency of the record retrieval.**

Data Partitioning groups tuples into parts: requesting any tuple retrieves its whole part, hiding which tuple was of interest. We compare two strategies.

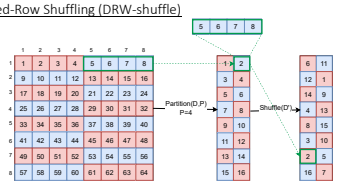
Query Geometry: We test four query shapes (isotropic, bisected, gradual, outlier) and measure how shape affects retrieval cost and leakage.

Secure Data Retrieval Methods to Hide the Access Pattern:
Shuffling randomizes record positions during setup or retrieval.
ORAM write-backs and shuffling happens with every query.

Slab-Wise Shuffling (SLW-shuffle)



Divided-Row Shuffling (DRW-shuffle)



Security Evaluation: Leakage Metrics

Size	Query Type	Shuffling Technique	Parts Avg	Parts STD	Parts H
[32]⁴	**Isotropic**	RCW-shuffle	230	71	1.7
	Avg = 150000	DRW-shuffle	82	68	5.5
	STD = 207000	SLW-shuffle	87	92	3.5
	Bisected	RCW-shuffle	250	38	1.5
	Avg = 120000	DRW-shuffle	85	69	6.7
	STD = 180000	SLW-shuffle	74	74	4.2
	Gradual	RCW-shuffle	250	33	.76
	Avg = 40900	DRW-shuffle	62	48	5
	STD = 36000	SLW-shuffle	42	30	3.6
	Outlier Min	RCW-shuffle	260	0	0
	Avg = 61000	DRW-shuffle	19	8.8	2.6
	STD = 28000	SLW-shuffle	74	23	.6
	Outlier Max	RCW-shuffle	171	91	4.9
	Avg = 730	DRW-shuffle	40	14	2.9
	STD = 710	SLW-shuffle	6.3	4.3	1.3

H: ORAM (Volume Leakage) Lower values : stronger security

Avg : Shuffling (Access Pattern Leakage) Lower Values: stronger security/better Efficiency

Finding: SLW-shuffle is more secure and efficient than DRW for nearly all query shapes. DRW only wins when the query is narrow on the indexed dimension and wide on the others (outlier-min).

SPANNS: Scalable Privacy-Preserving Approximate Nearest Neighbor Search

K-ANN Search: Returns the k nearest points to a query in a high-dimensional vector database. Powers RAG in LLMs, recommenders, ML retrieval.

HNSW Graph[7] is the fastest ANN algorithm at high accuracy (recall@k): a multi-layer graph traversed by greedy search from an entry point. Search complexity is poly-logarithmic.

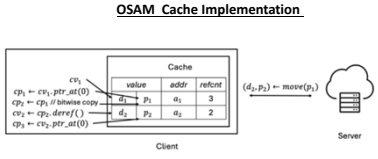
Goal: Oblivious K-ANN search that protects data, queries, results, and access patterns against an untrusted server. Prior approaches leak information, sacrifice accuracy, or fail to scale [8]. We present an oblivious HNSW-based K-ANN algorithm with high recall and scalability to larger dataset sizes for high-dimensional vector databases.

Optimization: 1-D splay tree with dummy nodes handles high in-degree HNSW nodes.

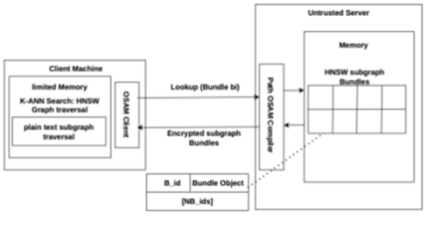
Oblivious Single Access Machine(OSAM)[9]: a weakening of ORAM that improves efficiency for pointer-chasing programs, including graph-based data structures.

Implementation: HNSW partitioned into subgraphs, each stored in one OSAM block; blocks maintain pointers to neighbors.

OSAM Cache Implementation



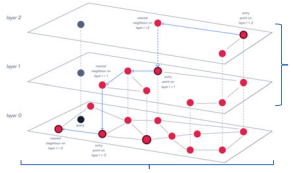
HNSW Oblivious Encrypted Search



HNSW Plain Text Search

Two phase Search : Zoom-in and Zoom-out

Zoom-in:
1- greedy search(BFS) in each layer to find the closest neighbor (efsearch=1).
2- Start search starting from the found closest node in the next layer and continue until the l=1.



Zoom-Out: greedy search with efsearch > 1

SPARQ: Scalable Privacy-Preserving Aggregate Range Queries[2]

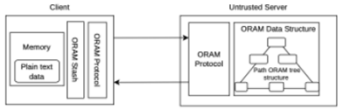
SPARQ enables aggregate range queries (SUM, AVG, COUNT, MIN, MAX, STDDEV) on encrypted multi-dimensional data without revealing the query, the data, or access pattern. Prior approaches require storage proportional to the domain size (impractical for large-scale sparse data) or use FHE (computationally expensive)*. We use oblivious multi-dimensional segment trees: storage scales with distinct values per dimension, not the domain.*

Result: 10× to 10^{10}× less storage and under 1.2 s server-side latency on a 32-thread machine for our largest 3D dataset.

ORAM-based Architecture

Precompute and store in the ORAM Storage Format
- Multidimensional Segment tree
- Each node of the tree→ *(node_id, value, meta)* in ORAM

Calculate the aggregation over a range by
- Traversing the tree by retrieving data from the ORAM
- Interactive algorithm

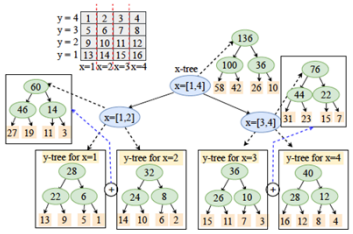


Storage Efficiency

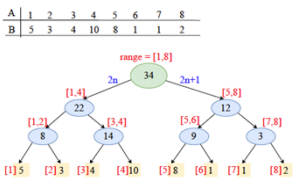
d	dataset	ST Max Act.	Key Value Based [3] SUM	[3] Min	[5] Quad	[5] Lin	[5] R. Cover	[4] Bucket.	FHE [6]C&R
1	books	4.2	4.2	5.9	8.1	7.8	5.9	5.9	4.0	3.9
	gowalla	7.0	7.0	7.7	9.0	13.4	7.0	7.9	7.4	6.7
	Spitz	6.2	4.6	6.0	4.7	6.5	6.0	6.1	5.5	4.0
	cali	6.6	5.5	6.0	7.7	8.5	6.0	6.6	6.2	5.2
2	gowalla	7.2	5.7	10.8	13.0	10.0	9.8	9.8	9.6	5.7
	gowalla	8.1	6.1	11.0	13.2	11.0	10.9	10.9	9.7	6.0
	synthetic	7.2	6.8	6.6	8.4	12.0	6.7	8.4	8.1	7.1
	synthetic	7.2	6.9	12.0	14.3	12.3	12.0	12.0	9.7	7.1
	synthetic	6.6	6.6	12.0	14.3	12.3	12.0	12.0	9.8	7.1
3	nh	6.3	5.2	5.4	7.3	7.2	5.4	6.9	6.0	5.5
	gowalla	6.4	5.7	6.0	8.0	8.7	6.0	7.8	7.4	6.4
	synthetic	7.2	7.1	18.0	21.4	18.0	18.0	18.0	13.5	8.3
	synthetic	8.1	7.6	18.0	21.4	21.0	21.0	21.0	13.3	8.3

Table 1. $\log_{10}$ of stored items. For Map and tree based solutions we list the number of items in the M. For the EMT approach which relies on an RM we list the number of stored items when padded to the maximum number of items associated with any keyword. ST.Max, is the size of segment tree size based on $\Pi_{i=1}^{d}(2n_i - 1)$. ST.Act, is the size of the segment tree which is stored in OMAP in practice per dataset based on the experiments.

2-d Segment tree



1-d Segment tree



Cryptography Efficiency

dataset	Oblivious Segment Tree Setup	Rd.	Search	Par.	Size	BW	Demertzis Prefix Array Setup	Search	Size	BW
Books 1D	15	15	0.05	0.01	17.0MB	171.0KB	830	.0018	1.5GB	190KB
gowalla 1D	14819	25	2.1	0.02	17.8GB	507.0KB	64K	0.0023	117GB	258KB
Spitz 2D	49	23	0.55	0.02	69MB	3.4MB	1.2K	0.0018	2.2GB	408KB
cali 2D	413	23	0.73	0.02	560MB	4.2MB	1.2K	0.0018	2.1GB	408KB
gowalla 2D	756	26	1.1	0.024	1.1GB	5.7MB	8M	0.0030	15TB	1.1MB
gowalla 2D	2220	29	1.7	0.029	2.2GB	8.1MB	104M	0.0033	190TB	1.4MB
synthetic 2D-2048-d	9811	25	2.4	0.03	8.9GB	5.7MB	4.9K	0.0020	9.1GB	497KB
synthetic 2D-2048-sp	15684	25	3.2	0.027	18GB	5.7MB	1.2B	0.0036	2.2PB	1.6MB
synthetic 2D-1024	6127	23	1.7	0.021	4.4GB	4.3MB	1.2B	0.0036	2.2PB	1.7MB
nh 3D	280	22	2.24	0.022	278MB	17MB	308	0.0016	569MB	665KB
gowalla-3D	706	24	3.45	0.027	1.1GB	18MB	1.2K	0.0018	2.2GB	816KB
synthetic 3D-128	23437	25	8.37	0.050	18GB	31MB	10^{13}	0.0054	2.6ZB	7.2MB
synthetic 3D-256	97118	28	21.02	0.046	71GB	54MB	10^{15}	0.0054	2.1ZB	10.0MB

Table 2. Measuring the ORAM performance over 1000 sample aggregation queries. Rd. is the average access depth plus one. Par. is the sum of the (average of) the maximum ORAM retrieval at each depth. Setup, search, and Par. are setup time, search time, and parallel search time in seconds.

Coordinating Structure of Shuffling and Indexing In Encrypted Range Queries

Approach. Coordinate index and shuffle structures in encrypted multi-dimensional range search. **Result.** Reduced retrieval overhead and lower leakage to the untrusted server.

Core Idea
- Encrypted range queries use range trees or quadtrees as indexes.
- Independent shuffling causes excess accesses and leaks query geometry.
- **Like-structured shuffling:** align record partitions to the index's geometry.

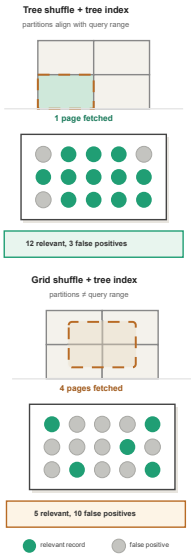
Cache Saving Write-Back We propose write-back schemes which prioritizes keeping likely-reused data present, without leaking measurably more to the server.

System-Wide Leakage
- **Standard model:** query tokens, response sizes.
- **Our model:** full retrieval trace — fetched pages, write-back timing, shared record overlap.

Policies safe in isolation can leak under the full trace.

We evaluate how index structure, query geometry, and write-back policies affect storage, query cost, communication, and four leakage channels: retrieval patterns, page co-occurrence, volume, and timing.

Main Takeaway Aligning index and shuffle structures cuts redundant retrieval, lowers storage pressure, and weakens leakage. It offers a practical middle ground: much stronger security than naive shuffling, but with far less overhead than full ORAM.



References

[1] M. Heidaripour, L. Kian, M. Rezapour, M. Holcomb, B. Fuller, G. Agrawal, and H. Maleki. Organizing records for retrieval in multi-dimensional range searchable encryption. In Proceedings of the 21st International Conference on Security and Cryptography- Volume 1: SECRYPT, pages 459–466. INSTICC, SciTePress, 2024.
[2] M. Heidaripour, M. Rezapour, B. Fuller, H. Maleki, and G. Agrawal, "SPARQ: Scalable privacy-preserving aggregate range queries," Cryptology ePrint Archive, 2026. [Online]. Available: https://eprint.iacr.org/2026/744.
[3] I. Demertzis, S. Papadopoulos, O. Papapetrou, A. Deligiannakis, M. Garofalakis, and C. Papamanthou. Practical private range search in depth. ACM Trans. Database Syst., 43(1), mar 2018.
[4] Z. Espiritu, E. A. Markatou, and R. Tamassia. Time-and space-efficient aggregate range queries over encrypted databases. PoPETS, 2022(4), 2022.
[5] F. Falzon, E. A. Markatou, Z. Espiritu, and R. Tamassia. Range search over encrypted multi-attribute data. Proceedings of the VLDB Endowment, 16(4), 2022.
[6] E. Kushnir, G. Moshkowich, and H. Shaul. Secure range-searching using copy-and-recurse. Proceedings on Privacy Enhancing Technologies, 3:626–644, 2024.
[7] Y. A. Malkov and D. A. Yashunin, "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 4, pp. 824–836, Apr. 2020, doi: 10.1109/TPAMI.2018.2889473.
[8] Zhu, J., Patel, L., Zaharia, M., & Popa, R. A. (2025, July). Compass: encrypted semantic search with high accuracy. In Proceedings of the 19th USENIX Conference on Operating Systems Design and Implementation (pp. 915-938).
[9] Appan, A., Heath, D. and Ren, L., 2024, December. Oblivious single access machines-A new model for oblivious computation. Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security (pp. 3080-3094).
[10] S. Pia, A. Appan, M. Rezapour, A. Shukla, N. Date, B. Fuller, L. Ren, and D. Heath. Oblivious single access machines are concretely efficient. Cryptology ePrint Archive, Paper 2026/451, 2026.