Poster: Oblivious k-Nearest Neighbors for Secure Map Applications

Jamie W. Lee
United States Naval Academy
jlee2039@gmail.com

Adam J. Aviv
George Washington University
aaviv@gwu.edu

Travis Mayberry
United States Naval Academy
mayberry@usna.edu

Daniel S. Roche
United States Naval Academy
roche@usna.edu

***Abstract*—In order to mitigate access pattern leakage and implement security in map applications, we have developed a novel network data structure, the ORAM-backed Hilbert B-tree. This data structure combines existing features such as the B-tree data structure, k-Nearest Neighbor (k-NN) search algorithms, Hilbert Curves, and Oblivious Random Access Memory (ORAM) algorithms to support oblivious dynamic data storage and 2D k-NN search queries using $O(log^2 n)$ bandwidth, functions that have not yet been conceived for such applications. This provides a significant improvement in security for map applications without compromising performance, preventing sensitive information such as the user's physical location from being leaked.**

I. Introduction

Cloud storage enables users to access and store large amounts of data on servers anytime and anywhere at little to no cost. Map applications are specific examples of cloud storage servers that allow users to query for nearby points of interest. Despite the many benefits of cloud storage, users are susceptible to data leakage: not only can users access the data they store, but so can the cloud itself. The cloud is *honest but curious*; although it follows established protocols honestly, it will attempt to gather as much information as it can. A naive approach would be to simply encrypt the data; however, this would only prevent adversaries from accessing the physical information contained directly in the files. Other forms of data leakage still exist in forms of metadata. With regards to map applications specifically, users may leak sensitive information through the access patterns made in search queries.

A more effective approach for enhancing security would be to encrypt both the access patterns of the file as well as its physical data. Users of cloud storage would want the server to be unaware of both the file contents and the access patterns they are making; in other words, the server should be *oblivious* to these details. For the past three decades, significant contributions in research have been made towards the development of oblivious data algorithms that accomplish these tasks. These algorithms, more specifically known as Oblivious Random Access Memory (ORAM) algorithms, obfuscate the user's access pattern to the queried file by randomizing locations of data in memory so that no two access patterns can be correlated. The security definition for ORAMs is as follows: the ORAM is secure if for any two sequences of logical accesses with the same number of operations, the transcripts for their physical accesses are indistinguishable [1]. By obfuscating the actual location of the stored data, servers cannot determine which files are actually being read or written to, and thus the servers are *oblivious* to the file location.

Previous work has been conducted to develop oblivious data structures using pointer-based and locality-based techniques [2]. Additionally, privacy-preserving k-NN searches have been explored through development of kd-tree based data structures to achieve data privacy in cloud storage [3] and Paillier public-key cryptosystems for location and query privacy [4]. Unlike previous research, however, our solution mitigates access pattern leakage in map applications at comparable accuracy and runtime. We propose a novel remotely-stored data structure network, the **ORAM-backed Hilbert B-tree**, which allows a user to execute k-Nearest Neighbor (k-NN) search queries in a cloud-based map application while mitigating access pattern leakage. Specifically, the implementation process provides the following features:

1) Development of a new oblivious data structure to support 2D search queries using $O(log^2 n)$ bandwidth,
2) Implementation with orthogonal Hilbert Space-Filling Curves and B-trees stored in an ORAM,
3) Oblivious k-NN search queries for 2D map applications in cloud services,
4) Support of a *dynamic* database in which data points can be added and/or removed obliviously.

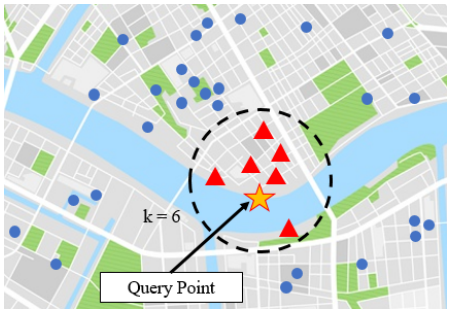

Fig. 1. A sample k-NN search query with k = 6 made in a map application.

Network and Distributed Systems Security (NDSS) Symposium 2020
23-26 February 2020, San Diego, CA, USA
ISBN 1-891562-61-4
https://dx.doi.org/10.14722/ndss.2020.23xxx
www.ndss-symposium.org

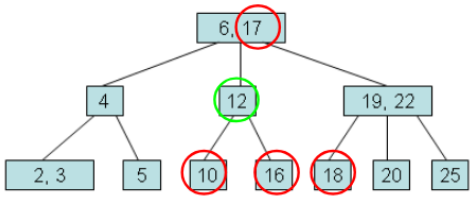

Fig. 2. A sample k-NN search query in a B-tree for ℓ = 12 and k = 4. The call to `search()` should return the list {10, 16, 17, 18}, as circled in red.

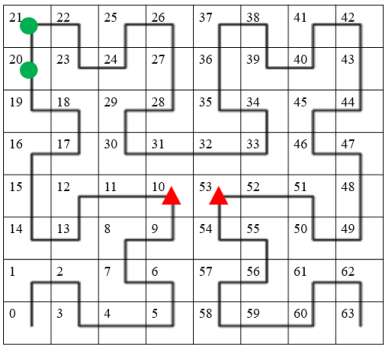

Fig. 3. Although Hilbert Curves preserve locality, there are points close together in 2D space but far apart in Hilbert space, as indicated by the difference in curve indexes of the two pairs of points.

II. Design and Implementation

We combine existing algorithms and data structures such as the B-tree data structure, k-NN search algorithms, Hilbert Curves, and ORAM algorithms to create this novel data structure. The implementation process and testing of this novel data structure involves several steps.

A. B-tree with k-NN

First, we implement the B-tree data structure with a new `search()` method that allows the user to input range k and query a key, returning the k closest values to the key even if the exact key does not exist in the structure.

B. Hilbert B-tree

To create a correlation between the 2D coordinate data to 1D space, we use the *Hilbert Curve*, a form of Space-Filling Curve which preserves locality. This is extremely beneficial for implementing k-NN queries in spacial databases such as maps. We apply the Hilbert Curve mechanism to the data structure to allow it to store and handle coordinate points.

C. Orthogonal Hilbert B-tree

The application of the Hilbert Curve to the B-tree in the previous step utilizes spacial locality to connect 1D data to 2D data. However, there exist a few coordinate points that are close together in 2D space but positioned further apart in linear space, as displayed in Figure 3. Therefore, the data is positioned separately in several B-trees on separate orthogonal Hilbert Curves. As a result, any two points which are positioned closely in 2D space are most likely close to each other on at least one of the orthogonal Hilbert Curves.

D. Oblivious B-tree

All operations made on the B-tree, such as `search()` and `insert()`, have the same number of accesses to memory so that they cannot be differentiated from one another. The server is now oblivious to the user's sequence of access patterns.

E. ORAM-Backed Hilbert B-tree

Additional obliviousness is added to the modified B-tree to create a ORAM-Backed B-tree. Each node of the B-tree is stored in a separate block in the ORAM. The ORAM algorithm encrypts and shuffles the logical addresses of the data stored in the B-tree, ensuring that the user's access patterns to data in the tree remain obfuscated.

F. Network ORAM-Backed Hilbert B-tree

The data structure is modified to operate as a cloud server with the use of Google Cloud Services. Now, the user can interact with a cloud to retrieve and query data obliviously while the cloud server handles data storage.

G. Performance and Testing

The data used to test the novel B-tree is obtained from publicly available GPS databases such as Geonames and OpenStreetMap. The accuracy and performance (in terms of run time) of the novel data structure is tested and compared to that of other data structures without an ORAM implementation. Factors such as the k value, input point of interest (such as hospitals or gas stations), and geographic location of the data set being queried is varied as to evaluate their impacts on runtime.

III. Conclusion

Access pattern leakage is a significant form of data leakage that can reveal confidential information about a user. However, our development of a novel B-tree data structure that combines existing algorithms and data structures such as k-Nearest Neighbor search queries, Hilbert space-filling curves, and ORAM algorithms, allows oblivious and dynamic storage of data as well as secure oblivious k-NN queries, functions that have not been developed for map applications. This ultimately mitigates access pattern leakage in map applications and prevents users from leaking information about their location. Through this novel development, we expect to produce a solution to accomplishes this goal with comparable accuracy and runtime.

References

[1] O. Goldreich and R. Ostrovsky. Software protection and simulation on oblivious RAMs. *J. ACM*, 43(3): 431–473, 1996.

[2] X.S. Wang, K. Nayak, C. Liu, T.-H.H. Chan, E. Shi, E. Stefanov, and Y. Huang. Oblivious Data Structures. *CCS*, 2014.

[3] R. Xu, K. Morozov, Y. Yang, J. Zhou, and T. Takagi. Efficient outsourcing of secure k-nearest neighbour query over encrypted database. *Computers and Security*, 69: 65–83, 2017.

[4] X. Yi, R. Paulet, E. Bertino, and V. Varadharajan. Practical approximate k nearest neighbor queries with location and query privacy. *IEEE Trans. Knowledge and Data Engineering*, 2016, 28(6): 1546-1559.

Oblivious k-Nearest Neighbors for Secure Map Applications

Jamie Lee [1] Adam J. Aviv [2] Travis Mayberry [1] Daniel S. Roche [1]

[1]United States Naval Academy and [2]George Washington University

Contributions

The novel ORAM backed Hilbert B-tree data structure accomplishes the following:

- Development of a new oblivious data structure to support 2D queries using $\mathbf{O(\log^2 n)}$ **bandwidth**
- Implementation of the data structure with **orthogonal Hilbert Curves and B-trees stored in an ORAM** to support oblivious 2D search queries, and
- **Oblivious k-NN search queries** for 2D map applications in cloud services
- Support of a **dynamic database** in which points can be added and/or removed obliviously

Introduction

Cloud storage enables cheap, efficient, and reliable data access and storage. **Map applications** utilize cloud storage servers and allow users to query for nearby facilities and locations such as restaurants, gas stations, or post offices.

Problem Statement

Cloud storage users are susceptible to data leakage through their access patterns. Even if the client trusts the server to store and retrieve data without modification, the server can attempt to obtain data about the user through factors relating to their access patterns. In map applications, the server may learn the user's location through their search queries, which is a significant security risk.

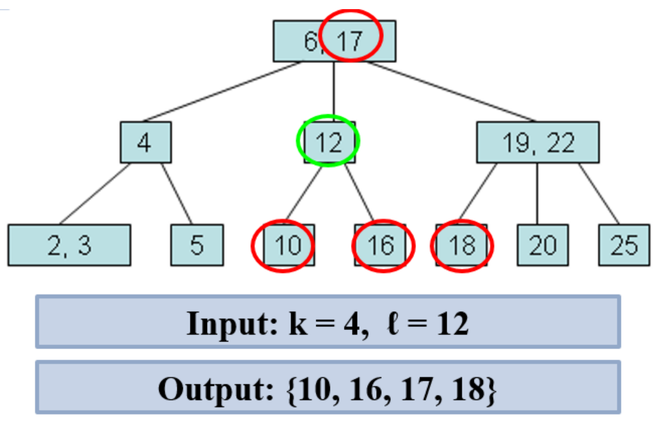

Figure 1:A sample k-NN search query in a B-tree for $\ell = 12$ and k = 4. The call to `search()` should return the list {10, 16, 17, 18}, as circled in red.

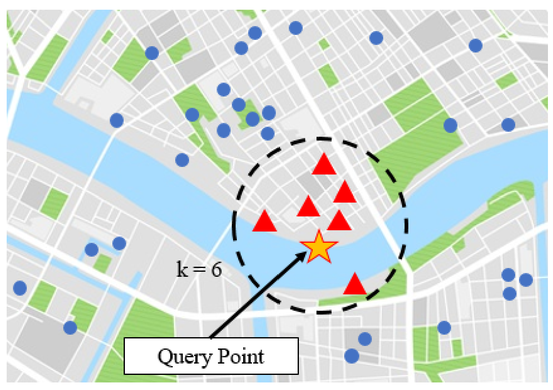

Figure 2:A sample k-NN search query with k = 6 made in a map application.

Solution

We propose a novel remotely-stored data structure, the **ORAM-backed Hilbert B-Tree**, with the following features:

- Modification of the **B-Tree data structure** to enable oblivious **k-Nearest Neighbor queries**
- Implementation of orthogonal **Hilbert space-filling curves** to store 2D coordinate data
- Storage of the B-Tree's nodes in an **oblivious RAM (ORAM)** to obfuscate the user's access patterns
- For any two sequences of logical accesses with the same number of operations, the transcripts for their physical accessses are indistinguishable [1].

Practical Impact and Relevance

This research **mitigates access pattern leakage in map applications and prevents users from leaking information about their location**, which provides a significant improvement in map application security without compromising performance. Our goal of enhancing map application security is aligned with the purpose of the NDSS of supporting research in network and distributed system security. This research describes one unique approach to preserving user privacy and developing trustworthy technology that can be used by users around the world.

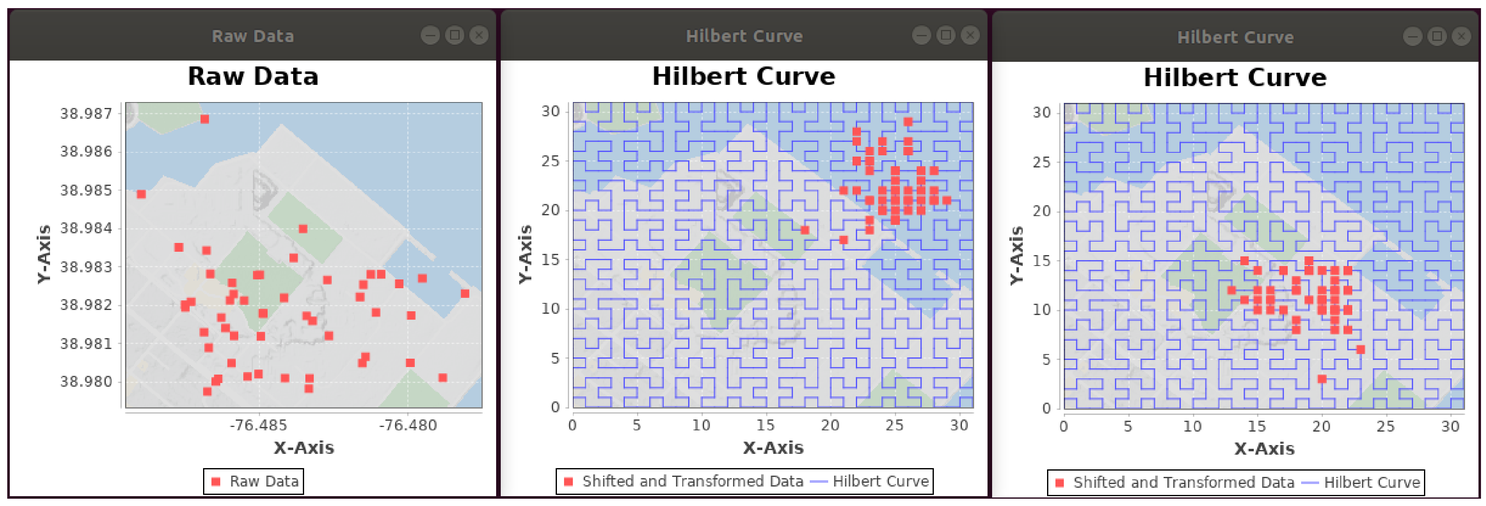

Figure 3:Plots generated with two Orthogonal Hilbert Curves. The leftmost plot displays the raw plot of points of interest in the U.S. Naval Academy. The middle and rightmost plots are the plot of the points randomly transformed to a curve of order 5.

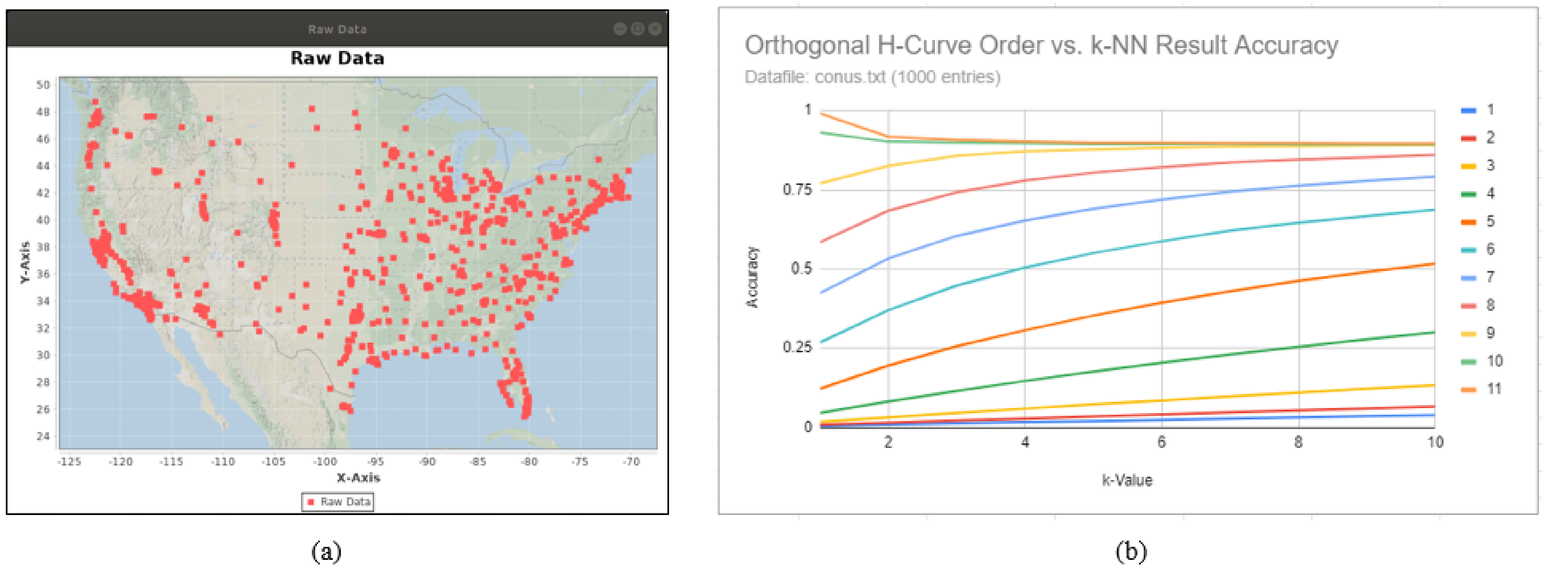

Figure 4:Accuracy measurements for k-NN query results made in a dataset of U.S cities for curve order n = 1, 2, ...10. For each curve order, the k-Nearest Neighbors for each point in the dataset were queried for k = 1, 2, ...10. This was conducted 20 times, and the results were averaged for each order. (a) Raw points plotted. (b) Results for two Orthogonal Hilbert Curves.

Conclusion

Our development of a novel B-tree data structure that combines existing algorithms and data structures such as k-Nearest Neighbor search queries, Hilbert curves, and ORAM algorithms, allows users to make k-NN queries securely, a function that has not been developed for map applications. Through this development, we expect to produce a solution to access pattern leakage in map applications that also performs with comparable accuracy and runtime.

References

[1] O. Goldreich and R. Ostrovsky.
Software protection and simulation on oblivious rams.
J. ACM, 43(3):431–473, 1996.

[2] C. Liu T.-H.H. Chan E. Shi E. Stefanov X.S. Wang, K. Nayak and Y. Huang.
Oblivious data structures.
CCS, 2014.

[3] Y. Yang J. Zhou R. Xu, K. Morozov and T. Takagi.
Efficient outsourcing of secure k-nearest neighbour query over encrypted database.
Computers and Security, 69:65–83, 2017.