IMPACT: International Journal of Research in Engineering & Technology (IMPACT: IJRET)
ISSN(E): 2321-8843; ISSN(P): 2347-4599
Vol. 2, Issue 6, Jun 2014, 9-16

AN EFFECTIVE SPATIAL QUERY INTEGRITY USING MERKLE HASH TREE

R. G. SAKTHIVELAN & ASHIBA SUNDARAM

Department of Computer Science, Anna University, Mahendra Engineering College, Namakkal, Tamil Nadu, India

ABSTRACT

In Database outsourcing paradigm, the authentication of the query results at the client remains a challenging problem. Existing system focuses on the Outsourced Spatial Database (OSDB) model and propose an efficient scheme, called VN-Auth, which allows a client to verify the correctness and completeness of the result set. This approach is based on neighborhood information derived from the Voronoi diagram of the underlying spatial data set and can handle fundamental spatial query types, such as k nearest neighbor and range queries, as well as more advanced query types like reverse k nearest neighbor, aggregate nearest neighbor, and spatial skyline. In VN-authentication framework system the authentication of the mobile clients usually have low selectivity (i.e., the user is interested in very few results) but require less time to response. If the query selectivity is high it requires more time for a response and it becomes the major difficult in VN-auth.

Hence proposed Merkle Hash Tree (MHT)-based approaches with VN-auth to reduce the time complexity when query selectivity is high. Merkle Hash Tree (MHT)-based approaches with VN-auth is named as VMN-Authentication framework. The Merkle Hash Tree (MHT) tree is constructed and signed by the certification authority and then distributed to untrusted directory services. Entities wishing to verify the validity of a certificate can query such a directory service and have confidence in the correctness of a response by using the returned data to verify the certificate authority's signature. Also it reduces the complexity of the system. In the extreme case where the client retrieves the whole database, the MHT approach would only return the root signature, while VN-Auth would need to return 6.n neighbors, where n is the database cardinality.

KEYWORDS: Query Authentication, Spatial Queries, Outsourced Databases

1. INTRODUCTION

Database outsourcing is a new paradigm that has been proposed recently and has received a lot of attention in the research community. The basic idea is that data owners (DO) delegate their database maintenance and functionalities to a third-party service provider (SP), and the SP is responsible for indexing the data and answering client queries. Since the third party can be untrusted or can be compromised, security concerns must be addressed before this delegation. There are three main entities in the Outsourced Database (ODB) model: the data owner, the database service provider and the client. In practice, usually there is a single or a few data owners, a few servers, and many clients. The data owner creates the database, along with the associated index and authentication structures and uploads to the servers.

The data owner may update the database periodically or occasionally, and that the data management and retrieval happens only at the servers. Clients may submit queries about the owner's data to the servers and get back results through the network. It is cheaper to maintain ordinary servers than to maintain truly secure ones, particularly in the distributed

environment. To guard against malicious/compromised servers, the owner must give the clients the ability to authenticate the answers they receive without having to trust the servers on the network. In that respect, query authentication has two important dimensions: correctness and completeness. Correctness means that the client must be able to validate that the returned records do exist in the owner's database and have not been modified. Completeness means that no answers have been omitted from the result which is returned back to the client.

We assume that the clients are mobile users who issue location-based queries (e.g., *k*NN or range queries), in order to discover points of interest (POIs) in their neighborhood. Since the SP is not the real owner of the data, *query integrity assurance* is a challenging problem that has to be addressed. In particular, the SP has to prove to the client that (i) the data is originated from the DO and, (ii) the result set is correct and complete.

The general framework that is commonly used is based on digital signatures and utilizes a public-key cryptosystem, such as RSA. Initially, the DO obtains, a *private* and a *public* key, through a trusted key distribution center. The public key is accessible by all the clients and private key is kept secret at the DO. Using private key, the DO digitally signs the data, by generating signatures. Then, it sends the signatures and the data to the SP, which constructs the necessary data structures for efficient query processing. When the SP receives a query request from a client, it generates a *verification object* (*VO*) that contains the result set along with the necessary authentication information. Finally, the SP sends the *VO* to the client, which can verify the results using the public key of the owner.

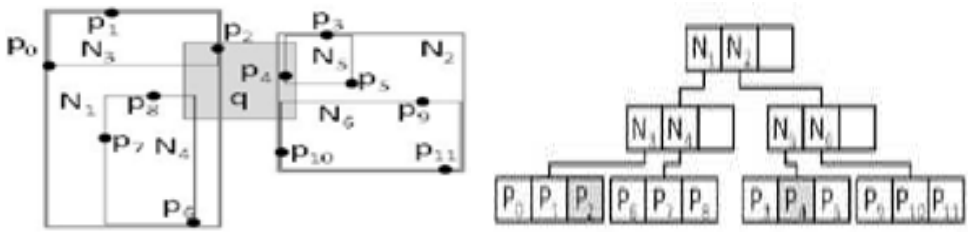
Figure 1: System Architecture

Currently, the state-of-the-art solution for authenticating spatial queries is Merkle R-tree (MR-tree). The MR-tree is a R-tree that is augmented with authentication information (i.e., hash digests). Every leaf node of the tree stores a digest that is computed on the concatenation of the binary representation of all objects in the node. Internal nodes are assigned a digest that summarizes the child nodes' MBRs (minimum bounding rectangles) and digests. Digests are computed in bottom-up fashion and the single digest at the root is signed by the DO. The resulting *VO* contains (i) all the objects in every leaf node visited, and (ii) the MBRs and digests of all the pruned nodes. With this information, the client can reconstruct the root digest and compare it against the one that was signed by the data owner. In addition, the client also examines the spatial relations between the query and each object/MBR included in the *VO*, in order to verify the correctness of the result.

The structure of the MR-tree as well as the verification process, suffer from several drawbacks. First, the authentication information (hash digests) embedded in the MR-tree reduces the node fanout, leading to more I/O accesses during query processing. Second, in the presence of updates, all the digests on the path from an affected leaf node to the root have to be recomputed. When updates are frequent, query performance is degraded. Finally, the overhead

of the *VO* can be significant, especially for queries that return only a few objects. This is due to the fact that SP has to return all objects that lie inside the leaf nodes that are visited during query processing. As an example, consider the range query *q* in Figure 2. Even though the result set includes only two objects (*p*2, *p*4), the corresponding *VO* has to return all 12 objects in the database. An extension of the MR-tree, called MR*-tree, mitigates this drawback, by ordering the entries of each node and constructs hierarchical relationships of the digests therein. It does not eliminate the *VO* overhead entirely, while at the same time it increases the verification cost at the client.

Then came VN-Auth, a novel approach that authenticates arbitrary spatial queries based on neighborhood information derived by the Voronoi diagram of the underlying spatial dataset. In particular, before delegating its database to the SP, the owner transforms each data object by creating a signature of the object itself along with information about its Voronoi neighbors. A key aspect of this method is that it separates the authentication information from the spatial index. As a result, all updates are restricted in the neighborhood of the affected objects. Furthermore, the *VO* only includes the transformed objects that belong to the result set. Compared to the MR-tree variants, VN-Auth produces significantly smaller verification objects and is more computationally efficient, especially for queries with low selectivity (ie. The user is interested in very few results). If the query selectivity is high it requires more time for a response and it becomes the major difficult in VN-Auth.



(a) Points and MBRs **(b) MR-Tree**

Figure 2: Range Query on MR-Tree

Hence proposed Merkle Hash Tree (MHT)-based approaches with VN-auth to reduce the time complexity when query selectivity is high. Merkle Hash Tree (MHT)-based approaches with VN-auth is named as VMN-Authentication framework. The Merkle Hash Tree (MHT) tree is constructed and signed by the certification authority and then distributed to untrusted directory services. Entities wishing to verify the validity of a certificate can query such a directory service and have confidence in the correctness of a response by using the returned data to verify the certificate authority's signature. Also it reduces the complexity of the system. In the extreme case where the client retrieves the whole database, the MHT approach would only return the root signature, while VN-Auth would need to return 6.n neighbors, where n is the database cardinality.

The remainder of the paper is organized as follows. Section 2 discusses Voronoi diagrams. Section 3 describes signature aggregation techniques. Section 4 describes the data transformation process, and Section 5 describes signature verification and introduces MHT for authentication. Section 6 concludes the paper with directions for future work.

2. PRELIMINARIES

2.1 Voronoi Diagrams

Given a set of distinct objects $Q= \{q1,q2,q3,\ldots q_n\}$ in R^m, the *Voronoi diagram* of *Q*, denoted as VD(*Q*), partitions

the space of R^m into *n* disjoint regions, such that each object q_i in *Q* belongs to only one region and every point in that region is closer to q_i than to any other object of *Q* in the Euclidean space. The region around q_i is called the *Voronoi cell* of q_i, denoted as $V\,C(q_i)$, and q_i is the *generator* of the Voronoi cell. Therefore, the Voronoi diagram of *Q* is the union of all oronoi cells VD(*Q*) ={$V\,C(q_1)$, $V\,C(q_2)$,...., $V\,C(q_n)$}. If two generators share a common edge, they are Voronoi neighbors.

If we connect all the Voronoi neighbors, we get the Delaunay triangulation DT(Q), which is the dual graph of VD(Q). Distance functions are utilized in R^2.

- **Property 1:** Given a set of distinct points P = {p_1, p_2,…. p_n} $\subset$ R^2, the Voronoi diagram VD(P) and the corresponding Delaunay triangulation DT(P) of P are unique.
- **Property 2:** The average number of Voronoi edges per Voronoi polygon does not exceed six. That is, the average number of Voronoi neighbors per generator does not exceed six.
- **Property 3:** Given the Voronoi diagram of P, the nearest neighbor of a query point q is p, if and only if q $\in$ V C(p).
- **Property 4:** Let p_1,…, p_k be the k (k > 1) nearest neighbors in P to a query point q. Then, p_k is a Voronoi neighbor of at least one point $p_i \in \{.p_1,..p_{k-1}\}$

3. SIGNATURE AGGREGATION

In our approach, the DO generates a signature for every object in the database, which is computed on the hash digest of the concatenation of the binary representation of the object and its Voronoi neighbors. In this way, the client can verify the authenticity of each object and its neighborhood. Here, we utilize RSA signatures which are typically 128 bytes in size. Alternatively, signatures based on Elliptic Curve Cryptography (ECC) can be used and which is significantly shorter, thus reduces the overall communication and storage cost. However, ECC algorithms are computationally very intensive, and would perform poorly on mobile devices with limited computational capabilities. The drawback of having one signature per database object is that it may increase the communication cost between the SP and the client.

Specifically, the SP has to transmit one 128-byte signature for every object in the result set, so the overhead will be large for queries with high selectivity (especially for mobile clients). To avoid this cost, a technique is employed, which is called *signature aggregation*. In particular, given *k* digests and their corresponding signatures (generated by the same signer), the SP can replace them with a single *Condensed-RSA* signature. Condensed-RSA has the same size as the original signatures (128 bytes), and is computed as the modular multiplication of the *k* signatures. Aggregate signatures are secure and can be computed by any party that possesses the individual signatures.

4. DATA TRANSFORMATION

Consider a DO that has compiled a large collection of *n* Point Of Interests (POIs) (e.g., restaurants) within a geographic region. Each POI *i* is represented as a unique object *pi* in the database, which has the form *⟨pi.location, pi.tail⟩*. The *location* attribute stores the spatial coordinates of the object and *tail* attribute stores some additional information about the object, such as name, address, phone number, web site, etc. Before transmitting the database to the SP, the DO transforms each object by attaching neighborhood and authentication information.

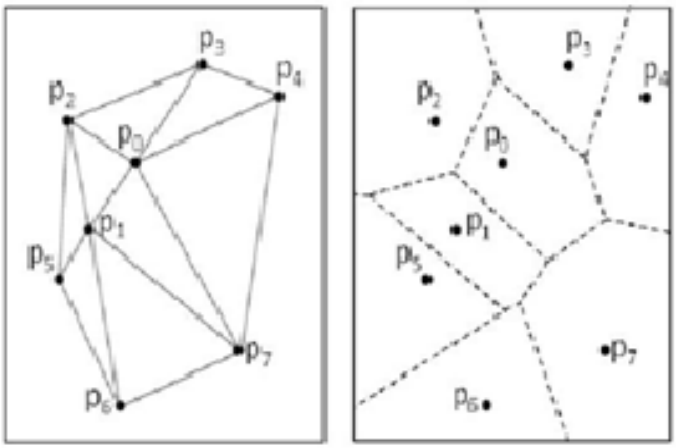


Figure 3: Spatial Dataset Example

Here, the DO initially computes the Voronoi diagram of the spatial dataset (as shown in Figure 3) and retrieves the Voronoi neighbors of each POI. Then, it appends a *neighbors* attribute to each and every object in the database that stores the locations of all its Voronoi neighbors. For example, the *neighbors* attribute of *p*5 is equal to:

*p*5.*neighbors* = {3, *p1.location; p4.location; p8.location*} Note that, the first value of the attribute specifies the exact number of neighbors. Here, we assume that the DO stores the Voronoi neighbors in a clockwise or counter-clockwise order to facilitate the on-the-fly reconstruction of Voronoi cells at the client. The final step for the DO is to sign each individual object, so that the client can verify the authenticity of the information. Specifically, for an object *pi*, its signature S is computed as S = *sign*(*h*(*pi.location*|*pi.tail*|*pi.neighbors*))

where *h* is a one-way, collision-resistant hash function and '|' denotes the concatenation of two binary strings. To summarize, each transformed object *pi* at the DO's site has the form *⟨pi.location, pi.tail,pi.neigbors,pi.*S*⟩.*

After the completion of data transformation process, the DO transfers all objects to the SP. Upon receiving the database objects, the SP builds appropriate spatial indices, and is then ready for query processing. The leaf level of the index only stores pointers to the location of the transformed objects on the disk.

Compared to the MR-tree variants, this approach has several advantages. First, the DO is oblivious to the query processing mechanisms at the SP. Consequently, the SP may utilize any spatial index and query processing algorithm without informing the DO. Second, the spatial index does not store any authentication information, and thus remains very compact and efficient. Third, database updates affects only their local regions, and do not propagate to the root of the index.

5. QUERY PROCESSING AT THE SP

Service Providers process queries on outsourced database (database stored in the cloud) on behalf of the data owner. In the case of verifiable queries, returning the query result to the clients is no longer sufficient. Instead, SP is required to return a Verification Object (VO) that contains a condensed signature that verifies the authenticity of all objects in VO and the result set of the query with some additional objects which are necessary for verification process.

6. SIGNATURE VERIFICATION

Generally two sequential steps are taken in a spatial query verification process. First, the aggregate signature of the VO is examined by the client to ensure that all objects which are returned originated at the DO. Second, all objects in the result set are evaluated to ensure that the geometric properties are satisfied and no legitimate objects are eliminated. Signature verification is common in all query types. When a client receives VO from SP, it verifies the aggregate signature using the public key of the DO. Specifically, the client simply performs a modular multiplication of the hash digests of all objects included in the VO, and verifies that the result matches the plaintext that is derived by decrypting the aggregate signature with the DO's public key. If the VO fails the signature verification process, the client considers that the result as corrupted and the verification process terminates. Otherwise, it continues with the geometric verification.

For the purpose of query authentication, Merkle Hash Tree is used, and is described in the following section.

6.1 The Merkle Hash Tree

An improved solution for authenticating a set of data values is the Merkle hash tree (see Figure 4). It solves the simplest form of the query authentication problem for point queries and datasets that can fit in main memory. The Merkle hash tree is a binary tree, where each leaf node contains the hash of a data value, and each internal node contains the hash of the concatenation of its two children. Verification of data values is based on the fact that the hash value of the root of the tree is authentically published (authenticity can be established by a digital signature). To prove the authenticity of any data value, in addition to the data value, the prover has to provide the verifier, the values stored in the siblings of the path that leads from the root of the tree to that value. The verifier, by iteratively computing all the appropriate hashes up the tree, can simply check if the hash computed for the root matches the authentically published value.

The security of the Merkle hash tree is based on the collision-resistance of the hash function used: It is computationally infeasible for a malicious prover to fake a data value, since this would require finding a hash collision somewhere in the tree (because the root remains the same and the leaf is different and hence, there must be a collision somewhere in between). Thus, the authenticity of any one of n data values can be proven at the cost of providing and computing $\log_2$ n hash values, which is much cheaper than storing and verifying one digital signature per data value. Furthermore, the relative position (leaf number) of any of the data values within the tree is authenticated along with the value itself.

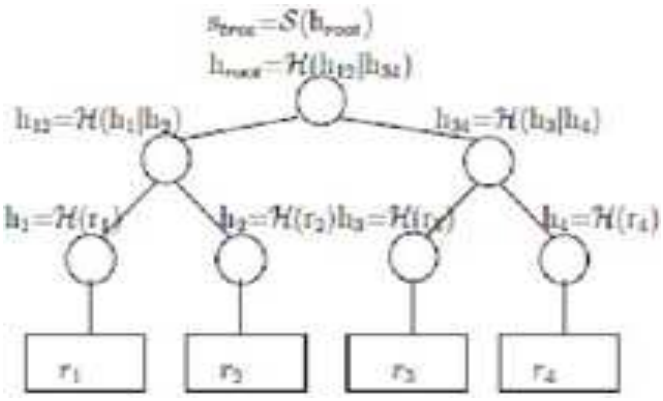


Figure 4: Merkle Hash Tree example

7. CONCLUSIONS

In this paper, we introduced Merkle Hash Tree approaches with VN-Auth query integrity assurance framework for outsourced spatial databases. Our approach separates authentication information from spatial index, thus allows efficient query processing at the service provider. Since the verification information depends only on the object and its Voronoi neighbors, database updates can be disseminated quickly to their local regions and are performed independently of all other updates in the database. Finally, we showed that our approach incurs lower query verification cost, especially for queries with high selectivity.

ACKNOWLEDGEMENTS

This work has been supported by Mahendra college authorities. So, we do here by acknowledge them and thanks for their guidance and encouragement.

REFERENCES

1. W. Cheng, H. Pang, and K.-L. Tan. Authenticating Multi-dimensional Query Results in Data Publishing. In *DBSec*, pages 60-73, 2006.
2. W. Cheng and K.-L. Tan. Authenticating kNN Query Results in Data Publishing. In *Secure Data Management*, pages 47-63, 2007.
3. W. Cheng and K.-L. Tan. Query assurance verification for outsourced multi-dimensional databases. *Journal of Computer Security*, vol 17 no.1, pp. 101-126, 2009.
4. A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD Conference*, pages 47-57, 1984.
5. H. HacigÄumÄus, B. R. Iyer, C. Li, and S. Mehrotra. Executing SQL over Encrypted Data in the Database-service-provider Model. In *SIGMOD Conference*, pages 216-227, 2002.
6. H. HacigÄumÄus, S. Mehrotra, and B. R. Iyer. Providing Database as a Service. In *ICDE*, page 29, 2002.
7. M. R. Kolahdouzan and C. Shahabi. Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases. In *VLDB*, pages 840-851, 2004.
8. W.-S. Ku, L. Hu, C. Shahabi, and H. Wang. Query integrity assurance of location-based services accessing outsourced spatial databases. In *SSTD*, pages 80-97, 2009.
9. F. Li, M. Hadjieleftheriou, G. Kollios, and L. Reyzin. Dynamic Authenticated Index Structures for Outsourced Databases. In *SIGMOD Conference*, pages 121-132, 2006.
10. E. Mykletun, M. Narasimha, and G. Tsudik. Authentication and Integrity in Outsourced Databases. In *NDSS*, 2004.
11. A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. Probability and Statistics. Wiley, NYC, 2nd edition, 2000.

12. H. Pang, A. Jain, K. Ramamritham, and K.-L. Tan. Verifying Completeness of Relational Query Results in Data Publishing. In *SIGMOD Conference*, pages 407-418, 2005.
13. H. Pang and K.-L. Tan. Authenticating Query Results in Edge Computing. In *ICDE*, pages 560-571, 2004.
14. M. Sharifzadeh and C. Shahabi. VoR-tree: R-trees with Voronoi Diagrams for Efficient Processing of Spatial Nearest Neighbor Queries. *PVLDB*, 3(1), 2010.
15. R. Sion. Query Execution Assurance for Outsourced Databases. In *VLDB*, pages 601-612, 2005.