ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 2, No 5, May 2013

An Overview of Partial Shuffle for Database Access Pattern Protection Using Reverse Encryption Algorithm

Priti V. Bhagat[1], Rohit Singhal[2]

[1]M.Tech student, Department of Computer Science & Engineering, I.E.T. Alwar, Rajasthan, India

[2]Astt. Professor, Department of Computer Science & Engineering, I.E.T. Alwar, Rajasthan, India

Abstract - **Encryption of database is an important topic for research, as secure and efficient encryption algorithms are needed that provide the ability to query over encrypted database and allow optimized encryption and decryption of data. There is always a compromise between the degree of security provided by encryption algorithm and the efficient querying on the database, because the encryption and decryption on database greatly degrade query performance. For this, we propose a new encryption algorithm; Reverse Encryption Algorithm (REA). REA is simple and fast enough for most applications. REA provides maximum security and limits the added time cost for encryption and decryption to as to not degrade the performance of a database system.**

Privacy protection is one of the fundamental security requirements for database outsourcing. A major threat is information leakage from database access patterns generated by query executions. Recent works propose to protect access patterns by introducing a trusted component with constant storage size. The resulting privacy assurance is as strong as PIR, though with O(1) online computation cost, they still have O(n) amortized cost per query due to periodically full database shuffles. In this wok, we design a novel scheme in the same model with provable security, which only shuffles a portion of the database.

Keywords: **Database, data privacy, information security**

I. INTRODUCTION

Privacy protection is one of the fundamental security requirements for database outsourcing. A major threat is information leakage from database access patterns generated by query executions. Private Information Retrieval (PIR) protocol allows a user to retrieve an item from a server in possession of a database without revealing which item they are retrieving. PIR is a weaker version of 1-out-of n oblivious transfer, where it is also required that the user should not get information about other database items. While this problem admits a trivial solution – sending the entire database to the client allows the client to query with perfect privacy-there are techniques to reduce the communication complexity of this problem, which can be critical for large databases. The Strong Private Information Retrieval (SPIR) is the retrieval with the additional requirements that the client only learn about the elements of the query. This requirements typical privacy needs of a database owner.

Private Information Retrieval (PIR) formulated the well-known cryptographic mechanism inhibiting information leakage from access patterns. Many PIR schemes have been proposed with the emphasis on lowering the communication complexity between the server and the user. Nonetheless, as pointed out by Sino and Carbunar, those PIR schemes incur even more turnaround time than transferring the entire database as a reply to the user, because the heavy computation incurred at the server outweighs the saved communication expense.

The Private Information Retrieval problem is only concerned with user's privacy, without requiring any protection of server's privacy. The database use in the different session process in different areas. The entire database as a reply to the user, because the heavy computation incurred at the server outweighs the saved communication expense. Compared with the standard PIR schemes, these PIR schemes works on encrypted data records rather than bits in plaintext.

However, how to query efficiently on the encrypted database becomes a challenge. This usually found that the system has to sacrifice the performance to obtain the security. When data is stored in encrypted form, we have to decrypt all the data before querying them. It is impractical because the cost of decryption over all the encrypted data is very expensive.

For this purpose, we put forward the innovative encryption algorithm, known as "Reverse Encryption Algorithm (REA)". Reverse Encryption Algorithm is efficient and reliable. To protect access pattern of the database generated by query, we follow this line of research and design a novel scheme which only shuffles a portion of the database.

II. THE MODELS OF PRIVATE INFORMATION RETRIEVAL

A. *Information-Theoretic PIR.*

Information-theoretic PIR protocols guarantee perfect privacy – even an unbounded server learns no information. However, they require replication of the data among several non-communicating servers. Information-theoretic PIR protocols were introduced and constructed by Chor et al. [1]. In particular, they construct the best known 2-server protocol with communication complexity $O(n^{1/3})$ (where n is the database length). More efficient constructions of k-server protocols for k > 2 appear in [2], [3], [4]. The best known 3-server PIR protocol is constructed in a lovely work of Yekhanin [5]; assuming that there are infinitely many Mersenne primes, he constructs a 3-server PIR protocol with communication complexity nO(1/log log n) for infinitely many values of n. Specifically, his protocol implies, without any assumptions, a 3-server PIR protocol with communication complexity n^{ε} for some $\varepsilon < 10^{-7}$. To date, Yekhanin's protocol is the best k-server PIR protocol for every constant k.

B. Computational PIR.

In computational PIR protocols a polynomial-time server cannot learn information on the index the client retrieves. In other words, unless the server runs in an unreasonable time, the privacy of the index is guaranteed. The first computational PIR protocol was a multi-server PIR protocol of [6] (assuming that one-way functions exist). Following this work, Kushilevitz and Ostrovsky [7] showed that there is a 1-server computational PIR protocol with sub-linear communication (assuming that the quadratic residuosity problem is hard). Subsequently, more efficient computational protocols, based on various hardness assumptions, were constructed [7], [8], [9]. The best 1-server PIR protocol was constructed by Lipmaa [8]; it is based on the so-called composite residuosity assumption and has communication complexity $O(\log^2 n)$ (ignoring the security parameter). It is important to note that 1-server computational PIR protocols with sublinear communication require some hardness assumptions [10].

Computational PIR protocols are used for constructing efficient protocols for more complex crypto-graphic tasks. They can be used to construct several cryptographic primitives, e.g., unconditionally hiding commitment [10], oblivious transfer protocols [11], and collision-resistant hash functions [12]. Further-more, PIR protocols can be used to construct efficient zero-knowledge arguments for a certain class of languages [13]. The twist is that the server performing the encryption should not know the criteria for choosing the relevant information; nevertheless, the length of the encryption should be shorter than the encrypted database. Other applications of PIR protocols for complexity theory are discussed in the survey [16].

C. Symmetric PIR.

In the above discussion of the PIR problem, we only protect the privacy of the client. While each server is not allowed to learn information about the bit that the client is interested in, the client can learn many bits of the database. This might be problematic in many scenarios. For example, if the server wants to charge the client for each bit it retrieves, then the client gets extra information for free. Gertner, Ishai, Kushilevitz, and Malkin [17] defined symmetric private information retrieval, abbreviated SPIR, where the server does not learn any information and the client only learns the bit that it wants. Such protocols were actually considered before the introduction of PIR protocols and were called oblivious transfer, abbreviated OT, or all-or-nothing disclosure of secrets [18]. The name oblivious transfer, coined by Rabin [18], illustrates the nature of the protocol where the server transfers information to the client, while being oblivious to which information it transfers. However, prior to the PIR literature, the communication complexity of OT protocols was not optimized. In other words, SPIR protocols can be thought of as oblivious transfer protocols with sub-linear communication.

III. PROPOSED METHODOLOGY

In this work, we shall study various approaches which are followed in realizing this system. We will further study factual aspects which can be made use of in designing and developing an efficient database access pattern protection with partial shuffle scheme. The important consideration made in this work is of using the existing standard methods and have developed the innovative algorithm with different functionalities.

A. System Model

The system consists of a group of users, a database D modeled as an array of n data items of equal length denoted by {d1,d2,....dn} , and a database host denoted by H . A trusted component denoted by T is embedded in H. T has an internal cache which stores up to k data item. No adversary can tamper T's executions or access its private space including the cache. T is capable of performing symmetric key encryption/decryption and pseudorandom number generation.

B. Basic Construction

1. Reverse Encryption Algorithm

We recommend the new encryption algorithm, "Reverse Encryption Algorithm (REA)", because of its simplicity and efficiency. Reverse Encryption Algorithm limits the added time cost for encryption and decryption. In this section we provide a comprehensive yet concise algorithm.

Reverse Encryption Algorithm is a symmetric stream cipher that can be effectively used for encryption and decryption of data. It takes a variable-length key. The Reverse Encryption Algorithm encipherment and decipherment consists of the same operations, except the two operations: 1) adds the key to the text in the encipherment and removes the keys from the text in the decipherment. 2) Execute divide operation on the text by 4 in the encipherment and execute multiply operation on the text by 4 in the decipherment. We execute divide operation by 4 on the text to narrow the range domain of the ASCII code table at converting the text.

Encryption Algorithm of the REA

The steps are (see Figure 1):
Step 1: Input the text and the key.
Step 2: Add the key to the text.
Step 3: Convert the previous text to ASCII code.
Step 4: Convert the previous ASCII code to binary data.
Step 5: Find out One's complement of the previous binary data.
Step 6: Gather each 8 bits from the previous binary data and obtain the Decimal value from it.
Step 7: Divide the previous Decimal value by 4.
Step 8: Obtain the ASCII code of the previous result divide and put it as one character.
Step 9: Obtain the remainder of the previous divide and put it as a second character.
Step 10: Return encrypted text.

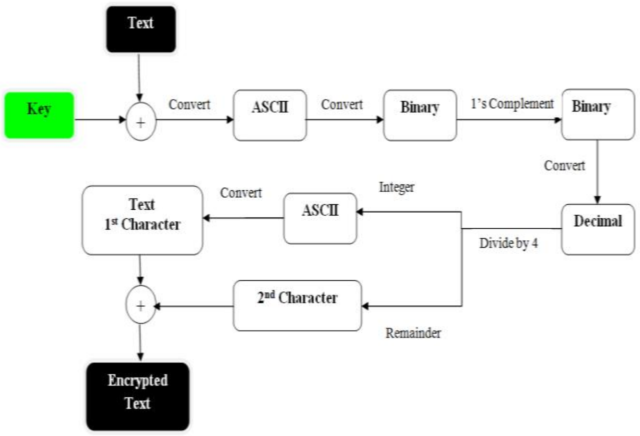

Figure 1: Block Diagram of REA encryption algorithm

Decryption Algorithm of the REA

The steps are (see Figure 2):
Step 1: Input the encrypted text and the key.
Step 2: Loop on the encrypted text to obtain ASCII code of characters and add the next character.
Step 3: Multiply ASCII code of the first character by 4.
Step 4: Add the next digit (remainder) to the result multiplying operation. (Consider result as Decimal value)
Step 5: Convert the previous Decimal value to binary data.
Step 6: Find out One's complement of the previous binary data.
Step 7: Gather each 8 bits from the previous binary data and obtain the ASCII code from it.
Step 8: Convert the previous ASCII code to text.
Step 9: Remove the key from the text.
Step 10: Return decrypted data.

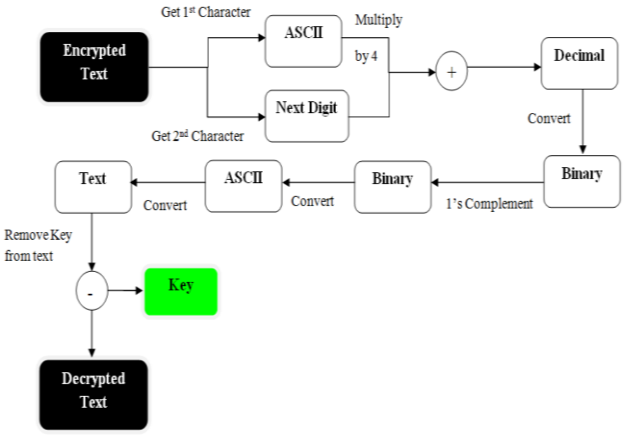

Figure 2: Block Diagram of REA decryption algorithm

2. Twin Retrieval Algorithm

Initially, all the entries of database are labeled as white. Once a record is fetched, it is labeled as black. For a query on di, T executes a twin retrieval algorithm, if di is available in the cache, T randomly fetches a pair of records, black and white, respectively; otherwise, it retrieves the needed record and another random record in a different color.

Algorithm

```
INPUT: a query on di, B.
OUTPUT: di.
If di not in the cache then
j ← σ (i).
u ← binary_search (j, B);
If u ≠ NULL then
di is black; set v ← B [πs (u)] and read Ds[v] and read a random white record from database;
Else
di is white; read a random black record from database and read D[j] which stores di;
End if
Else
Read a random black and a white record from Ds into the cache.
End if
Return di to the user.
```

Where, di is the ith entry in the original database D, B is the array of addresses of all black records, stored in ascending order. σ is the initial permutation used for shuffling and Ds[v] is vth entry in Ds.

3. Partial Shuffle Algorithm

To protect information leakage from database access pattern generated due to query executions by shuffling database entries. Due to full database shuffle computation cost increases. So, in the proposed scheme only Black records (touched records) are shuffled and re-encrypted. Note that it is unnecessary to shuffle white records (untouched records). A white record does not leak any query information for the following two reasons. First, all records are encrypted and therefore a white record itself does not compromise privacy. Second, since it is white, there exists no access pattern involving it. Therefore, it is observed that an encrypted record is not touched does not help the adversary to derive any information about (existing) user queries. The objective of database shuffle is to remix the touched database entries with the untouched ones, so that future executions appear independent with preceding ones.

Algorithm

```
INPUT: B with (1+s) k/2 black records.
OUTPUT: Ds+1.
Secretly generate a random permutation πs+1: [1, |B|] → [1, |B|], and a new key sks+1.
For (I= $I_f$=1, I ≤ |B| - k; I++) do
While True do
j ← $\pi^{-1}_{s+1}$ ($I_f$);  t ← $\sigma^{-1}$ (B [j]);
If $d_t$ is in the cache, $I_f$ ← If+1; else break;
End while
δ ← |{di|di is in cache and is white and σ(i) < B[j]}|,
v ← πs(j-δ);
δ ← |{di|di is in cache and is white and σ(i) < B[v]}|,
v ← v +δ;
Fetch $D_s$ [B[v]] as $d_t$.
If I = $I_f$ then
Write $\varepsilon_{sks+1}$($d_t$, t) into $D_{s+1}$[B [I]];
Else
Insert (t, $d_t$) into cache.
j ← $\pi^{-1}_{s+1}$ (I);  t ← $\sigma^{-1}$ (B [j]);
Retrieve $d_t$ from the cache and write $\varepsilon_{sk+1}$($d_t$, t) to $D_{s+1}$[B [I]].
End If
$I_f$ = $I_f$ + 1;
```

End for
Encrypt and write the remaining k records in to the cache to D_{s+1} according, securely eliminate π_{s-1}. Quite the s^{th} sessions.

IV. CONCLUSION

We have presented a novel hardware-based scheme to prevent database access patterns from being exposed to a malicious server. By virtue of twin-retrieval and partial-shuffle, our scheme avoids full-database shuffle and reduces the amortized server computation complexity.

Cryptographic support is an important mechanism of securing important data. In this work, we introduce a new encryption algorithm, which we call "Reverse Encryption Algorithm (REA)". REA is simple and fast enough for most applications. Our new encryption algorithm REA can reduce the cost time of the encryption/decryption operations and improve the performance.

REFERENCES

[1]. B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan. Private information retrieval. In *Proc. of the 36th IEEE Symp. on Foundations of Computer Science*, pages 41–51, 1995. Journal version: *J. of the ACM*, 45:965–981, 1998.

[2]. A. Ambainis. Upper bound on the communication complexity of private information retrieval. In P. Degano, R. Gorrieri, and A. Marchetti-Spaccamela, editors, *Proc. of the 24th International Colloquium on Automata, Languages and Programming*, volume 1256 of *Lecture Notes in Computer Science*, pages 401–407. Springer-Verlag, 1997.

[3]. A. Beimel, Y. Ishai, and E. Kushilevitz. General constructions for information-theoretic private infor-mation retrieval. *J. of Computer and System Sciences*, 71(2):213–247, 2005.

[4]. A. Beimel, Y. Ishai, E. Kushilevitz, and J. F. Raymond. Breaking the $O(n^{2k_{¡}1}$) barrier for information-theoretic private information retrieval. In *Proc. of the 43rd IEEE Symp. on Foundations of Computer Science*, pages 261–270, 2002.

[5]. S. Yekhanin. Towards 3-query locally decodable codes of subexponential length. In *Proc. of the 39th ACM Symp. on the Theory of Computing*, pages 266–274, 2007.

[6]. B. Chor and N. Gilboa. Computationally private information retrieval. In *Proc. of the 29th ACM Symp. on the Theory of Computing*, pages 304–313, 1997.

[7]. E. Kushilevitz and R. Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *Proc. of the 38th IEEE Symp. on Foundations of Computer Science*, pages 364–373, 1997.

[8]. H. Lipmaa. An oblivious transfer protocol with log-squared communication. In J. Zhou and J. Lopez, editors, *the 8th Information Security Conference (ISC'05)*, volume 3650 of *Lecture Notes in Computer Science*, pages 314–328. Springer-Verlag, 2005.

[9]. E. Mann. Private access to distributed information. Master's thesis, Technion – Israel Institute of Technology, Haifa, 1998.

[10]. A. Beimel, Y. Ishai, E. Kushilevitz, and T. Malkin. One-way functions are essential for single-server private information retrieval. In *Proc. of the 31st ACM Symp. on the Theory of Computing*, pages 89–98, 1999.

[11]. G. Di-Crescenzo, T. Malkin, and R. Ostrovsky. Single-database private information retrieval implies oblivious transfer. In *Advances in Cryptology – EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 122–138. Springer-Verlag, 2000.

[12]. Y. Ishai, E. Kushilevitz, and R. Ostrovsky. Sufficient conditions for collision-resistant hashing. In J. Kilian, editor, *Proc. of the Second Theory of Cryptography Conference – TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 445–456. Springer-Verlag, 2005.

[13]. Tauman Kalai and R. Raz. Succinct non-interactive zero-knowledge proofs with preprocessing for LOGSNP. In *Proc. of the 47th IEEE Symp. on Foundations of Computer Science*, pages 355–366, 2006.

[14]. R. Ostrovsky and W. E. Skeith III. Private searching on streaming data. *J. of Cryptology*, 20(4):397– 430, 2007.

[15]. D. Boneh, E. Kushilevitz, R. Ostrovsky, and W. E. Skeith III. Public key encryption that allows PIR queries. In A. Menezes, editor, *Advances in Cryptology – CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 50–67. Springer-Verlag, 2007.

[16]. R. Ostrovsky and W. E. Skeith III. A survey of single-database private information retrieval: Techniques and applications. In T. Okamoto and X. Wang, editors, *Public Key Cryptography: 10th Inter-national Workshop on Practice and Theory in Public Key Cryptosystems, PKC 2007*, volume 4450 of *Lecture Notes in Computer Science*, pages 393–411. Springer-Verlag, 2007.

[17]. Y. Gertner, Y. Ishai, E. Kushilevitz, and T. Malkin. Protecting data privacy in private information retrieval schemes. *J. of Computer and System Sciences*, 60(3):592–629, 2000. Conference version in *Proc. of the 30th ACM Symp. on the Theory of Computing*, pages 151–160, 1998.

[18]. M. O. Rabin. How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981. Available online in the Cryptology ePrint Archive, Report 2005/187, eprint.iacr.org/2005/187.

Priti V. Bhagat received her BE (Computer Engineering) from Bapuraoji Deshmukh College of Engineering, Wardha in June 2007. Presently she is working as a Lecturer at Datta Meghe Institute of Engineering, Technology & Research, Sawangi (M), Wardha.

Rohit Singhal Presently he is working as Assistant Professor at IET, Alwar.