Extending Order Preserving Encryption for Multi-User Systems

Liangliang Xiao
University of Texas at Dallas
xll052000@utdallas.edu

I-Ling Yen
University of Texas at Dallas
ilyen@utdallas.edu

Dung T. Huynh
University of Texas at Dallas
huynh@utdallas.edu

Abstract. Several order preserving encryption (OPE) algorithms have been developed in the literature to support search on encrypted data. However, existing OPE schemes only consider a single encryption key, which is infeasible for a practical system with multiple users (implying that all users should have the single encryption key in order to encrypt or decrypt confidential data). In this paper, we develop the first protocols, DOPE and OE-DOPE, to support the use of OPE in multi-user systems. First, we introduce a group of key agents into the system and invent the DOPE protocol to enable "distributed encryption" to assure that the OPE encryption key is not known by any entity in the system. However, in DOPE, if a key agent is compromised, the share of the secret data that is sent to this key agent is compromised. To solve the problem, we developed a novel oblivious encryption (OE) protocol based on the oblivious transfer concept to deliver and encrypt the shares obliviously. Then, we integrate it with DOPE to obtain the OE-DOPE protocol. Security of OE-DOPE is further enhanced with additional techniques. Both DOPE and OE-DOPE can be used with any existing OPE algorithms while retaining all the advantages of OPE without requiring the users to share the single encryption key, making the OPE approach feasible in practical systems.

Keywords. Order preserving encryption, cloud computing, multi-user systems, chosen plaintext attack.

1 Introduction

In recent years, cloud computing has raised a lot of interests and many companies are looking into cloud solutions for their IT needs. Among various cloud paradigms, "database as a service (DaaS)" [3], [2], [6] has been actively investigated. In DaaS, data owners outsource their data as well as the access logic to cloud service providers, removing the burden of storage provision, backup, data management, access control, etc. from individual users and companies. With the many benefits of cloud computing and outsourcing [8], the security concerns emerge. For example, if the cloud service provider is compromised, the attacker can retrieve the sensitive data of the client companies. Or if there is a change in management of the cloud service provider, such as reorganization or buyout [9], the potential threat increases due to the additional exposure to multiple management personnel and the unestablished policies regarding the handling of critical information in such situations.

The security problems with the outsourced data can be solved if the sensitive data are encrypted. However, in many data-centric applications, it is necessary to perform search on data objects to find a group of data satisfying the given criteria. In order to facilitate efficient search on encrypted data, several order preserving encryption (OPE) algorithms have been proposed in the literatures [1] [4] [5] [7] [10]. OPE algorithms are generally deterministic symmetric-key based and ensure that the ciphertexts preserve the order of the plaintexts. Thus, search can be performed on encrypted data efficiently using conventional DBMS techniques, such as establishing the B+ tree on ciphertexts.

One limitation in all the existing OPE schemes is that there is no consideration of users. Consider a database hosted in the cloud and is accessed by many users with different access privileges. The critical data in the database are encrypted by an OPE scheme using a master encryption key. The server should not have the knowledge of the master key. When a user sends a query to the server, critical data in the query need to be encrypted and the returned results need to be decrypted. In conventional OPE schemes, it is implicitly assumed that the user knows the master key and, hence, is able to encrypt and decrypt the corresponding data. However, in practice, giving the master key to all the users is insecure. There is a significant probability for the server (or an adversary who compromises the server) to collude with one of the users and compromise the entire database. A potential solution is to use different encryption keys for different data. But it may not be easy to design an OPE to support comparisons and search on data that are encrypted using different keys.

In this paper, we develop protocols to support multi-user data-centric systems where OPE schemes are used to protect the sensitive data that need to be searched in encrypted form. Consider a database server (DB) hosting data encrypted by an OPE scheme using a master encryption key k. Assume that a user sends a query q to DB where q contains a confidential data x. Then, the DB processes q (and x) and send a response r to the user where r contains the confidential data y in encrypted form. Our goal is to develop an OPE protocol such that: (1) No user or any entity in the system has the knowledge of the master encryption key k. (2) The confidential data x and y are protected and no entity in the system besides the user is able to know what x or y is. (3) The protocol shall encrypt x such that when q reaches DB, x is already encrypted by the OPE scheme using the master key k. (4) The protocol shall allow the user to retrieve the plaintext y.

To achieve the above goal is challenging and there is no existing solution yet. Consider the request communication protocol for delivering the confidential data x to the DB. Without having any entity in the system knowing the OPE master encryption key k, how can a data be encrypted by k? A potential solution is to secret share the master key by a group of key agents and let the key agents "distributedly" encrypt the confidential data. At the same time, the user can secret share the confidential data x and pass the shares to the key agents. But how should the key and the data be shared and distributedly encrypted such that after the encrypted shares are assembled into the ciphertext, the order preserving property is preserved? Existing data sharing schemes cannot achieve this goal. We develop a novel digit based OPE (DOPE) protocol to realize the goal. In DOPE, we share the secret data x by mapping it into p "digits". Correspondingly, each of the p key agents holds a different encryption key. Each of the p digits is encrypted by a separate key agent with different key using an existing OPE scheme (any OPE scheme can be used with our protocol). The ciphers of the digits are sent to DB and integrated to the final ciphertext. Since the cipher of each digit is order-preserving, the integrated ciphertext is order-preserving.

The basic DOPE protocol discussed above has some security problems. A key agent can see the plain

digit, which reveals part of the confidential data x. A possible approach to cope with this problem is to use the secure two-party computation [14] algorithm, which involves oblivious transfer (OT) [12] and evaluation of garble circuits. However, both oblivious transfer and circuit evaluation incur a high computation and communication overhead. We invent a novel technique, oblivious encryption (OE, alternate to oblivious transfer), to enable the key agents to "obliviously" encrypt the "digits" without a high overhead. For each digit, the user mixes it with some randomly selected data to form a vector and sends the vector to the key agent to encrypt. At the same time, the user sends the "location" information directly to the DB so that the DB can correctly select the encrypted digit. If the vector size is n, then the probability for the key agent to correctly guess the digit is $1/n$. To achieve better security assurance, each digit can be further divided into t "micro-digits" and OE is applied to each of the micro-digits. Thus the probability for the key agent to successfully guess each complete digit becomes $1/n^t$. As can be seen, t controls the tradeoffs between security and performance of the protocol and should be chosen properly.

The scheme above still has security threats. An adversary can compromise a key agent to obtain the key for encrypting one digit of the confidential data. If the adversary also compromises the DB, then he can use the key to compromise the same digit of every data in the DB. To prevent such attack, we can use a chain of key agents to encrypt each digit. Thus, unless all the key agents in one chain are compromised, the key for the digit cannot be compromised. However, we still need to establish the protocol carefully to minimize the security risk. If the DB as well as the first key agent in the chain is compromised, then the adversary can obtain the "location" information from the DB to identify the plain digit the first key agent has received. Thus, we require each key agent in the chain randomly permutes the vector it receives (vector permutation). After permutations, the adversary still can restore the attack by linking the plaintext (retrieved from the first key agent) and the ciphertext (retrieved from DB) based on their orders. To cope with the attack, each key agent in the chain will substitute half elements in the vector (data mutation) to randomize the orders of the elements in the vector. With properly adjusted "location" information, the DB will be able to select the micro-digits correctly but the adversary can no longer use the location information and order information to identify the data on any key agent, unless it compromises all the key agents on one chain. We develop a complete solution, the OE-DOPE protocol, based on the basic-DOPE, OE, and the key agent chain with vector permutation and data mutation approaches.

The response communication protocol can be developed in a similar way as the request communication protocol. However, in many data centric applications, it is likely that the response would contain many more confidential data objects. The cost for transferring them using the reversed request communication protocol may be high. Thus, we use a simple, but very effective solution to achieve secure and efficient response delivery. We maintain two ciphers in the database, one encrypted using OPE with a master key and the other encrypted using a regular encryption scheme with different keys. Search can be performed on the ciphers from OPE. In a response, we only need to include the ciphertexts encrypted using the regular encryption algorithm, greatly saving the communication cost.

We prove that both the basic-DOPE and the OE-DOPE protocols assure one-wayness security, i.e., the adversary cannot compromise the confidential data fully, if the underlying OPE scheme we use (existing ones) has the one-wayness property. We also study the performance of our protocols and the results show that both protocols are reasonably efficient, as long as the underlying OPE scheme is reasonably efficient. Also, as expected, OE-DOPE incurs a higher overhead than basic-DOPE, but offers a better security.

The rest of the paper is organized as follows. Existing OPE algorithms are reviewed in Section 2. In Section 3, we introduce the system model, specify the problem, and discuss our approach. In Section 4, we construct the DOPE encryption scheme and the corresponding basic-DOPE request protocols for multi-user systems. In Section 5, the improved OE-DOPE protocol is introduced. Section 6 presents the performance study of the protocols. Section 7 concludes the paper. All proofs in this paper are relegated to the Appendix due to space limitation.

2 Related Work

There have been a number of OPE schemes proposed in the literature [1] [4] [5] [7] [10]. In [4], the OPE scheme first generates a sequence of random numbers and then, encrypts an integer x by adding the first x random numbers to it. The algorithm is inefficient and can be only used in a static system where no

new data can be inserted to the database. In [7], the OPE is constructed by using a mapping function composed of partition and identification functions. The partition function divides the range into multiple partitions, and the identification function assigns an identifier to each partition. Thus the encryption algorithm cannot compare all the plaintexts (e.g. the plaintexts in the same partition cannot be compared). In [1], the OPE is constructed in three steps: modeling the input and target distributions as linear splines, flattening the plaintext database into a flat (uniformly distributed) database, and transforming the flat database into the cipher database. Since the encryption algorithm needs to process the whole database to model the data distribution, it could be expensive for large databases. In [10], a sequence of strictly increasing polynomial functions are used to construct the OPE. The encryption of an integer x is the outcome of the iterative operations of those functions on x.

None of the OPE algorithms above are constructed using formal cryptographic basis. Thus, it is difficult to analyze the security of these OPE algorithms. In [5], a cryptography based OPE construction approach is proposed. It first defines the ideal OPE object whose encryption function is selected uniformly at random from the set of all strictly increasing functions. The ideal OPE object is not feasible, but can be used as the security goal for the real OPE scheme. In [5], a real OPE scheme is constructed, where a plaintext x is mapped to its cipher by a "binary-search-like" process in the cipher space with the searched points being mapped back to the plaintext space using the hypergeometric distribution. More specifically, let the plaintext domain be $[m_i]$ and the ciphertext range be $[n_i]$ in step i. For the middle point $y_i \in [n_i]$, it will be mapped to $x_i \in [m_i]$ with the probability $\binom{y_i}{x_i} \cdot \binom{n_i - y_i}{m_i - x_i} / \binom{n_i}{m_i}$. It has been proven in [5] that the real OPE scheme is computationally indistinguishable to the ideal OPE object. In other words, the security of the real OPE scheme is reduced to that of the ideal OPE object. In [13], it has been shown that the ideal OPE object achieves one-wayness security.

3 System Model and Overview

3.1 System Model

We consider a simplified system architecture which consists of a single server hosting a database and a set of users. Let DB denote the server and $U = \{U_j \mid j \geq 1\}$ denote the set of users. To ensure security, we also consider a set of key agents which mediates the communication between the users and the DB. Let KA denote the set of key agents.

For convenience, we assume that there are only numerical data in DB. Data of other types can be represented by numerical data easily. For each critical data item x, the DB maintains two ciphertexts $C_{OPE}(x)$ and $C_{CE}(x)$. $C_{OPE}(x)$ is encrypted using a specialized OPE scheme with a master key k. Note that the existing OPE scheme cannot be used directly to support multi-user systems and we develop a general approach to adapt any existing OPE scheme into a corresponding **digit based OPE** (DOPE) scheme. The cipher $C_{OPE}(x)$ of a data item x is encrypted using DOPE.

$C_{CE}(x)$ is encrypted using a classical encryption scheme (e.g. AES). The purpose of storing $C_{CE}(x)$ is to support efficient transmission of responses. For each data item x, a different data key dk_x is used to generate $C_{CE}(x)$. A user with access privilege to data item x will be granted key dk_x. In real implementation, the data items with the same access privileges can be grouped together into an access domain and only one key is needed for each access domain.

3.2 Definition of OPE

Let the plaintext domain be $\{0,1\}^\lambda = \{0, \ldots, 2^\lambda - 1\}$ and the ciphertext range be $\{0,1\}^\mu = \{0, \ldots, 2^\mu - 1\}$. The formal definition of OPE scheme is presented as follows.

Definition 1: Let $SE^{\lambda,\mu} = (K^{\lambda,\mu}, E^{\lambda,\mu}, D^{\lambda,\mu})$ be an OPE scheme, where $K^{\lambda,\mu}$: $\{0,1\}^* \rightarrow \{0,1\}^*$ is the key generation algorithm, $E^{\lambda,\mu}$: $\{0,1\}^\lambda \times \{0,1\}^* \rightarrow \{0,1\}^\mu$ is the encryption algorithm, and $D^{\lambda,\mu}$: $\{0,1\}^\mu \times \{0,1\}^* \rightarrow \{0,1\}^\lambda$ is the decryption algorithm. It satisfies that $D^{\lambda,\mu}(E^{\lambda,\mu}(x), k), k) = x$, $\forall\, x \in [m]$ and $E^{\lambda,\mu}(x, k) < E^{\lambda,\mu}(x', k)$, $\forall\, x < x'$.

Generally, the value of μ could impact the security of $E^{\lambda,\mu}$. But μ must be bounded by a polynomial of λ to keep the efficiency of $SE^{\lambda,\mu}$. Next, we define the one-wayness security of an OPE scheme as follows.

Definition 2: We say that the OPE scheme $SE^{\lambda,\mu} = (K^{\lambda,\mu}, E^{\lambda,\mu}, D^{\lambda,\mu})$ achieves one-wayness security if $\Pr[A(E^{\lambda,\mu}(x, k), \text{PCP}) = x] = \text{neg}(\lambda)$, where A is a PPT (probabilistic polynomial time) adversary, x is chosen uniformly randomly from the plaintext domain, $\text{PCP} = \{(x_i', E^{\lambda,\mu}(x_i', k)) \mid 1 \le i \le h\}$ is the set of h (h is bounded by a polynomial of λ) plaintext ciphertext pairs known by A and x_1', …, x_h' are also chosen uniformly randomly from the plaintext domain, and neg denotes a negligible function.

3.3 Problem Specification, Adversary Model, and Security Requirement

We construct a new OPE approach for multi-user systems. The approach includes a new OPE construction that is tightly coupled with a request communication protocol Q and a response communication protocol P.

In the request protocol Q, a user U_i issues a request (query) q to the DB, where q may contain some secret data that needs to be transmitted with q to the DB. For simplicity, assume that there is only one secret data item in q and let x denote that data item. Protocol Q should transfer q to DB while ensuring the correct and secure computation of $C_{OPE}(x)$ and $C_{CE}(x)$ in the request transmission process. (Note that U_i can encrypt x using dk_x and obtain $C_{CE}(x)$. But since U_i does not have the OPE master key k, it is not possible for U_i to compute $C_{OPE}(x)$. Thus, a set of key agents (KA) are introduced to perform the OPE encryption.)

In the response protocol P, the DB sends back the response r to the user. The response r may include a set of encrypted data objects $\{C_{CE}(y_1), C_{CE}(y_2), \ldots, C_{CE}(y_t)\}$ and/or $\{C_{OPE}(y_1), C_{OPE}(y_2), \ldots, C_{OPE}(y_t)\}$ (the protocol decides whether to send $C_{CE}(y_i)$ or $C_{OPE}(y_i)$ or both). Protocol P should ensure the secure delivery of r to U_i and that U_i can decrypt the information in r to obtain the query results $y_1, y_2, \ldots, y_t$.

Protocols Q and P have certain security requirements. The system entities, users, DB, and key agents, may collude to acquire additional information. We unify the possible collusions and construct a passive adversary A who tries to gain extra information by compromising some entities in the system. We assume that the key agents and DB are better protected than the users. Therefore we assume that the adversary cannot compromise all key agents simultaneously. Thus, we assume the adversary structure

$$\text{AS} = \{U_A \cup KA_A, U_A \cup KA_A \cup \{DB\} \mid U_A \subset U, KA_A \subset KA\},$$

where U_A is the set of compromised users and KA_A is the set of compromised key agents (note that U_A and KA_A could be empty). The system should ensure the security requirement $\Pr[A(View) = x] = \text{neg}(\lambda)$ is satisfied, where neg denotes a negligible function and *View* is the instance event randomly selected from the event space of what the adversary A can observe in the system by compromising entities in AS. Let $U(x)$ denote the set of users who can access the critical data x. Assume that none of the users in $U(x)$ are compromised by A. The security requirement can be interpreted as: for critical data x, if A does not compromise the users in $U(x)$, then the probability for A to retrieve x based on the information gathered from the compromised entities is negligible.

3.4 Our Approach

We design a simple and effective response protocol P to deliver the responses very efficiently. We simply include $C_{CE}(y_1), C_{CE}(y_2), \ldots C_{CE}(y_t)$ in r. The user should have access rights to $y_1, y_2, \ldots, y_t$ and, hence, should have the encryption keys $dk_{y_1}, dk_{y_2}, \ldots, dk_{y_t}$ to decrypt the data items in r. Consider the security of the system against adversary A (assume that A has not compromised the users in $U_{y_1}, \ldots, U_{y_t}$, where U_{y_j} is the set of users who can access y_j). Since the protocol only transfers $C_{CE}(y_1), C_{CE}(y_2), \ldots C_{CE}(y_t)$, A cannot get the encryption keys $dk_{y_1}, dk_{y_2}, \ldots, dk_{y_t}$ and cannot compromises $y_1, y_2, \ldots, y_t$. Note that the design of P is fully discussed here and will not be discussed further.

The request communication protocol Q cannot avoid the OPE encryption and are more complex. We design two protocols for Q: basic-DOPE and OE-DOPE, where OE-DOPE offers a better security protection to the secret data x in request q during the communication process. Basic-DOPE and OE-DOPE protocols are discussed in Sections 4 and 5, respectively. Both basic-DOPE and OE-DOPE protocols can be used with any existing OPE scheme.

4 The Basic DOPE Protocol

In the basic-DOPE protocol, we use p key agents, KA_0, …, KA_{p-1}. to encrypt confidential data x into $C_{OPE}(x)$. The critical data x is divided into p "digits". The i-th "digit" is sent to KA_i to be encrypted by the underlying OPE using a key k_i, $0 \le u < p$. The encrypted digits are sent to DB and integrated into the ciphertext $C_{OPE}(x)$.

The basic-DOPE and OE-DOPE protocols are coupled with the encryption algorithm DOPE. In Subsection 4.1, we present the construction of the DOPE encryption algorithm. Then we prove the correctness and analyze the security of the DOPE encryption scheme in Subsections 4.2 and 4.3, respectively. The basic-DOPE protocol is introduced in Subsection 4.4.

4.1 Construction of the DOPE Encryption Scheme

We construct the DOPE encryption scheme $SE_p^{\lambda,\mu} = (K_p^{\lambda,\mu}, E_p^{\lambda,\mu}, D_p^{\lambda,\mu})$ based on OPE scheme $SE^{\lambda',\mu'}$, where $\lambda = t\cdot\lambda'$ and $\mu = t\cdot\mu'$, as follows. The key generation algorithm $K_p^{\lambda,\mu}$ invokes $K^{\lambda',\mu'}$ to generate the OPE key k including p subkeys k_j, $0 \le j < p$. The process of the encryption algorithm $E_p^{\lambda,\mu}$ include: (1) representing the plaintext as p "digits" in base $2^{\lambda/p}$ number system, (2) encrypting the p "digits" by $E^{\lambda',\mu'}$, and (3) integrate the encrypted p "digits" back to a single value ciphertext in base $2^{\mu'}$ number system. Accordingly the decryption algorithm $D_p^{\lambda,\mu}$ uses the inverse process of $E_p^{\lambda,\mu}$ to decrypt the ciphertext. We describe the processes of $SE_p^{\lambda,\mu}$ in Figure 1.

$K_p^{\lambda,\mu}$: Invoke $K^{\lambda,\mu}$ to generate the set of keys $k = \{k_0, \ldots, k_{p-1}\}$.

$E_p^{\lambda,\mu}$: Input: plaintext x, $x \in \{0,1\}^{\lambda}$.
Output: ciphertext $C_{OPE}(x)$, $C_{OPE}(x) \in \{0,1\}^{\mu}$.
Let $\lambda' = \lambda/p$ and $\mu' = \mu/p$.
Express x in base $2^{\lambda'}$ number system, i.e., $x = \sum_{0\le j<p} x_j \cdot (2^{\lambda'})^j$, $0 \le x_j < 2^{\lambda'}$.
$C_{OPE}(x) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_j, k_j) \cdot (2^{\lambda'})^j$.

$D_p^{\lambda,\mu}$: Input: ciphertext $C_{OPE}(x)$, $C_{OPE}(x) \in \{0,1\}^{\mu}$.
Output: plaintext x, $x \in \{0,1\}^{\lambda}$.
Let $\lambda' = \lambda/t$ and $\mu' = \mu/t$.
Express $C_{OPE}(x)$ in base $2^{\mu'}$ number system, i.e., $C_{OPE}(x) = \sum_{0\le j<p} y_j \cdot (2^{\mu'})^j$.
Compute $x_j = D^{\lambda',\mu'}(y_j, k_j)$, $0 \le j < p$ and $x = \sum_{0\le j<p} x_j \cdot (2^{\lambda'})^j$.

Figure 1. DOPE scheme $SE_p^{\lambda,\mu}(K_p^{\lambda,\mu}, E_p^{\lambda,\mu}, D_p^{\lambda,\mu})$.

4.2 Correctness of $SE_p^{\lambda,\mu}$

We analyze the correctness of $SE_p^{\lambda,\mu}$ in Proposition 1.

Proposition 1: $SE_p^{\lambda,\mu}$ is correct, i.e. $D_p^{\lambda,\mu}(E_p^{\lambda,\mu}(x, k), k) = x$ and $E_p^{\lambda,\mu}$ is an OPE algorithm.

4.3 Security of $SE_p^{\lambda,\mu}$

According to the construction in Figure 1, $E_p^{\lambda,\mu}(x, k) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_j, k_j) \cdot (2^{\lambda'})^j$. The security of $E_p^{\lambda,\mu}(x, k)$ can be reduced to the security of $E^{\lambda',\mu'}(x_j, k_j)$ where $\lambda' = \lambda/p$, $\mu' = \mu/p$, and $0 \le j < p$. According to [13], there exists OPE scheme $SE^{\lambda',\mu'}$ to achieve the one-wayness security where (1) $\mu' \ge 3\lambda'$ and (2) h (the number of plaintext ciphertext pairs known by the adversary) are bounded by a polynomial of λ'. Hence the values of μ and p are critical to the security of $SE_p^{\lambda,\mu}$. We set $\mu \ge 3\lambda$ to satisfy (1), and set $p = O(\lambda^c)$ to satisfy (2), where $0 < c < 1$ is a constant. The one-wayness security of $SE_p^{\lambda,\mu}$ is proven Theorem 1.

Theorem 1: Assume that there is an OPE scheme $SE^{\lambda',\mu'} = (K^{\lambda',\mu'}, E^{\lambda',\mu'}, D^{\lambda',\mu'})$ achieves one-wayness security for $\mu' \ge 3\lambda'$. Consider the DOPE scheme $SE_p^{\lambda,\mu}$ constructed based on $SE^{\lambda',\mu'}$ in Figure 1. Then $SE_p^{\lambda,\mu}$ also achieves the one-wayness security for $\mu \ge 3\lambda$ and $p = O(\lambda^c)$, $0 < c < 1$, even if the adversary knows a proper subset of keys in k. Specifically, $\Pr[A(E_p^{\lambda,\mu}(x, k), \text{PCP}, k') = x] = \text{neg}(\lambda)$, for $\mu \ge 3\lambda$, where PCP = $\{(x_i', E_p^{\lambda,\mu}(x_i', k)) \mid 1 \le i \le h\}$, and $k' \subset k = \{k_0, \ldots, k_{p-1}\}$.

4.4 The Basic DOPE Communication Protocol

Let KA_i, $0 \le i \le p-1$ be the p key agents. Without loss of generality, we assume that $p = O(1)$, i.e. there are a constant number of key agents. Let $SE^{\lambda',\mu'} = (K^{\lambda',\mu'}, E^{\lambda',\mu'}, D^{\lambda',\mu'})$ be the underlying OPE scheme, where $\mu \ge 3\lambda$, $\lambda' = \lambda/p$ and $\mu' = \mu/p$. We assume that at the system initialization time, some trusted party uses $K^{\lambda,\mu}$ to generate $k = (k_0, \ldots, k_{p-1})$, and distribute k_i to KA_i, $0 \le i \le p-1$. The basic-DOPE protocol realizes DOPE encryption scheme through KA_i, $0 \le i \le p-1$ and its pseudo code is presented in Figure 2. Figure 3 (page 12) shows the structure and message flow of the basic-DOPE protocol.

Let plaintext $x \in \{0,1\}^{\lambda}$, $\lambda' = \lambda/p$, $\mu' = \mu/p$, and $p = O(1)$.

(1) The user U_i express x in base $2^{\lambda'}$, i.e., $x = \sum_{0 \le j < p} x_j \cdot (2^{\lambda'})^j$, $0 \le x_j < 2^{\lambda'}$. U_i sends x_j to KA_j, $0 \le j \le p-1$.
(2) For $0 \le j \le p-1$, KA_j computes $y_j = E^{\lambda',\mu'}(x_j, k_j)$ and sends y_j to DB.
(3) DB combines $C_{OPE}(x) = \sum_{0 \le j < p} y_j \cdot (2^{\mu'})^j$.

Figure 2. The pseudo code for the basic-DOPE protocol.

The efficiency, correctness (i.e., DOPE encryption result is the same as the ciphertext $C_{OPE}(x)$), and security proof of the basic-DOPE protocol are given in Theorem 2.

Theorem 2: The basic-DOPE protocol is efficient and correct, and achieves the one-wayness security against the adversary structure $AS = \{U_A \cup KA_A, U_A \cup KA_A \cup \{DB\} \mid U_A \subset CU, KA_A \subset KA\}$.

5 The OE-DOPE Protocol

5.1 Security Issue in the Basic DOPE Protocol

Consider the following two attacks against the basic-DOPE protocol.

(1) The adversary A compromises the key agent KA_u. Then A can view the "digit" x_u of the plaintext x in the process of the basic DOPE protocol.
(2) The adversary A compromises DB and the key agent KA_u for some $0 \le u < p$. Then, for any ciphertext $C_{OPE}(x) = \sum_{0 \le u < p} E^{\lambda',\mu'}(x_u, k_u) \cdot (2^{\mu'})^u$ stored on DB, A can use the key k_u retrieved from KA_u to compute $x_u = D^{\lambda',\mu}(E^{\lambda',\mu'}(x_u, k_u), k_u)$.

In both situations, the adversary A retrieves the partial information x_u of x, which implies that $\lambda' = \lambda/p$ bits of the critical data item are leaked. However, since we assume that A cannot compromise all key agents, A cannot retrieve the whole plaintext x. That is why the basic-DOPE protocol can achieve one-wayness security. But revealing λ/p bits of some critical data items may be unacceptable in many applications. Hence, it is desirable to enhance the security of the request communication protocol.

5.2 Oblivious Encryption

The attack in (2) is relatively easy to prevent. We substitute KA_u by a chain of key agents $KA_{u,0}, \ldots, KA_{u,q-1}$. The key k_u is also split into $k_{u,0}, \ldots, k_{u,q-1}$ and distributed to $KA_{u,0}, \ldots, KA_{u,q-1}$, for all u. Critical data x_u is encrypted through the chain $KA_{u,0}, \ldots, KA_{u,q-1}$ by the OPE $E^{\lambda_0,\mu_0}, \ldots, E^{\lambda_{q-1},\mu_{q-1}}$. The resulting ciphertexts, after encrypted by the chain of KAs, is order preserving because the composition of OPEs is still an OPE. Now the adversary cannot retrieve x_j from $C_{OPE}(x)$ unless it retrieves q keys $k_{u,0}, \ldots, k_{u,q-1}$ by compromising the key agents $KA_{u,0}, \ldots, KA_{u,q-1}$.

In principle, the attack in (1) can be prevented by secure computation, where the user has the input x_u and the key agent has the input k_u. The user and the key agent can securely compute the function $E^{\lambda',\mu'}$, $\lambda' = \lambda/p$ and $\mu' = \mu/p$, by any two party computation protocol. However, existing two party computation protocols have high overhead. Therefore we develop the technique of OE (oblivious encryption) to enable the key agent to encrypt x_u without knowing the actual value of x_u (i.e., the probability for the key agent to know x_u is negligible). In OE, x_u is further expressed in the base $2^{\lambda''}$ number system, where $\lambda'' = \lambda'/t$ and $t = \lambda'^c$, $0 < c < 1$. Let $x_{u,0}, \ldots, x_{u,t-1}$ be the t "micro-digits" of x_u. Then, in the "micro-digit" domain $\{0,1\}^{\lambda''}$ of $x_{u,v}$, the user sends a vector, including $x_{u,v}$ and $\lambda'' - 1$ random plaintexts to the key agent KA_u, $0 \le v < t$.

KA_u encrypts all of the elements in the vector (KA_u does not know which one is $x_{u,v}$) and sends the encrypted vector to the DB. At the same time, the user sends the location information $l_{u,v}$ of $x_{u,v}$ in the λ'' random plaintexts to the DB so that the DB can identify the encrypted $x_{u,v}$ and integrates them into $C_{OPE}(x)$. By further dividing the digits into t micro-digits, the probability for KA_u to successfully guess x_u drops to $1/(\lambda'')^t$, which is a negligible function of λ.

5.3 Vector Permutation and Data Mutation

The protocol above has a new security issue. If both $KA_{u,0}$ (the first key agent in the chain) and the DB are compromised, then the location information (sent from the user to DB) can be used to identify $x_{u,v}$ in the λ'' random plaintexts (sent from the user to $KA_{u,0}$). Consequently, $x_u = \sum_{0 \le v < t} x_{u,v} \cdot (2^{\lambda''})^v$ can be derived. To cope with this attack, the key agent $KA_{u,j}$ permutes the vector (original or encrypted) by a permutation $\pi_{u,v,j}$ (randomly generated by the user) before sending them to the next key agent $KA_{u,j+1}$. Thus, $l'_{u,v} = \pi_{u,v,q-1} \circ \ldots \circ \pi_{u,v,0}(l_{u,v})$ (instead of $l_{u,v}$) will be the location information the user sends to the DB. However, using permutations alone cannot guarantee the security because the encryption $E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$ preserves the order. Thus A can still correctly link the λ'' plaintexts (retrieved from $KA_{u,0}$) to the λ'' ciphertexts (retrieved from DB) according to their orders and, hence restore the above attack. To prevent the adversary from using orders to establish the links, each key agent $KA_{u,j}$, $1 \le j < q$, will substitute half elements in the vector (the set of the locations will be provided by the user) with new random values to change the order of the ciphertext of $x_{u,v}$ in the vector. Consequently, the adversary cannot use the location information and order information to indentify $x_{u,v}$ in the λ'' random plaintexts (sent from the user to $KA_{u,0}$).

We now construct the OE-DOPE protocol. Let $KA = \{KA_{u,j} \mid 0 \le u < p, 0 \le j < q\}$, which is logically a KA grid of dimension $p*q$. We assume that there are a fixed number of key agents and, hence, $p, q = O(1)$. Let $\{0,1\}^{\lambda}$ be the plaintext domain, $\lambda' = \lambda/p$, and $\lambda'' = \lambda'/t$ where $t = (\lambda')^c$ for some constant $0 < c < 1$. For $0 \le j < q$, let $SE^{\lambda_j,\mu_j} = \left(K^{\lambda_j,\mu_j}, E^{\lambda_j,\mu_j}, D^{\lambda_j,\mu_j}\right)$ be the OPE schemes satisfying $\lambda_0 = \lambda''$, $\mu_j = \lambda_{j+1}$ for $0 \le j < q-1$, and $\mu_j = 3\lambda_j$ for $0 \le j < q$. Therefore, $SE^{\lambda'',\mu''} = (K^{\lambda'',\mu''}, E^{\lambda'',\mu''}, D^{\lambda'',\mu''})$ is also an OPE scheme, where $E^{\lambda'',\mu''} = E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$. Since $\mu_j = 3\lambda_j$ for $0 \le j < q$, we have $\mu'' = 3^q \cdot \lambda'' > 3\lambda''$. Also since q is a constant, we have $\mu'' = O(\lambda'')$ and, hence, $SE^{\lambda'',\mu''}$ is an efficient OPE scheme. We assume that at the system initialization time, some trusted party has used K^{λ_j,μ_j} to generate the keys $k_j = (k_{0,j}, \ldots, k_{p-1,j})$, and distributed $k_{u,j}$ to $KA_{u,j}$, $0 \le u < p$ and $0 \le j < q$. The pseudo code of OE-DOPE protocol is shown in Figure 4. The structure and message flow of the OE-DOPE protocol is illustrated in Figure 5 (page 12).

Let plaintext $x \in \{0,1\}^{\lambda}$, $\lambda' = \lambda/p$, and $\lambda'' = \lambda'/t$ where $t = (\lambda')^c$ for some constant $0 < c < 1$.

The user U_i:

(1) Expresses x in base $2^{\lambda'}$, i.e., $x = \sum_{0 \le u < p} x_u \cdot (2^{\lambda'})^u$, $0 \le x_u < 2^{\lambda'}$, further expresses x_u in base $2^{\lambda''}$, i.e., $x_u = \sum_{0 \le v < t} x_{u,v} \cdot (2^{\lambda''})^v$, $0 \le x_{u,v} < 2^{\lambda''}$.

(2) For each $x_{u,v}$, randomly selects $\lambda'' - 1$ distinct elements in $\{0,1\}^{\lambda''}$. Let the elements together with $x_{u,v}$ be $w_{u,v,0} < \ldots < w_{u,v,l_{u,v}} = x_{u,v} < \ldots < w_{u,v,\lambda''-1}$.

(3) Randomly generates the $\pi_{u,v,j}$: $\{0, \ldots, \lambda''-1\} \to \{0, \ldots, \lambda''-1\}$, $0 \le u < p$, $0 \le v < t$, and $0 \le j < q$.

(4) Randomly selects the set of locations $L_{u,v,j}$ satisfying $\pi_{u,v,j-1} \circ \ldots \circ \pi_{u,v,0}(l_{u,v}) \notin L_{u,v,j} \subset \{0, \ldots, \lambda''-1\}$, $|L_{u,v,j}| = \lambda''/2$, $0 \le u < p$, $0 \le v < t$, and $1 \le j < q$.

(5) Sends $W_{u,v} = (w_{u,v,0}, \ldots, w_{u,v,\lambda''-1})$ to $KA_{u,0}$, , $0 \le u < p$ and $0 \le v < t$
sends $\pi_{u,v,0}$ to $KA_{u,0}$ and $(\pi_{u,v,j}, L_{u,j})$ to $KA_{u,j}$, $0 \le u < p$, $0 \le v < t$, and $1 \le j < q$
sends $l'_{u,v} = \pi_{u,v,q-1} \circ \ldots \circ \pi_{u,v,0}(l_{u,v})$ to DB, $0 \le u < p$ and $0 \le v < t$.

The key agent $KA_{u,v}$:

(1) Let $W^{(0)}_{u,v} = W_{u,v}$. For $0 \le j < q-1$, $KA_{u,j}$ encrypts every elements in $W^{(j)}_{u,v}$ by E^{λ_j,μ_j} using the key $k_{u,j}$ except those at the locations in $L_{u,v,j}$, uses random values as the encryption results for the elements at the locations in $L_{u,v,j}$, permutes the order of the encryptions by $\pi_{u,v,j}$, and sends the result $W^{(j+1)}_{u,v}$ to $KA_{u,j+1}$.

(2) $KA_{u,q-1}$ encrypts every elements in $W^{(q-1)}_{u,v}$, permutes the order of the encryptions by $\pi_{u,v,q-1}$, and sends the result $W^{(q)}_{u,v}$ to DB.

(3) Selects the $l'_{u,v}$-th element in $W^{(q)}_{u,v}$ to compute $C_{OPE}(x) = \sum_{0 \le u < p} (\sum_{0 \le v < t} W^{(q)}_{u,v}[l'_{u,v}] \cdot 2^{\mu''}) \cdot 2^{\mu'}$.

Figure 4. The OE-DOPE protocol.

As shown in Figure 4, the user U_i expresses the plaintext x in base $2^{\lambda'}$ number system and further expresses the "digit" x_u in base $2^{\lambda''}$ number system. For the "micro-digit" $x_{u,v}$, the vector $W_{u,v} = (w_{u,v,0}, ..., w_{u,v,\lambda''-1})$ is created such that $x_{u,v}$ is in $W_{u,v}$ at a random position $l_{u,v}$. The user also randomly generates the permutations $\pi_{u,v,j}$ and the set of location $L_{u,v,j}$, $|L_{u,v,j}| = \lambda''/2$. $(\pi_{u,v,j}, L_{u,v,j})$ is sent to KA_u, $0 \le u < p$, $0 \le v < t$, and $0 \le j < q$, and $l'_{u,v} = \pi_{u,v,q-1} \circ \ldots \circ \pi_{u,v,0}(l_{u,v})$ is sent to DB, $0 \le u < p$ and $0 \le v < t$. Then $W_{u,v}$ is sent to $KA_{u,0}$ so that the elements in $W_{u,v}$ are encrypted, subsituted, and permutated through $KA_{u,0}$ to $KA_{u,q-1}$. Finally, the DB identifies the encryptions of $x_{u,v}$ according to $l'_{u,v}$, and integrates them to get $C_{OPE}(x)$. According to the formula in (7) in Figure 4, $C_{OPE}(x)$ has the following encryption structure: $C_{OPE}(x)$ is the ciphertext encrypted by the DOPE scheme $SE_p^{\lambda,\mu}$. $SE_p^{\lambda,\mu}$ is based on the underlying DOPE scheme $SE_t^{\lambda',\mu'}$. And $SE_t^{\lambda',\mu'}$ is based on the underlying OPE scheme $SE^{\lambda'',\mu''}$, where $E^{\lambda'',\mu''} = E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$.

We now prove the efficiency, correctness (i.e., the ciphertext $C_{OPE}(x)$ preserves the order of the plaintexts), and the security of the OE-DOPE protocol in the following theorem.

Theorem 3: The OE-DOPE protocol is efficient and correct. Furthermore, consider the adversary structure AS. Suppose that a user $U_i \notin U_A$ sends x to DB. If the adversary A does not compromise all the key agents $KA_{u,0}, \ldots, KA_{u,q-1}$, simultaneously, then the probability for A to retrieve the "digit" x_u of x is negligible, where $x = \sum_{0 \le u < p} x_u \cdot (2^{\lambda'})^u$.

Note that if the adversary A compromises less than q key agents, then it implies that A does not compromise the key agents $KA_{u,0}, \ldots, KA_{u,q-1}$ simultaneously for any $0 \le u < p$. According to Theorem 3, the probability for A to retrieve any "digit" x_u of x is negligible. We summarize the conclusion in the following corollary.

Corollary 1: Consider the adversary structure AS. Suppose that a user $U_i \notin U_A$ sends x to DB. If the adversary A compromises less that q key agents, then the probability for A to retrieve every "digit" of x is negligible.

6 Performance Study

We study the performance of the protocols basic-DOPE and OE-DOPE using different underlying OPE schemes. To the best of our knowledge, five OPE schemes have been proposed in the literatures [4] [7] [1] [5] [10]. None of them, except for the OPE algorithm proposed in [10], have cryptographic security proofs. As discussed in Section 2, the OPE algorithm proposed in [4] can only be used in a static system where no new data can be inserted to the database. The algorithm given in [7] is not a full solution because it cannot compare all the plaintexts. The OPE algorithm developed in [1] needs to process the whole database to model the data distribution. Thus, we only consider the OPE algorithms proposed in [5] and [10] in our experimental study.

The performance of the OPE schemes has never been analyzed in the literature. Thus, we first study the performance of the Hyper and Poly OPE schemes. We randomly generate a polynomial with degree 10 for the Poly scheme. The domain of the plaintext is $\{0,1\}^{\lambda}$ and we choose $\lambda = \{8, 16, 32, 64, 96, 128, 256, 512, 1024\}$ and $c = 0.5$. The ciphertext range is $\{0,1\}^{\mu}$ and we consider $\mu = 3\lambda$ for Hyper OPE scheme and $\mu = 10\lambda$ for Poly OPE scheme. The experiments are run on a 2.50GHz Intel Core 2 Duo Processor. Table 1 shows the execution time in milliseconds for Hyper and Poly to encrypt a single critical data item of λ bits.

λ	Hyper OPE	Poly OPE
8	20.37022	0.0003
16	4965.81013	0.0007
24	520073.44982	0.0008
32		0.0010
64		0.0027
128		0.0077
256		0.0261
512		0.0929
1024		0.3710

Table 1. Performance of Hyper and Poly OPE schemes.

As can be seen, Hyper OPE scheme is far more expensive than Poly OPE scheme. In Hyper OPE

scheme, the process for realizing the hypergeometric random variable is very time consuming. In Poly OPE scheme, it evaluates the randomly selected polynomial with the plaintext as input, which is much less time consuming than Hyer OPE. But Hyper OPE scheme can be proven to achieve one-wayness security, while there is no security proof for Poly OPE scheme.

Now we compare the performance of the basic-DOPE and the OE-DOPE protocols integrated with Hyper and Poly OPE schemes. In the two request communication protocols, the request is sent from the user to the KA and then to the DB. To factor in the communication latencies between the system entities, we allocate the user, the key agents and the DB to different PlanetLab [11] computers and measure the communication latencies between them. The user is in Dallas and the DB is in Los Angeles. The basic-DOPE protocol requires p key agents and we choose $p = 4$ (make λ divisible by p). The four key agents are allocated to Phoenix (Arizona), Salt Lake City (Utah), Carson City (Nevada), and Eugene (Oregon). The OE-DOPE protocol requires $p*q$ key agents. We use the same p value and consider $q = 2$. The 4 additional key agents (out of 8) are allocated in the same city as the other key agents in the same chain. The request message without the critical data is of size 170 bytes (based on the average of the sizes of some common queries). The critical data size is λ bits.

For comparison purpose, we also consider the "No Encryption" (NE) request communication protocol, the Hyper request communication protocol, and the Poly request communication protocol. In NE, the user directly sends the query (with the critical data in plaintext) to DB. In Hyper/Poly, the user knows the master OPE key and encrypts the confidential data using the Hyper/Poly OPE scheme with the master key and send the ciphertext directly to DB. Table 2 shows the performance comparisons (in milliseconds) of the NE, Hyper, basic-DOPE and OE-DOPE protocols using the Hyper OPE scheme. Table 3 shows the performance comparisons (in milliseconds) of the NE, Poly, and the basic-DOPE and OE-DOPE protocols using the Poly OPE scheme.

λ	NE	Hyper	basic-DOPE + Hyper	OE-DOPE + Hyper
8	85.87	106.24	506.06	7718.90
16	85.92	5051.73	1525.28	317766.88
32	86.03		20537.11	9.19E+07
64	86.23		4965977.56	
96	86.44		5.2E+08	

Table 2. Comparisons of the basic-DOPE protocol and OE-DOPE protocol with Hyper OPE scheme.

λ	NE	Poly	basic-DOPE + Poly	OE-DOPE + Poly
8	85.87	85.87	166.62	194.35
16	85.92	85.92	167.03	200.88
32	86.03	86.03	167.83	214.18
64	86.23	86.23	169.32	239.34
96	86.44	86.44	170.73	262.81
128	86.64	86.64	172.05	285.09
256	87.43	87.45	176.79	366.91
512	88.92	89.01	184.65	513.02
1024	91.62	91.99	197.23	786.72

Table 3. Comparisons of the basic-DOPE protocol and OE-DOPE protocol with Poly OPE scheme.

As shown in Tables 2 and 3, the OE-DOPE protocol is more expensive than the basic-DOPE protocol. This is because: (1) There are extra random data to be transmitted and encrypted in the OE-DOPE protocol to facilitate oblivious encryption. (2) After encryption by one key agent, the ciphertext grows. The longer the chain, the larger the size of the ciphertext becomes (For example, Poly OPE scheme takes 0.0028 milliseconds to encrypt 64 bits plaintext. But after encryption, the size of the plaintext becomes 640 bits. It then takes 0.16 milliseconds to encrypt the cipher of 640 bits and will generate a new cipher of 6400 bits.) But the OE-DOPE protocol achieves a higher security level than the basic-DOPE protocol. Compare with the baseline NE protocol, the basic-DOPE protocol using Hyper OPE scheme is over 200 times slower for $\lambda = 32$, and the basic-DOPE protocol using Poly OPE scheme is at most 2 times slower for any λ, $8 \leq \lambda \leq 1024$. The Poly protocol has a similar performance to that of the NE protocol since the encryption time of Poly OPE is very small for $8 \leq \lambda \leq 1024$.

Note that in OE-DOPE, the key agents are logically deployed in p rows and q columns. We study the influence of q on the performance of the OE-DOPE protocol, We set $p = 4$, and vary q from 2 to 4. The $p \cdot q$

key agents are physically allocated to Phoenix (Arizona), Salt Lake City (Utah), Carson City (Nevada), and Eugene (Oregon). Key agents in the same row are allocated in the same city. The user is still in Dallas and the DB is still in Los Angeles. The request message without the critical data is of size 170 bytes and the critical data is of size λ bits (we vary λ in the experiments). The performance results (in milliseconds) of the OE-DOPE protocol using Hyper OPE scheme (denoted by OHyper) and Poly OPE scheme (denoted by OPoly) are given in Table 4.

λ	OHyper q=2	OHyper q=3	OPoly q=2	OPoly q=3	OPoly q=4
8	7718.9	48776.23	194.35	238.30	402.90
16	317766.88		200.88	266.33	516.80
32	9.19E+07		214.18	316.26	720.34
64			239.34	404.95	1183.25
96			262.81	488.20	1960.84
128			285.09	580.90	3091.75
256			366.91	1053.45	11920.64
512			513.02	1849.54	57943.90
1024			786.72	5411.71	313460.75

Table 4. Performances of the OE-DOPE protocol using Hyper OPE scheme/Poly OPE scheme for different q.

As can be seen, the execution time for the OE-DOPE protocol with the Hyper OPE or Poly OPE scheme increases with increasing q. The impact of q is more significant for larger λ. With q = 3 and for a 32-bit critical data, the OE-DOPE protocol takes 0.7 seconds, which is an acceptable performance.

7 Conclusion

In this paper, we first identify the problem in existing OPE schemes, namely, none of the OPE algorithms can be used in multi-user systems where users are not supposed to have the same access rights to the critical data. Then, we develop two novel protocols to extend existing OPE schemes for multi-user data-centric systems. Users can encrypt their secret data using our OPE protocols without knowing the OPE encryption key. Also, we develop a simple and effective response protocol to allow efficient delivery of secret data in the response to the user. Our protocols are general and can be used with any OPE scheme. We have proven their correctness and security. We have also studied their performance and the results show that the protocols have a fairly reasonable overhead when the underlying OPE scheme is relatively efficient.

Our future research will focus on developing efficient OPE algorithms that can be cryptographically proven to be secure. We will investigate methods to improve the performance of the OPE scheme given in [5]. We will also study the problems of the OPE scheme proposed in [10] and enhance it for to achieve provable one-wayness security.

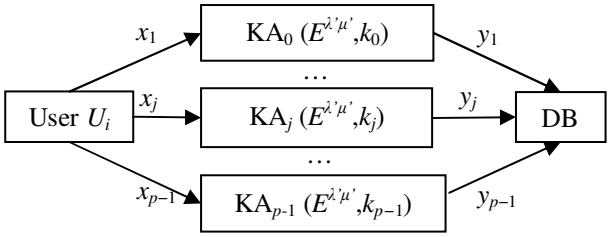

Figure 3. The structure and message flow of the basic-DOPE protocol.

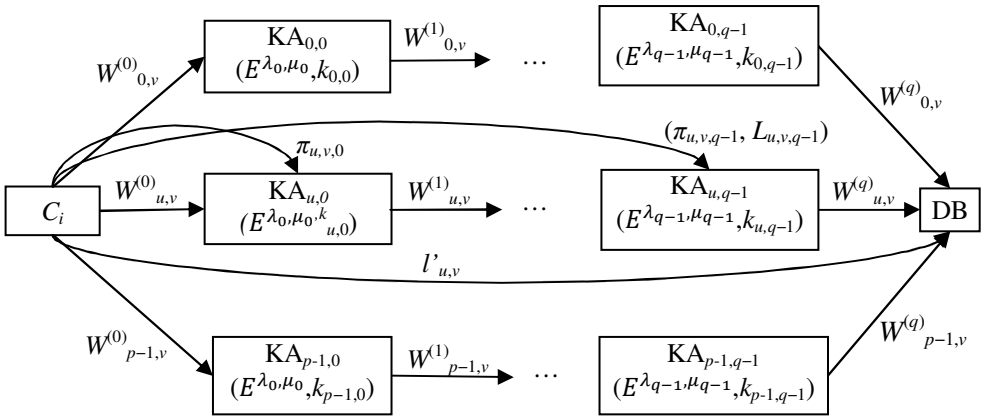

Figure 5. Message Flow of the OE-DOPE protocol.

8 References

[1] R. Agrawal, J. Kiernan, R. Stikant, and Y. Xu, "Order-preserving encryption for numeric data", In SIGMOD'04, pp 563-574, ACM, 2004.

[2] A. Swaminathan, Y. Mao, G.M. Su, H. Gou, A.L. Varna, S. He, M. Wu, D. W. Oard, "Confidentiality-preserving rank-ordered search", StorageSS 2007, pp 7-12, 2007.

[3] G. Amanatidis, A. Boldyreva and A. O'Neill, "Provably-Secure Schemes for Basic Query Support in Outsourced Databases", Working Conference on Data and Applications Security 2007 Proceedings, Lecture Notes in Computer Science, Vol. 4602, pp 14-30, 2007.

[4] G. Bebek, "Anti-tamper database research: Inference control techniques", Technical Report EECS 433 Final Report, Case Western Reserve University, 2002.

[5] A. Boldyreva, N. Chenette, Y. Lee, A. O'Neill, "Order-Preserving Symmetric Encryption", Advances in Cryptology - Eurocrypt 2009, 2009.

[6] S. Evdokimov, O. G¨unther, "Encryption Techniques for Secure Database Outsourcing", Lecture Notes in Computer Science, Vol 4734, pp 327-342, 2008.

[7] H. Hacig¨um¨us¸, B.R. Iyer, C. Li, and S. Mehrotra, "Executing SQL over encrypted data in the database-service-provider model", Proc. of the ACM SIGMOD Conf. on Management of Data, Madison,Wisconsin, 2002.

[8] ComputerWorld. J.P. Morgan signs outsourcing deal with IBM. Dec. 30, 2002.

[9] Oracle: Oracle Buys PeopleSoft, http://www.oracle.com/corporate/press/2004_dec/acqisition.html.

[10] G. Ozsoyoglu, D. Singer, S.S. Chung, "Anti-tamper databases: Querying encrypted databases", Proc. of the 17th Annual IFIP WG 11.3 Working Conference on Database and Applications Security, Estes Park, Colorado, 2003.

[11] PlanetLab, http://www.planet-lab.org.

[12] M.O. Rabin, "How to exchange secrets by oblivious transfer", Technical Report TR-81, Aiken Computation Laboratory, Harvard University, 1981.

[13] L. Xiao, O. Bastani, I. Yen, "Security Analysis for Order Preserving Encryption Schemes", Tech Report UTDCS-01-12, 2012, http://utdallas.edu/~xll052000/OPEproof-TR3.pdf.

[14] A.C. Yao, "Protocols for secure computations", Foundations of Computer Science, pp. 160–164, 1982.

9 Appendix

Proposition 1: $SE_p^{\lambda,\mu}$ is correct, i.e. $D_p^{\lambda,\mu}(E_p^{\lambda,\mu}(x, k), k) = x$ and $E_p^{\lambda,\mu}$ is an OPE algorithm.

Proof. First we prove that $D_p^{\lambda,\mu}(E_p^{\lambda,\mu}(x, k), k) = x$. Let $x = \sum_{0\le j<p} x_j \cdot (2^{\lambda'})^j$ and $E_p^{\lambda,\mu}(x, k) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_j, k_j) \cdot (2^{\mu'})^j$. Then according the process shown in Figure 1,

$$D_p^{\lambda,\mu}(E_p^{\lambda,\mu}(x, k), k) = \sum_{0\le j<p} D^{\lambda',\mu'}(y_j,k_j) \cdot (2^{\lambda'})^j$$
$$= \sum_{0\le j<p} D^{\lambda',\mu'}(E^{\lambda',\mu'}(x_j, k_j), k_j) \cdot (2^{\lambda'})^j = \sum_{0\le j<p} x_j \cdot (2^{\lambda'})^j = x.$$

Now we prove that $E_p^{\lambda,\mu}$ is an OPE algorithm. For $x_1 = \sum_{0\le j<p} x_{1,j} \cdot (2^{\lambda'})^j$ and $x_2 = \sum_{0\le j<p} x_{2,j} \cdot (2^{\lambda'})^j$, we consider three situations.

(i) $x_1 < x_2$. Then $\exists\, j_0$ s.t. $x_{1,j} = x_{2,j}$ for $j > j_0$ and $x_{1,j} < x_{2,j}$ for $j = j_0$. Therefore
$E^{\lambda',\mu'}(x_{1,j}, k_j) = E^{\lambda',\mu'}(x_{2,j}, k_j)$ for $j > j_0$ and $E^{\lambda',\mu'}(x_{1,j}, k_j) < E^{\lambda',\mu'}(x_{2,j}, k_j)$ for $j = j_0$.
Hence $E_p^{\lambda,\mu}(x_1, k) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_{1,j}, k_j) \cdot (2^{\mu'})^j < \sum_{0\le j<p} E^{\lambda',\mu'}(x_{2,j}, k_j) \cdot (2^{\mu'})^j = E_p^{\lambda,\mu}(x_2, k)$.
(ii) $x_1 = x_2$. It can be proven that $E_p^{\lambda,\mu}(x_1, k) = E_p^{\lambda,\mu}(x_2, k)$ analogously to (i).
(iii) $x_1 > x_2$. It can be proven that $E_p^{\lambda,\mu}(x_1, k) > E_p^{\lambda,\mu}(x_2, k)$ analogously to (i).
According to (i) (ii) (iii), $E_p^{\lambda,\mu}$ is an OPE algorithm.

Theorem 1: Assume that there is an OPE scheme $SE^{\lambda',\mu'} = (K^{\lambda',\mu'}, E^{\lambda',\mu'}, D^{\lambda',\mu'})$ achieves one-wayness security for $\mu' \ge 3\lambda'$. Consider the DOPE scheme $SE_p^{\lambda,\mu}$ constructed based on $SE^{\lambda',\mu'}$ in Figure 1. Then $SE_p^{\lambda,\mu}$ also achieves the one-wayness security for $\mu \ge 3\lambda$ and $p = O(\lambda^c)$, $0 < c < 1$, even if the adversary knows a proper subset of keys in k. Specifically,

$$\Pr[A(E_p^{\lambda,\mu}(x, k), \text{PCP}, k') = x] = \text{neg}(\lambda)$$

for $\mu \ge 3\lambda$, where $\text{PCP} = \{(x_i', E_p^{\lambda,\mu}(x_i', k)) \mid 1 \le i \le h\}$, and $k' \subset k = \{k_0, \ldots, k_{p-1}\}$.

Proof. We reduce what the adversary view in the plaintext domain $\{0,1\}^{\lambda}$ and ciphertext range $\{0,1\}^{\mu}$ to $\{0,1\}^{\lambda'}$ and $\{0,1\}^{\mu'}$, where $\lambda' = \lambda/p$ and $\mu' = \mu/p$. Suppose that

$$x = \sum_{0\le j<p} x_j \cdot (2^{\lambda'})^j.$$

Then $E_p^{\lambda,\mu}(x, k) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_j, k_j) \cdot (2^{\mu'})^j$. It implies that the adversary knows

$$E^{\lambda',\mu'}(x_j, k_j)$$

for $0 \le j < p$. Suppose that

$$x_i' = \sum_{0\le j<p} x_{i,j}' \cdot (2^{\lambda'})^j$$

for $1 \le i \le h$. Then $E_p^{\lambda,\mu}(x_i', k) = \sum_{0\le j<p} E^{\lambda',\mu'}(x_{i,j}', k_j) \cdot (2^{\mu'})^j$. It implies that the adversary knows

$$\text{PCP}_j = \{(x_{i,j}', E^{\lambda',\mu'}(x_{i,j}', k_j)) \mid 1 \le i \le h\}$$

for $0 \le j < p$. Since k_j are independently generated for $0 \le j < p$ and $\Pr[A(E^{\lambda',\mu'}(x_j, k_j), \text{PCP}_j, k') = x_j] = 1$ for $k_j \in k'$,

$$\Pr[A(E^{\lambda',\mu'}(x, k), \text{PCP}, k') = x_i] = \prod_{0\le j<p} \Pr[A(E^{\lambda',\mu'}(x_j, k_j), \text{PCP}_j, k') = x_j]$$
$$= \prod_{k_j \notin K'} \Pr[A(E^{\lambda',\mu'}(x_j, k_j), \text{PCP}_j) = x_j].$$

Since $\mu \ge 3\lambda$, $\mu' = \mu/p \ge 3(\lambda/p) = 3\lambda'$. Since h is bounded by a polynomial of λ and $p = O(\lambda^c)$, $0 < c < 1$, h is also bounded by a polynomial of $\lambda' = \lambda/p$. Therefore

$$\Pr[A(E^{\lambda',\mu'}(x_j, k_j), \text{PCP}_j) = x_i] = \text{neg}(\lambda')$$

for $k_j \notin k'$. Since $p = O(\lambda^c)$ for some constant $0 < c < 1$, a negligible function of $\lambda' = \lambda/p$ is also a negligible function of λ. Hence $\Pr[A(E_p^{\lambda,\mu}(x, k), \text{PCP}, k') = x] = \text{neg}(\lambda)$.

Theorem 2: The basic-DOPE protocol is efficient and correct, and achieves the one-wayness security against the adversary structure $\text{AS} = \{U_A \cup \text{KA}_A, U_A \cup \text{KA}_A \cup \{\text{DB}\} \mid U_A \subset CU, \text{KA}_A \subset \text{KA}\}$.

Proof. Since $\lambda' = \lambda/p$ and $\mu' = \mu/p$, the OPE algorithm $E^{\lambda',\mu'}$ is efficient. Also, the processes to express x in base $2^{\lambda'}$ and combines the encryptions to $C_{OPE}(x)$ are efficient. Therefore, the basic-DOPE protocol is efficient.

The basic-DOPE protocol is correct because the DB receives $C_{OPE}(x) = \sum_{0\le j<p} y_j \cdot (2^{\mu'})^j = \sum_{0\le j<p} E^{\lambda',\mu'}(x_j, k_j) \cdot (2^{\mu'})^j = E_p^{\lambda,\mu}(x, k)$.

For security, we assume that the adversary A compromises DB and $U_A \subset U$, $KA_A \subset KA$. Then the adversary knows some plaintext ciphertext pairs in the set PCP, where the plaintexts are from the users in U_A and the ciphertexts are from DB. Also, the adversary A can retrieve the keys in k', where $k' = \{k_j \mid \text{KA}_j$

$\in$ KA_A}. Now consider a user $U_i \notin U_A$ sends $E_p^{\lambda,\mu}(x, k)$ to DB. Then it is equivalent for the adversary to compromise $E_p^{\lambda,\mu}(x, k)$ given PCP and k'. Since $\mu \geq 3\lambda$ and $p = O(1)$, according to Theorem 1, $\Pr[A(E_p^{\lambda,\mu}(x, k), \text{PCP}, k') = x] = \text{neg}(\lambda)$. Hence, the basic-DOPE protocol achieves the one-wayness security against the adversary structure AS.

Theorem 3: The OE-DOPE protocol is efficient and correct. Furthermore, consider the adversary structure AS. Suppose that a user $U_i \notin U_A$ sends x to DB. If the adversary A does not compromise all the key agents $KA_{u,0}, \ldots, KA_{u,q-1}$, simultaneously, then the probability for A to retrieve the "digit" x_u of x is negligible, where $x = \sum_{0 \leq u < p} x_u \cdot (2^{\lambda'})^u$.

Proof. The efficiency can be proven by a routine check. In the protocol, the user need to create $W_{u,v}$ including λ'' elements in $\{0,1\}^{\lambda''}$, $0 \leq u < p$ and $0 \leq v < t$. Since p is a constnat, $\lambda'' = \lambda'/t = \lambda/(p \cdot t)$ and $t = \lambda'^c$, $0 < c < 1$, it is efficient for the user to create $W_{u,v}$. Also, it is efficient for the user to create $\pi_{u,v,j}$ and $L_{u,v,j}$, $0 \leq u < p$, $0 \leq v < t$, and $0 \leq j < q$. Then the $KA_{u,j}$ need to encrypt the elements in $W^{(j)}{}_{u,v}$ by using E^{λ_j,μ_j}. Note that $\lambda_0 = \lambda'' = \lambda/(p \cdot t)$, $\mu_j = \lambda_{j+1}$ for $0 \leq j < q-1$, and $\mu_j = 3\lambda_j$ for $0 \leq j < q$. Since p and q are constants, $\lambda_j = \mu_j = O(\lambda)$. Therefore the encryption E^{λ_j,μ_j} is efficient. It is also efficient to perform the permutation $\pi_{u,v,j}$ and $L_{u,v,j}$ on the encryptions. Finally, it is efficient for the DB to identify the location of the encryptions of $x_{u,v}$ and integrate them to get $C_{OPE}(x)$.

According to the encryption structure, $C_{OPE}(x)$ is the ciphertext encrypted by the basic DOPE scheme $SE_p^{\lambda,\mu}$. $SE_p^{\lambda,\mu}$ is based on the underlying basic DOPE scheme $SE_t^{\lambda',\mu'}$. And $SE_t^{\lambda',\mu'}$ is based on the underlying OPE scheme $SE^{\lambda'',\mu''}$, where $E^{\lambda'',\mu''} = E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$. Hence $C_{OPE}(x)$ preserves order and the OE-DOPE protocol is correct.

For security, first note that if E^{λ_j,μ_j} is a ROPF (random order-preserving function) for $0 \leq j < q$, then the composition $E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$ is also a ROPF. Therefore the basic DOPE scheme $SE_t^{\lambda',\mu'}$ based on the underlying OPE scheme $SE^{\lambda'',\mu''}$ where $E^{\lambda'',\mu''} = E^{\lambda_{q-1},\mu_{q-1}} \circ \cdots \circ E^{\lambda_0,\mu_0}$ has one-wayness security according to Theorem 1. Hence, the basic DOPE scheme $SE_p^{\lambda,\mu}$ based on the underlying basic DOPE scheme $SE_t^{\lambda',\mu'}$ also achieves one-wayness security according to Theorem 1. Now suppose that the adversary A compromises DB and retrieves $C_{OPE}(x)$. Then A can derive $E^{\lambda'',\mu''}(x_{u,v}) = E^{\lambda_{q-1},\mu_{q-1}}(\cdots E^{\lambda_0,\mu_0}(x_{u,v}, k_{u,0}) \cdots, k_{u,q-1})$ from $C_{OPE}(x)$, $0 \leq u < p$ and $0 \leq v < t$. In order to retrieve x_u, A needs to retireve all $x_{u,v}$ for $0 \leq v < t$. Since E^{λ_j,μ_j} has one-wayness security and A does not compromises all $KA_{u,j}$ for $0 \leq j < q$, it implies that the probability for A to derive x_u is negligible. Additionally, since the adversary A does not compromise the key agents $KA_{u,0}, \ldots, KA_{u,q-1}$ simultaneously, A cannot retrieve all $\pi_{u,v,j}$ for $0 \leq j < q$, or all $k_{u,j}$ for $0 \leq j < q$. Furthermore, the order information in $W^{(q)}{}_{u,v}$ cannot be used to link them to the plaintexts in $W_{u,v}$. Thus, A cannot identify $x_{u,v}$ in the vector $W_{u,v}$ even if it compromises $KA_{u,0}$ and DB. But if the adversary compromises the key agent $KA_{u,j}$, $1 \leq j < q$, it can narrow down $x_{u,v}$ from λ'' elements to $\lambda''/2$ element based on $L_{u,v,j}$. Hence the probability for the adversary to retrieve $x_{u,v}$ in $W_{u,v}$ is at most $2^{q-1}/\lambda''$ (note that the adversary can compromise at most $q-1$ key agents in a chain). Consequently, the probability for the adversary to retrieve $x_u = \sum_{0 \leq v < t} x_{u,v} \cdot (2^{\lambda''})^v$ is $2^{q-1}/\lambda''^t$. Since $p = q = O(1)$, $t = \lambda'^c = (\lambda/p)^c$ for some constant $0 < c < 1$. It implies that $2^{q-1}/\lambda''^t$ is a negligible function of λ. Hence, the probability for A to retrieve x_u of x is negligible.