Security with Privacy - A Research Agenda (Invited Paper)

Elisa Bertino and Bharath K. Samanthula

Cyber Center and Department of Computer Science, Purdue University
305 N. University Street, West Lafayette, IN 47907, USA
Email: {bertino, bsamanth}@purdue.edu

***Abstract*—Data is one of the most valuable assets for organization. It can facilitate users or organizations to meet their diverse goals, ranging from scientific advances to business intelligence. Due to the tremendous growth of data, the notion of big data has certainly gained momentum in recent years. Cloud computing is a key technology for storing, managing and analyzing big data. However, such large, complex, and growing data, typically collected from various data sources, such as sensors and social media, can often contain personally identifiable information (PII) and thus the organizations collecting the big data may want to protect their outsourced data from the cloud. In this paper, we survey our research towards development of efficient and effective privacy-enhancing (PE) techniques for management and analysis of big data in cloud computing. We propose our initial approaches to address two important PE applications: (i) privacy-preserving data management and (ii) privacy-preserving data analysis under the cloud environment. Additionally, we point out research issues that still need to be addressed to develop comprehensive solutions to the problem of effective and efficient privacy-preserving use of data.**

I. Introduction

With the advances in technology, organizations are able to collect huge volumes of data; for example, IBM creates 2.5 quintillion bytes of data everyday from different data sources, such as sensors, weblogs, GPS signals and social media [1]. Due to the tremendous growth of data [2], the notion of big data has certainly gained momentum in recent years. Big data essentially deals with the efficient management of large-volume, complex, and growing datasets from multiple sources and the extraction of useful knowledge from these datasets [3]. Many of today's applications across multiple domains, such as social networks [4], healthcare [5], finance [6], manufacturing [7], cyber security [8], [9], biology [10], and physics [11], require the collection, management, integration, and analysis of big datasets. The President Obama's administration announced in 2012 the "Big Data Research & Development" initiative to exploit big data for enhancing research and innovation [12].

In particular, as discussed by Bertino [13], technological advances and novel applications, such as sensors, cyber-physical systems, smart mobile devices, cloud systems, data analytics, and social networks, are making possible to capture, and quickly analyze huge amounts of data from which to extract information critical for security-related tasks. In the area of cyber security, such tasks include user authentication, access control, anomaly detection, user monitoring, and protection from insider threat [14]. By analyzing and integrating data collected on the Internet and Web one can identify connections and relationships among individuals that may in turn help with homeland protection. By collecting and mining data concerning user travels and disease outbreaks one can predict disease spreading across geographical areas. And those are just a few examples; there are certainly many other domains where data technologies can play a major role in enhancing security.

The use of data for security tasks raises however major privacy concerns. Collected data, even if anonymized by removing identifiers such as names or social security numbers, when linked with other data may lead to re-identify the individuals to which specific data items are related to. Also, as organizations, such as governmental agencies, often need to collaborate on security tasks, data sets are exchanged across different organizations, resulting in these data sets being available to many different parties. The big question is thus "how to share and analyze big data in a privacy-preserving manner?" A report recently released by the White House [15] has emphasized the need to reconcile research based on big data with privacy.

When dealing with big data management and analysis, cloud computing represents today's one of the most convenient computing and storage infrastructures. However, the use of the cloud further complicates the problem of data privacy. A solution often advocated to address the problem of data privacy in the cloud is based on encryption by which data is encrypted before being outsourced to the cloud. Recent research has thus focused on techniques for querying and managing encrypted data on the cloud without requiring data decryption (e.g., [16]–[22]). However, a major drawback of those approaches is the lack of scalability and limited applicability.

In this paper, we survey our research towards the goal of developing efficient and effective privacy-enhancing (PE) techniques, tools, and systems for the management of big data on the cloud and outline research directions.

The rest of the paper is organized as follows. Section II introduces two architectural frameworks on which the discussion in this paper is based. Section III presents our initial results and open research issues in privacy-preserving data management and data analytics. Section IV outlines a few conclusions.

II. A Discussion Framework

An important observation underlying the discussions in this paper is that the specific PE technology to use depends on the intended use of data. In this respect, it is important to distinguish between the use of data for analytic purposes, such as performing data mining on the data, and for operational

COLLABORATECOM 2014, October 22-25, Miami, United States

DOI 10.4108/icst.collaboratecom.2014.257687

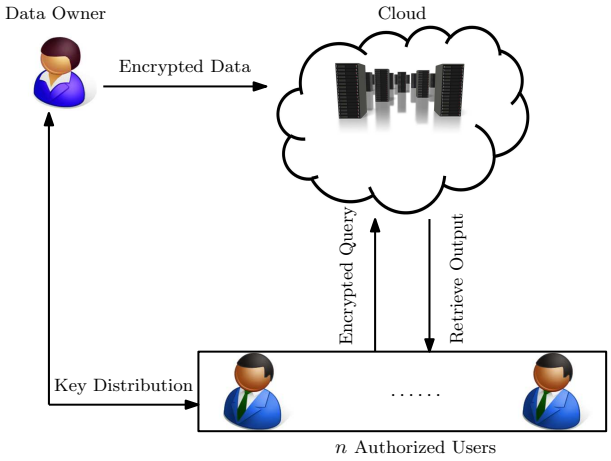


Fig. 1: The PPDM Architecture

use in which specific data records may have to retrieved. This distinction leads to two major categories of PE technology: (i) privacy-preserving data management (PPDM) and (ii) privacy-preserving data analytics (PPDA).

With respect to PPDM, in what follows we consider the setting shown in Figure 1, where the data owner (i.e., an organization holding a big dataset) encrypts the data and stores it in the cloud. Relevant research issues in this setting include how to support SQL queries and other DBMS functions on the encrypted data. Moreover, as data is typically selectively shared among different users of the organization, such as customers or employees, techniques for fine-grained access control on encrypted data stored in a cloud are critical.

On the other hand, for PPDA, we consider two different settings (see Figure 2): (i) single-user and (ii) multi-user collaborative setting. In the single-user setting, we have a single data owner who wants to outsource (due to lack of proper resources, such as technical expertise, computational power and storage) her encrypted data and analytical tasks to a cloud. Under the multi-user setting, we have multiple data owners, each independently holding their own database (e.g., different hospitals holding their patients' medical data), who want to perform a data analytical task on their combined data. The data owners however are not willing to share their own databases among each other, even though they are willing to share the results of the data analysis task. For this purpose, data owners outsource their respective encrypted data to a cloud and the cloud can perform the analytical task on their combined encrypted data and return the results to each data owner.

It is however important to notice that any given task can be broken down into sub-tasks and some of the sub-tasks (which we refer to as basic primitives) remain common across different tasks. Thus to construct effective solutions, i.e., to solve any given task in PPDM and PPDA, one needs to first have efficient implementations of all those basic primitives. Such basic primitives include secure equality, comparison, division, and modulo operation. Research needs to carried out to identify additional basic primitives that can be used as building blocks to support any given task of PPDM and PPDA. It is also important to notice that, even though implementations exist for most (but not all) of the basic primitives, it is always desirable to either improve existing implementations or develop more efficient solutions. Furthermore, in order to make such primitives applicable for use in big datasets, it is important to investigate implementations based on parallel and distributed data processing techniques, such as MapReduce [23], that are typically supported by cloud environments. Such strategy requires however that the solutions to be developed are amenable to parallelization.

III. Approaches and Research Directions

This section discusses approaches for constructing novel PPDM and PPDA frameworks for big data in the cloud environment. With respect to each proposed framework, we present some initial approach and highlight research challenges.

A. Privacy-Preserving Basic Cryptographic Primitives

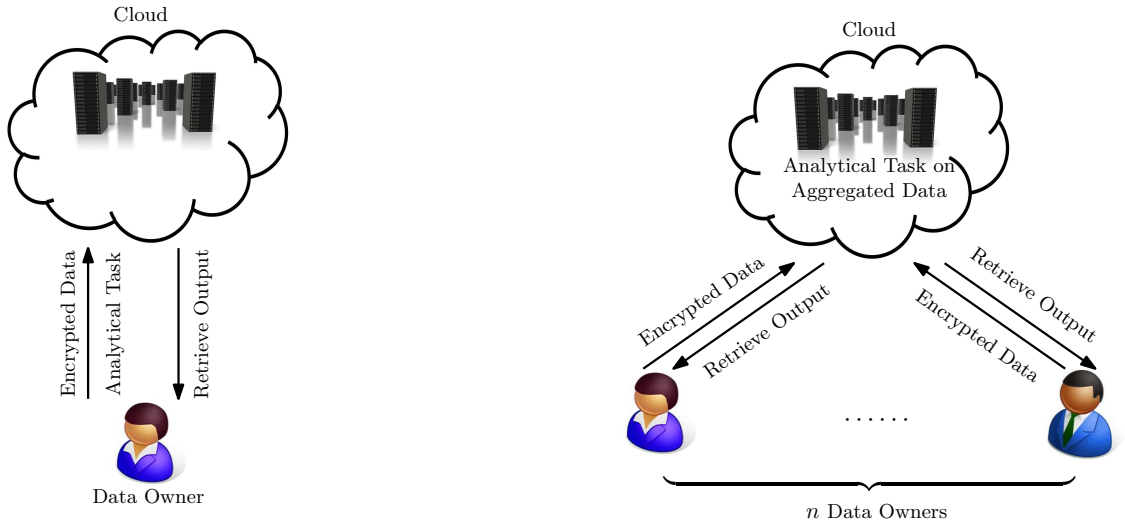
As mentioned in the previous section, basic primitives (e.g., secure equality and comparison) act as important building blocks in constructing solutions to PPDM and PPDA. In particular to encrypted data, cryptographic solutions to basic primitives can offer maximum security when properly designed. For example, if the cloud wants to compare two integers given their encrypted values, it can utilize the existing secure comparison solutions to compare the two encrypted values in a privacy-preserving manner.

1) Related Work: When data is encrypted using fully homomorphic encryption schemes (e.g., [24]), the cloud can perform arbitrary operations over encrypted data in a privacy-preserving manner. However, such schemes are very expensive [25] and their usage in big data applications has yet to be explored. As an alternative, several solutions have been proposed to basic cryptographic primitives (e.g., [26]–[28]) using different techniques, such as additive and multiplicative homomorphic encryption schemes (e.g., [29]). However, to suit the diverse needs of PPDM and PPDA in big data applications, such solutions may not be sufficient.

2) Our work: In our recent works [30]–[32], we proposed efficient solutions to various basic primitives, such as secure multiplication and minimum. Note that our solutions can also be useful in many other secure applications that deal with encrypted data, such as secure electronic voting [33], private auctioning and bidding [34]. In order to meet the needs of big data, we have recently demonstrated [32] how to execute some basic cryptographic primitives using parallelization to improve efficiency by a significant factor. E.g., if the cloud want to compute the minimum value (in encrypted form) out of k encrypted integers, the cloud can generate an execution tree and evaluate it (in parallel) using our secure minimum protocol [32] to get the desired output in a privacy-preserving manner.

3) Research Directions: In what follows, we point out two important objectives for basic primitives in big data applications.

Efficient Basic Primitives for Big Data. We emphasize that it is critical to develop more efficient solutions to various basic cryptographic primitives. We propose three possible directions to achieve this. (i) Develop probabilistic or heuristic based solutions that are expected to be more efficient than existing solutions. For example, in [30], we developed a probabilistic-based solution to the secure bit-decomposition



(a) Single-user data analytics

(b) Multi-user data analytics

Fig. 2: Two types of settings under the PPDA framework

primitive that is much more efficient than the existing solutions. (ii) Develop solutions that can allow cloud to precompute (as part of offline phase) some expensive operations, such as encryptions of random numbers and exponentiations. The more expensive operations we push to offline, the more efficient the solution is. Due to pre-computation of expensive operations in the offline phase, the actual online computation time of a basic primitive is expected to improve. (iii) Another direction is to develop parallel solutions to basic primitives. That is, one need to develop solutions where the underlying operations can be executed in parallel on multiple threads.

Stronger Security Guarantees. Some existing solutions to the basic cryptographic primitives (e.g., [26], [28]) offer weak security guarantees by leaking different information to the cloud. Also, recent results [35] show that access pattern information need to be protected from the cloud to ensure maximum security. Informally speaking, if the information known to the cloud before and after the execution of a basic primitive remains the same, then we can say that it offers strong security [36]. Therefore, while developing efficient solutions to basic primitives, it is also important to ensure that they satisfy the standard security definitions.

B. Privacy-Preserving Data Management

The management of encrypted data stored in a cloud poses several challenges, the most important of which are fine-grained access control and query processing. In a typical organizational setting, different portions of data are shared among employees and customers of the organization owning the data (fine-grained access control). Also, from the database perspective, standard SQL queries should be supported over encrypted data stored in the cloud (query processing). However, to ensure data confidentiality, both fine-grained access control and query processing over encrypted data should be done in a privacy-preserving manner. Thus, a PPDM framework that can facilitate fine-grained access control and query processing over encrypted data in the cloud is of great interest. Given the wide variety of SQL queries and big data, it is always desirable to construct a comprehensive PPDM framework that is effective (can support all standard DBMS tasks), efficient, and scalable.

1) Related Work: Several techniques have been proposed to address access control (e.g., [37]–[41]) and evaluation of specific queries (e.g., [42]–[49]) over encrypted data. The idea of utilizing specialized encryption techniques, such as order preserving encryption [50], [51], additive homomorphic encryption [29], and so on, to perform different relational operations has been firstly introduced in CryptDB [19]. The same idea has been extended to support more complex analytical queries in MONOMI [52]. While such work has established the architectural foundation for systematic query processing and access control, those techniques suffer from two major limitations. The first is that the minimum access control granularity supported by its encryption based access control mechanism is the column. Such a granularity is too coarse to satisfy the requirements of several real-world applications. The second limitation is the onions of encryption. An onion is a multiple layers of encryptions. Each layer is applied for a specific query operation or purpose, and the encryption layers from the external layer to the most internal layer are increasingly weaker. It is easy to see that, although onions offer multiple levels of security, the security decreases over time when the outer layers are removed. Hence, the real security level an onion can guarantee is the protection offered by the inner most encryption. Furthermore, to support diverse operations, multiple onions need to be generated (e.g., an *order* onion is necessary to support range queries).

2) DBMask: DBMask [53] is a recently proposed system overcoming the limitations of CryptDB and supporting attribute-based fine-grained access control for data on the cloud. In what follows we present the key elements of its design and then outline open research directions.

System Architecture. The DBMask system includes four entities: *data owner*, *data user*, *proxy*, and *data server*. The overview of the DBMask architecture and the interactions among different entities are illustrated in Figure 3. The data owner uses different secret keys to encrypt different portions of

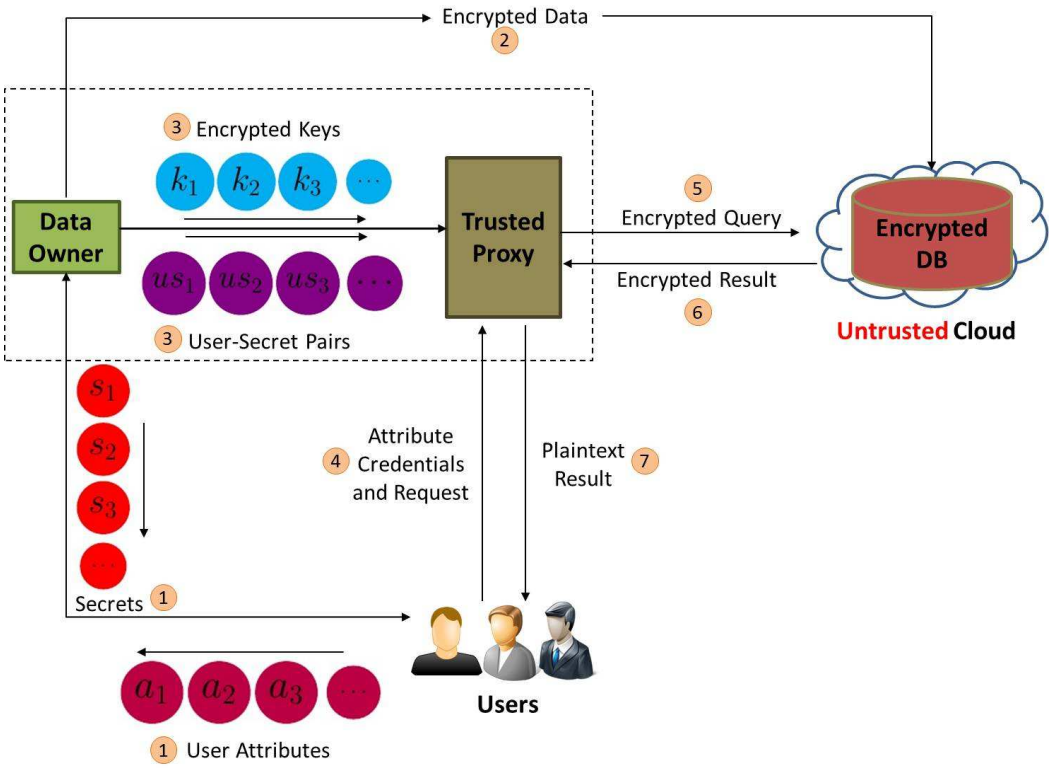


Fig. 3: The Proposed DBMask Architecture

data, according to the access control policies. The secret keys are organized in a lattice for efficient management. The data owner can also build secure indices over the encrypted data to improve the search performance. The encrypted data together with their secure indices are uploaded to the data server (e.g., a cloud). A data user with authenticated attributes can verify itself to the proxy. The successful attribute based verification of the user to the proxy allows the proxy to either derive or obtain one or multiple secret keys required to encrypt the user query. Given a plaintext query submitted by the user, the proxy uses these keys to rewrite the query into an encrypted query, which can then be executed on the encrypted data in the data server. The encrypted query results are returned from the data server to the proxy, which decrypts the results using the secrets established at the time of verification and forwards them to the data user. Notice that during the query processing stage, the data server learns neither the query being executed nor the result set of the query.

Fine-Grained Access Control. DBMask uses attribute based access control (ABAC) model [54], [55] which has the following three characteristics: (i) Users have a set of identity attributes that describe properties of users; for example, organizational role(s), seniority, age and so on. (ii) Data is associated with ABAC policies that specify conditions over identity attributes. (iii) A user whose identity attributes satisfy the ABAC policy associated with a data item is allowed to access the data item.

In DBMask, access can be controlled at different granularity levels such as column level, row level, and cell level. Each column, row, or cell, depending on the desired level of access control, has an associated ABAC policy. In the case of column and cell level control, the policy attachment is performed by adding an additional column to every column or cell in the table. In the case of row level access control, the policy attachment is performed by adding a single additional column in the table. Upon receiving an SQL query from a user for a table T, the proxy needs to determine the ABAC policies attached to T that are satisfied by the users attributes and restrict the query to only those columns, rows or cells depending on the granularity level by adding a predicate to the user query. For ease of presentation, we focus on row level access control in order to discuss DBMask's ABAC model. In this case, each tuple (row) in a database table is attached an ABAC policy. Informally an ABAC policy (ACP for short) over T is defined as a tuple (s, o) where: o denotes a set of rows in T and s is a Boolean expression over a set of attribute conditions that must be satisfied in order to access o. Also, we observe that grouping users based on the ABAC policies they satisfy enhances access control enforcement as it provides one level of indirection. Such a grouping of users allows one to enforce access control policies on a set of users instead than on individual users. Moreover, relationships between groups can be exploited to improve the key management. Considering the fact that every ACP can be converted into disjunctive normal form (DNF), we define a *g*roup G as a set of users which satisfy a specific conjunction of attribute conditions in an ABAC policy.

The idea of groups is similar to user-role assignment in role based access control (RBAC) model, but in our approach, the assignment is performed automatically based on identity attributes. Given the set of data owner defined ABAC policies, the following steps are followed to identify the groups:

- Convert each ABAC ACP into DNF. Note that this conversion can be done in polynomial time.
- For each distinct disjunctive clause, create a group.

For example, consider the following two ACPs: $ACP_1 = C_1 \wedge (C_2 \vee C_3)$ and $ACP_2 = C_2$ with the attribute conditions C_1, C_2 and C_3. The ACPs can be rewritten in DNF form as follows: $ACP_1 = (C_1 \wedge C_2) \vee (C_1 \wedge C_3)$ and $ACP_2 = C_2$. In this example, there are three groups G_1, G_2, G_3 for the set of users satisfying the attribute conditions $C_1 \wedge C_2$, $C_1 \wedge C_3$, and C_2, respectively.

DBMask exploits the hierarchical relationship among groups in order to support hierarchical key derivation and improve the performance and efficiency of key management. We introduce the concept of *G*roup Poset as follows to achieve this objective. A group poset is defined as the partially ordered set (poset) of groups where the binary relationship is $\subseteq$. In the above example, $G_1 \subseteq G_3$ and there is no ordering between G_1 and G_2.

Hierarchical key encryption techniques reduce the number of keys to be managed. However, a major drawback is that assigning keys to each node and giving them to users beforehand makes it difficult to handle dynamics of adding and revoking users. For example, when a user is revoked, one needs to update the keys given to other users through private communication channels. We address this drawback while utilizing the benefits of the hierarchical model by proposing a hybrid approach combining broadcast and hierarchical key management. A broadcast group key management (BGKM) allows one to efficiently handle group keys when user dynamics change. We utilize a recent expressive scheme called AB-GKM (attribute based GKM) as the broadcast GKM scheme [56]. Instead of directly assigning keys to each node in the hierarchy, we assign a AB-GKM instance to each node and authorized users can derive the key using the key derivation algorithm of AB-GKM.

SQL-Aware Comparison. DBMask currently supports both numerical and keyword comparison and is designed so that any comparison friendly numerical [57], [58] or keyword [42], [59], [60] encryption schemes can be utilized to perform relational operations over encrypted data. We refer to these schemes as privacy-preserving numerical comparison (PPNC) and privacy-preserving keyword comparison (PPKC). Both encryption schemes can be summarized into four algorithms: *Setup*, *EncVal*, *GenTrapdoor* and *Compare*, which we use to perform comparisons over encrypted values in DBMask. The *Setup* algorithm takes as input a set of parameters P and initializes the underlying encryption scheme. Given a numerical or keyword value x, the *EncVal*(x) algorithm produces an encrypted value e_x for x. Given an input (numerical or keyword) value t, the *GenTrapdoor* algorithm produces an encrypted value e_t for t, called the trapdoor. Finally, given an encrypted value e_x and a trapdoor value e_t, the *Compare* algorithm compares them and outputs the desired result.

DBMask Protocols. DBMask implements several protocols that support four main functions of DBMask: *system initialization, user registration, data encryption and upload, and data querying and retrieval*. Note that, as shown in Figure 3, DBMask consists of four entities: data owner, data user, proxy, and data server.

During the *system initialization* phase, the data owner runs the Setup algorithms of the underlying cryptographic constructs, that is, AB-GKM.Setup, PPNC.Setup and PPKC.Setup (we use the dot notation to refer to an algorithm of a specific cryptographic construct. For example, AB-GKM.Setup refers to the Setup algorithm of the AB-GKM scheme). The data owner makes available the public security parameters to the proxy so that the proxy can generate trapdoors during data querying and retrieval phase. The data owner also converts the ACPs into DNF and groups users satisfying the same disjunctive clauses. As mentioned earlier, these groups are used to construct the Group poset to perform hierarchical key derivation along with the AB-GKM based key management.

TABLE I: Sample Patients' Medical Data

ID	Age	Diagnosis	Groups
1	35	HIV	G_1
2	30	Cancer	G_1, G_2
3	40	Asthma	G_2, G_3
4	38	Gonorrhea	G_1

In the *user registration* phase, users first get their identity attributes certified by a trusted identity provider. These certified identity attributes are cryptographic commitments that hide the actual identity attribute value but still bind the value to users. Users register their certified identity attributes with the data owner using the oblivious commitment based envelope (OCBE) protocol [61]. The data owner generates the secrets for the identity attributes using the AB-GKM scheme and gives the encrypted secrets to users. Users can decrypt and obtain the secrets only if they present valid certified identity attributes. The data owner maintains a database of user-secret values. When a user or an identity attribute is revoked, the corresponding association(s) from the user-secret database is (are) deleted. The user-secret database is also stored at the proxy with the secrets encrypted using a password only each user possesses. Each user has a different password encrypting her own secrets. Every time the user-secret database changes, the data owner synchronizes its changes with the proxy.

Since it is difficult to support both fine-grained access control and comparison under one encryption scheme, in DBMask each cell in an original table is encrypted twice during the *data encryption and upload* phase. The first encryption is for fine-grained access control and the second is for privacy-preserving matching. Correspondingly, each column in the original table is expanded to two. We denote the column resulting from the encryption for fine-grained access control as *data-col*, and the one resulting for the encryption for privacy-preserving matching as *match-col*.

Let us first discuss the creation of data-col. Given a cell in the original table, its encryption in the corresponding data-col is generated by a secret key derived from the AB-GKM scheme [56] as follows. Consider the row containing the cell in the original table. Based on the ACPs, each row is assigned one or more group labels. The set of groups decides the key, under which the cell in the row is encrypted. If two groups are connected in the group poset, only the label of less privileged group is assigned to the row. The intuition behind is that users in the more privileged group can reach the less privileged group by following the hierarchical relation in the group poset. After removing the labels of groups with higher privileges, a row can still be associated with multiple

groups. For each remaining group G_i, a group secret key k_i is generated by executing the AB-GKM.KeyGen algorithm. In order to avoid multiple encryptions (i.e., one group secret key for one encryption), the AB-GKM.KeyGen algorithm (denoting the key generation algorithm of AB-GKM scheme) is again executed to generate a master group key k using the group keys k_i's as secret attributes to the algorithm. As a consequence, if a user belongs to any of the groups assigned to the row, the user can access the row by executing the key derivation algorithm of AB-GKM algorithm twice. The first execution generates the group key and second derives the master key.

For example, consider the sample patient's medical data given in Table I. Each group G_i is assigned a unique k_i. Rows 1 and 4 are encrypted using key k_1. Since rows 2 and 3 have multiple groups, in order to avoid multiple encryptions/decryptions, a master key is assigned using AB-GKM by considering the group keys as input secrets to the AB-GKM.KeyGen algorithm. Row 2 is encrypted using key k_{12} generated from the AB-GKM instance having k_1 and k_2 as input secrets. Similarly, row 3 is encrypted with key k_{23}.

Table II shows the final encrypted data with both encrypted data-col's and comparison friendly match-col's that need to be outsourced to the cloud, where $comp_n$ and $comp_k$ refer to the encryption functions under PPNC and PPKC, respectively.

Once the encrypted data is outsourced to the cloud server, the next phase is *data querying and retrieval*. DBMask is designed to process a query over encrypted data using a *filtering-refining* procedure. Initially, an authorized user sends a plaintext SQL query to the proxy, as if the outsourced database were unencrypted. In other words, encryption and decryption of the data in the database is transparent to the users. The proxy parses the query and generates an abstract syntax tree of the query as follows. The query is first filtered by removing clauses, such as `ORDER BY` and aggregate functions, that cannot be computed on the server. Then the proxy adds the columns referenced by filtered clauses or aggregate functions to the projections of the filtered query. The query is then rewritten for the cloud by which each column to be included in the query result (i.e., column following the `SELECT` keyword in the query) is replaced by its corresponding "data-col" and each predicate in the `WHERE` clause is replaced with a user defined function (UDF). For each numerical matching predicate, the UDF includes the trapdoor value computed by the proxy using PPNC.GenTrapdoor algorithm and invokes the PPNC.Compare algorithm. Similarly, for each keyword matching predicate, the UDF includes the trapdoor value computed by the proxy using PPKC.GenTrapdoor algorithm and invokes the PPKC.Compare algorithm. Also, a predicate is added to the `WHERE` clause to determine the group(s) of the user requesting the query before the rewritten query is sent to the cloud server.

Upon receiving the rewritten query, the cloud executes it over the encrypted database and filters the tuples that do not satisfy the predicates in the query before sending back the encrypted result set to the proxy. The proxy generates the necessary keys for decrypting the result set using the AB-GKM.KeyDer algorithm with the public information and the user secrets as well as the hierarchical key derivation. If the proxy has removed some clauses (e.g., `ORDER BY`) and/or aggregate functions (e.g., `SUM`) from the original query in the query filtering step, it populates an in-memory database with the decrypted result set and refines the query result according to the constraints in the clauses and/or aggregate functions by running the original query. If no term from the query is removed, the decrypted result set is the final result and the proxy sends the final plaintext result back to the user.

3) Research Directions: While DBMask is an effective initial solution and provides a good starting point towards developing a comprehensive solution to PPDM, several research challenges need to be addressed which we highlight them below.

Support for Additional Relational Operations. In our initial solution to DBMask (as discussed above), the cloud can perform only comparison operations over encrypted data (both numerical and textual data) and the rest of the SQL operations are performed by the proxy. That is, the query execution is a two-step approach in DBMask. The ideal scenario for DBMask would be to execute all the SQL query operations by the cloud itself. This is especially beneficial when the proxy do not have enough resources. Also, if the total computations of a SQL query are performed by the cloud (which is assumed to have better resources than the proxy), then the query-response time is expected to improve. However, achieving the ideal scenario under DBMask is challenging. In our future work, we plan to extend DBMask to support additional relational operations, such as RANGE and JOIN queries, on the cloud side.

Efficient Basic Cryptographic Primitives for PPDM. While the existing cryptographic primitives (e.g., PPNC and PPKC) are useful for performing certain relational operations over encrypted data, it is always desirable to develop more efficient solutions for these operations in order to handle big data. For this purpose, it is important to first identify all the cryptographic primitives required for evaluating SQL queries and then systematically investigate them to improve their efficiency either through algorithmic optimizations of the existing solutions or by proposing new solutions for them.

Privacy Enhancements. Though the initial design of DBMask is such that the contents of the database is hidden from the cloud server by encryption, DBMask stores group information in plaintext format and thus leaks data access patterns [17], [35], [62], [63] to the cloud. Therefore, it is critical to investigate techniques to hide such information and incorporate such techniques into DBMask without affecting the other functions offered by DBMask. Also privacy issues related to information leakage to the proxy must be investigated in order to develop an enhanced privacy-aware DBMask system that can protect the confidentiality of the outsourced database and the user's query from the proxy, the server and any other adversary at all times.

Efficient and Scalable Solutions for PPDM. The specific research issues can be better explained with the help of the following example. Suppose the cloud server has to perform comparison operation on 1 Million encrypted data records (stored in the cloud) using an encrypted search input. In such a case, we need to (i) develop an efficient secure comparison protocol and (ii) devise an efficient plan to divide the 1 Million secure comparison operations into independent sub-tasks and run them in parallel (on multiple nodes) to obtain best performance. Also, due to the distributed nature of data processing,

TABLE II: Encrypted Patients' Medical Data Outsourced to the Cloud Server

ID-enc	ID-com	Age-enc	Age-com	Diag-enc	Diag-com	Groups
$E_{k_1}(1)$	$comp_n(1)$	$E_{k_1}(35)$	$comp_n(35)$	E_{k_1}(HIV)	$comp_k$(HIV)	G_1
$E_{k_{12}}(2)$	$comp_n(2)$	$E_{k_{12}}(30)$	$comp_n(30)$	$E_{k_{12}}$(Cancer)	$comp_k$(Cancer)	G_1, G_2
$E_{k_{23}}(3)$	$comp_n(3)$	$E_{k_{23}}(40)$	$comp_n(40)$	$E_{k_{23}}$(Asthma)	$comp_k$(Asthma)	G_2, G_3
$E_{k_1}(4)$	$comp_n(4)$	$E_{k_1}(38)$	$comp_n(38)$	E_{k_1}(Gonorrhea)	$comp_k$(Gonorrhea)	G_1

we need to minimize the underlying communication costs and network delays. It is thus important to develop efficient solutions to the basic cryptographic primitives (mentioned above) and integrate them in the DBMask architecture. Specifically, it is critical to focus on developing parallel algorithms for each basic primitive that allows one to exploit thread-level parallelism at each node and on how the cloud server can efficiently decompose a given query into independent subqueries and run them on multiple nodes.

Support for other Database Functionalities. Apart from the standard SELECT queries, a comprehensive DBMask solution should also support other common functionalities (e.g., INSERT and UPDATE) of database management systems. However, while incorporating such functionalities into DBMask, we need to ensure that access patterns to data [35] are protected properly. For example, if an authorized user wants to update the contents of a particular data record stored in the cloud, then both the contents of the data record and the information related to which data record is being updated have to be protected from the cloud.

C. Privacy-Preserving Data Analytics (PPDA)

In our knowledge-driven world, it is typical that organizations want to extract useful information by analyzing their large volumes of data. Also, collaborative data analytical models need to be used when data comes from multiple parties. Specifically, consider a group of organizations, each holding a private dataset, who want to perform certain analytical task on their combined data for mutual benefit or other purposes (e.g., in public or government interest). E.g., in a collaborative research effort, hospitals may want to know the age groups that are highly prone for different diseases by applying classification on their combined patients' medical data. On one hand, organizations may not have enough resources (e.g., technical expertise, computational power and storage) to locally perform data analytical tasks on big data. Also, in the case of collaborative setting, since an organization's data are its most valuable asset and due to various privacy concerns, it may not be willing to share its data with others. To overcome these issues, the problem of privacy-preserving data analytics (PPDA) in the cloud has gained significant attention in recent years, where users can outsource their encrypted data to a cloud and the cloud can perform the analytical task over encrypted data in a privacy-preserving manner. Given the wide variety of data analytical tasks, developing a comprehensive PPDA framework that is effective, efficient, and scalable remains a topic of great interest for big data applications.

1) Related Work: There has been significant amount of work on privacy-preserving data mining (e.g., [27], [64]–[70]) by which multiple parties can collaboratively compute a data mining task without revealing one's private data to others. Unfortunately, there has been little work on privacy-preserving data analytics in the cloud. More specifically, existing approaches along this direction are either greatly limited to specific tasks, such as k nearest neighbors [31], [71], clustering [72], [73], classification [32] and association rule mining [22], [74]–[76], or mostly focused on analyzing the trade-offs among different metrics (e.g., [77], [78]). To our knowledge, none of the existing work address the PPDA problem effectively and the associated scalability issues for big data applications.

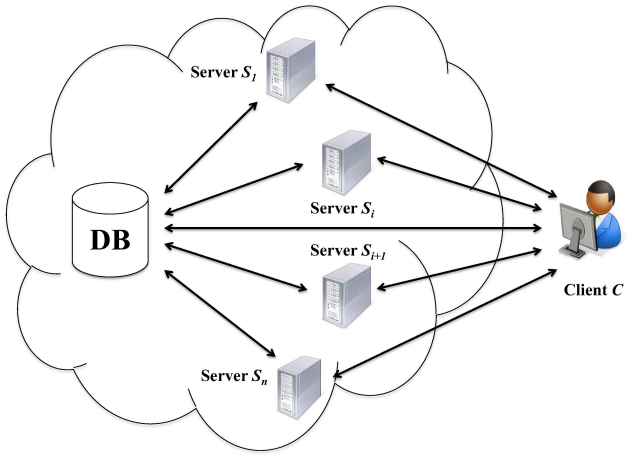


Fig. 4: The Proposed Collaborative PPDA Architecture

2) Collaborative PPDA - An Initial Approach: We plan to develop a comprehensive PPDA framework that can efficiently solve any given data analytical task in a privacy-preserving manner under the cloud environment. As noted in Section II, we consider the PPDA problem under two different settings: (i) single-user and (ii) multi-user. For succinctness, we consider the single-user setting and describe our proposed framework and initial solution approach for PPDA.

System Architecture. In our initial research, we consider the data-analytics-as-a-service scenario in cloud computing with $n(\geq 3)$ multiple servers, as shown in Figure 4, where there exist three types of participants, each playing a different role, as follows: (i) a group of cloud data analytical servers cooperating to provide privacy-preserving data-analytics-as-a-service; (ii) a cloud database server keeping the user data in a database and playing a role of the gateway between the user and the cloud data analytical servers; and (iii) a cloud user storing data in the cloud database server and outsourcing data analytical task to the multiple cloud data analytical servers.

Technical Approach. We model the privacy-preserving data analytical process in the above system setting into the

following three phases:

- **Phase 1:** The cloud user C transforms his original data with a secret information sk (known to the cloud user only) and uploads the transformed data to the cloud database server DB. With the secret information sk, the transformed data can be restored to the original data. Without the secret information, it is hard to decode the transformed data stored by the cloud user.

- **Phase 2:** To outsource an analytical task to the cloud, the cloud user C divides his secret information into n pieces and distributes them to n cloud data analytical servers, respectively. As long as not all servers collude to recover the transformed data, the user data remains private. Any t (denoting the threshold) out of n servers cooperate to mine the transformed data stored by the user in the database server DB and output mined patterns (which are still transformed) to the user.
- **Phase 3:** Finally, the cloud user C recovers the returned patterns with his secret information sk into plain patterns in the end.

In our model, we assume that some analytical servers in the cloud are trusted not to collude with other servers to recover the transformed data. This assumption is reasonable and has been commonly used in electronic election protocols, such as [79], [80], which even requires higher user privacy. Based on the above model, we have achieved some initial research outcomes. One of the initial outcomes is privacy-preserving association rule mining in cloud computing (where $t = n$) [81], which can be briefly described as follows:

- **Initialization.** The cloud user C generates his public/private key pair (pk, sk) at first. Next, he splits the private key sk into n sub-keys $sk_1, \ldots, sk_n$ such that $sk = sk_1 + \cdots + sk_n$ and distributes $sk_1, \ldots, sk_n$ to n cloud servers, respectively, through the secure channels. We assume that at least one out of the n servers is trusted not to collude with other servers. Then he encrypts his data with the ElGamal encryption scheme [82] and the public key pk and uploads the encrypted data to the cloud database server. After that, he encrypts the minimum support s and broadcasts it to n cloud servers.
- **Frequent Itemset Mining.** On the basis of the n sub-keys, the n cloud data mining servers cooperate to anonymize the encrypted data of the user (stored in the database server) by adding encryption of fake data and then find out the frequent encrypted itemsets with the Aprori algorithm [83] on the basis of the encrypted minimum support s. The encrypted frequent itemsets are then returned to the user.
- **Frequent Itemset Retrieval.** Finally, the user decrypts the encrypted frequent itemsets with the private key sk.

Besides the Aprori algorithm, our frequent itemset mining is built on two privacy-preserving techniques - the Plaintext Equality Test (PET) [84] and the Conditional Gate (CG) [85]–[87]. With the PET, the n analytical servers can cooperate to determine whether two ciphertexts are the encryption of the same item without having to decrypt the two ciphertexts. By the CG, the n analytical servers can cooperate to effectively and efficiently compare two numbers given their encryptions without need for decryption.

Unlike the existing solutions [22], [74]–[76], the fake data in our solution is added and removed by the n servers instead than by the user. The main advantage of our solution is removing the requirement for the cloud user to store data and add fake data locally. What the cloud user is required to do is encrypting its data before uploading it to the cloud and decrypting the mined association rules received from the cloud. The user may upload his data to the cloud in a real time way in the case that the user does not have local data storage.

On the basis of the existing data analytical algorithms, our initial approach makes use of the several cryptographic primitives, apart from the privacy-preserving techniques PET and CG, to protect the confidentiality of the cloud user's data when multiple data analytical servers cooperate to mine the data of the cloud user. The above ideas can be further extended to other PPDA tasks, such as clustering and classification, and can also be enhanced to design a PPDA framework for the multi-user setting.

3) Research Directions: In what follows, we will point out several research challenges associated with PPDA by taking our initial approach for privacy-preserving association rule mining as a baseline. In order to develop a comprehensive PPDA framework for big data applications, we need to address the following research challenges.

Efficient Basic Cryptographic Primitives for PPDA. It is critical to investigate the set of basic cryptographic primitives (e.g., PET, CG and secure division) needed for the PPDA framework and propose efficient solutions to each one of them. Here one can utilize the ideas mentioned in Section III-A3.

Develop a Suite of PPDA Protocols. We emphasize that it is critical to construct a suite of privacy-preserving data analytical algorithms over encrypted data in the cloud. In our initial research [81], we have successfully constructed three solutions for privacy-preserving association rule mining for cloud data. However, different data analytical tasks will require different privacy-preserving solutions. Nonetheless, most of these tasks share some common cryptographic primitives which have to be identified first and then efficient solutions need to be designed for each one of them. One can utilize the primitives as well as strategies from our initial solution to solve other privacy-preserving data analytical problems. Very recently, we have proposed an alternative approach to solve the k-nearest neighbor classification problem over encrypted data under the single-user setting [32]. While our solutions [32], [81] act as a good starting point, we believe that more research need to be done to develop new privacy-preserving algorithms for clustering, classification, association rule learning, anomaly detection, regression, summarization, future learning and graph analysis problems, under both the single-user and multi-user settings in the cloud environment.

Implementation and Performance Evaluation. To evaluate the efficiency and scalability of PPDA solutions, one has to develop prototype implementations of privacy-preserving data-analytics-as-a-service as a platform, composed of three types of software - the user software, the database server software and the data analytics server software. In particular to the database and data analytics servers software, one has to implement

the underlying parallel computations using multi-threading (for parallel computations on a server) and MapReduce (for parallel computations on multiple servers) techniques. Besides performing theoretic computation and communication analysis, one has to execute the prototypes on real big datasets [88] to see the actual running time.

IV. Conclusions and Future Work

Big data deals with data collection from multiple sources, probably containing sensitive or personally identifiable information (PII). Cloud computing is naturally the first choice to store and analyze big data. However, for various privacy reasons, users typically encrypt their data before storing it on the cloud. This raises an important question: "how can the cloud manage big data in an effective and efficient manner?" The development of privacy-preserving techniques, tools, and systems over encrypted data in cloud has gained tremendous interest in recent years.

In this paper, we surveyed our approaches towards addressing two specific problems: (i) privacy-preserving data management (PPDM) and (ii) privacy-preserving data analytics (PPDA). Also, we pointed out various open issues that still need to be addressed to develop comprehensive solutions to PPDM and PPDA. Addressing these issues will be the primary focus of our future work. Though in this paper we focus on encrypted data, other data transformation techniques such as anonymization can also be considered. We will investigate such alternative techniques and analyze their applicability to PPDM and PPDA.

Acknowledgments

The work reported in this paper has been partially supported by the Purdue Cyber Center and by the National Science Foundation under grants CNS-1111512 and CNS-1016722.

References

[1] IBM, "Big data at the speed of business," http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html.

[2] J. F. Gantz and D. Reinsel, "The Digital Universe in 2020: big data, bigger digital shadows, and biggest growth in the far east," IDC, December 2012, http://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf.

[3] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE TKDE*, vol. 26, no. 1, pp. 97–107, Jan 2014.

[4] W. Tan, M. Blake, I. Saleh, and S. Dustdar, "Social-network-sourced big data analytics," *IEEE Internet Computing*, vol. 17, no. 5, pp. 62–69, Sept 2013.

[5] J. Sun and C. K. Reddy, "Big data analytics for healthcare," in *ACM SIGKDD*, 2013, pp. 1525–1525.

[6] B. Warner, "'Big Data' researchers turn to google to beat the markets," Bloomberg Businessweek, April, 2013, http://www.businessweek.com/articles/2013-04-25/big-data-researchers-turn-to-google-to-beat-the-markets.

[7] H. Khatri, "Trends in manufacturing operations: Leveraging big data across the value chain," Oracle, January, 2013, http://www.oracle.com/us/corporate/profit/archives/opinion/011813-hkhatri-1899121.html.

[8] Y. Lee and Y. Lee, "Toward scalable internet traffic measurement and analysis with hadoop," *SIGCOMM Computer Communication Review*, vol. 43, no. 1, pp. 5–13, Jan. 2013.

[9] IBM, "Security intelligence with big data," http://www-03.ibm.com/security/solution/intelligence-big-data/.

[10] Brown University, "Bigdata: analytical approaches to massive data computation with applications to genomics," http://bigdata.cs.brown.edu/.

[11] Lawrence Berkeley National Laboratory, "Big data hits beamline," http://crd.lbl.gov/news-and-publications/news/2013/big-data-hits-beamline/.

[12] T. Kalil, "Big data is a big deal," White House, 2012, http://www.whitehouse.gov/blog/2012/03/29/big-data-big-deal.

[13] E. Bertino, "Security with privacy - opportunities and challenges," Panel Position Paper, Proceedings of COMPSAC, 2014.

[14] E. Bertino, "Data protection from insider threats," Synthesis Lectures on Data Management, Morgan & Claypool Publishers, 2012.

[15] Executive Office of the President, "BIG DATA: SEIZING OPPORTUNITIES, PRESERVING VALUES," US White House, Washington, May 2014, http://www.whitehouse.gov/sites/default/files/docs/big_data_privacy_report_may_1_2014.pdf.

[16] R. Sion, "Towards secure data outsourcing," *Handbook of Database Security*, pp. 137–161, 2008.

[17] P. Williams, R. Sion, and B. Carbunar, "Building castles out of mud: practical access pattern privacy and correctness on untrusted storage," in *CCS*. ACM, 2008, pp. 139–148.

[18] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in *IEEE ICDE*, 2011, pp. 601–612.

[19] R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, "CryptDB: protecting confidentiality with encrypted query processing," in *ACM SOSP*, 2011, pp. 85–100.

[20] A. Sahai, "Computing on encrypted data," *Information Systems Security*, pp. 148–153, 2008.

[21] M. Kuzu, M. S. Islam, and M. Kantarcioglu, "Efficient similarity search over encrypted data," in *IEEE ICDE*, 2012, pp. 1156–1167.

[22] F. Giannotti, L. Lakshmanan, A. Monreale, D. Pedreschi, and H. Wang, "Privacy-preserving mining of association rules from outsourced transaction databases," *IEEE Systems Journal*, vol. 7, no. 3, pp. 385–395, Sept 2013.

[23] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008.

[24] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *ACM STOC*, 2009, pp. 169–178.

[25] C. Gentry and S. Halevi, "Implementing gentry's fully-homomorphic encryption scheme," in *EUROCRYPT*. Springer, 2011, pp. 129–148.

[26] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *ACM SIGMOD*, 2004, pp. 563–574.

[27] P. Bunn and R. Ostrovsky, "Secure two-party k-means clustering," in *ACM CCS*, 2007, pp. 486–497.

[28] I. F. Blake and V. Kolesnikov, "One-round secure comparison of integers," *Journal Mathematical Cryptology*, vol. 3, no. 1, pp. 37–68, 2009.

[29] P. Paillier, "Public key cryptosystems based on composite degree residuosity classes," in *Eurocrypt*. Springer, 1999, pp. 223–238.

[30] B. K. Samanthula and W. Jiang, "An efficient and probabilistic secure bit-decomposition," in *ACM ASIACCS*, 2013, pp. 541–546.

[31] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *IEEE ICDE*, 2014, pp. 664–675.

[32] B. K. Samanthula, Y. Elmehdwi, and W. Jiang, "k-nearest neighbor classification over semantically secure encrypted relational data," IEEE Transactions on Knowledge and Data Engineering (TKDE), To appear, http://arxiv.org/abs/1403.5001.

[33] M. R. Clarkson, S. Chong, and A. Myers, "Civitas: Toward a secure voting system," in *IEEE Security and Privacy*, may 2008, pp. 354 –368.

[34] C. Cachin, "Efficient private bidding and auctions with an oblivious third party," in *ACM CCS*. ACM Press, 1999, pp. 120–127.

[35] M. S. Islam, M. Kuzu, and M. Kantarcioglu, "Access pattern disclosure on searchable encryption: Ramification, attack and mitigation," in *NDSS*, 2012.

[36] O. Goldreich, *The Foundations of Cryptography*. Cambridge University Press, 2004, vol. 2, ch. General Cryptographic Protocols, pp. 599–746.

[37] S. Coull, M. Green, and S. Hohenberger, "Controlling access to an oblivious database using stateful anonymous credentials," in *PKC*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 501–520.

[38] J. Camenisch, M. Dubovitskaya, and G. Neven, "Oblivious transfer with access control," in *CCS*. ACM, 2009, pp. 131–140.

[39] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Security and Privacy*, 2007, pp. 321–334.

[40] S. Yu, C. Wang, K. Ren, and W. Lou, "Attribute based data sharing with attribute revocation," in *ASIACCS*. ACM, 2010, pp. 261–270.

[41] B. K. Samanthula, Y. Elmehdwi, G. Howser, and S. Madria, "A secure data sharing and query processing framework via federation of cloud computing," *Information Systems*, Elsevier, 2013.

[42] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *IEEE Security and Privacy*, 2000, pp. 44–55.

[43] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, "Public-key encryption with keyword search," in *EUROCRYPT*, 2004.

[44] H. Hacigümüs, B. R. Iyer, C. Li, and S. Mehrotra, "Executing sql over encrypted data in the database-service-provider model," in *SIGMOD*, 2002, pp. 216–227.

[45] H. Hacigümüs, B. Iyer, and S. Mehrotra, "Efficient execution of aggregation queries over encrypted relational databases," in *DASFAA*. Springer, 2004, ch. 10, pp. 633–650.

[46] H. Hacigümüş, B. Iyer, and S. Mehrotra, "Query optimization in encrypted database systems," in *DASFAA*. Springer, 2005, vol. 3453, pp. 43–55.

[47] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," *CRYPTO*, pp. 535–554, 2007.

[48] B. K. Samanthula and W. Jiang, "Efficient privacy-preserving range queries over encrypted data in cloud computing," in *Sixth International Conference on Cloud Computing (CLOUD)*. IEEE, 2013, pp. 51–58.

[49] B. K. Samanthula, W. Jiang, and E. Bertino, "Privacy-preserving complex query evaluation over semantically secure encrypted data," in *19th European Symposium on Research in Computer Security (ESORICS)*. Springer, 2014, pp. 400–418.

[50] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *SIGMOD*. ACM, 2004, pp. 563–574.

[51] A. Boldyreva, N. Chenette, and A. O'Neill, "Order-preserving encryption revisited: Improved security analysis and alternative solutions," in *CRYPTO*, 2011, vol. 6841, pp. 578–595.

[52] S. Tu, M. F. Kaashoek, S. Madden, and N. Zeldovich, "Processing analytical queries over encrypted data," in *Proceedings of the 39th international conference on Very Large Data Bases*. VLDB Endowment, 2013, pp. 289–300.

[53] N. Mohammed, M. I. Sarfraz, J. Cao, and E. Bertino, "Dbmask: Fine-grained access control on encrypted relational databases," Submitted to ACM Conference on Data and Application Security and Privacy, 2015.

[54] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT*. Springer-Verlag, 2005, pp. 457–473.

[55] National Cybersecurity Center of Excellence, "Attribute Based Access Control," NIST, February, 2014, http://csrc.nist.gov/nccoe/Building-Blocks/abac.html.

[56] M. Nabeel and E. Bertino, "Towards attribute based group key management," in *ACM CCS*, 2011, pp. 821–824.

[57] A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill, "Order-preserving symmetric encryption," in *EUROCRYPT*. Springer, 2009, pp. 224–241.

[58] M. Nabeel, N. Shang, and E. Bertino, "Efficient privacy preserving content based publish subscribe systems," in *SACMAT*, 2012, pp. 133–144.

[59] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *CCS*. New York, NY, USA: ACM, 2006, pp. 79–88.

[60] S. Kamara, C. Papamanthou, and T. Roeder, "Dynamic searchable symmetric encryption," in *ACM CCS*, 2012, pp. 965–976.

[61] J. Li and N. Li, "OACerts: Oblivious attribute certificates," *IEEE TDSC*, vol. 3, no. 4, pp. 340–352, 2006.

[62] S. De Capitani di Vimercati, S. Foresti, S. Paraboschi, G. Pelosi, and P. Samarati, "Efficient and private access to outsourced data," in *ICDCS*. IEEE Computer Society, 2011, pp. 710–719.

[63] S. De Capitani di Vimercati, S. Foresti, and P. Samarati, "Managing and accessing data in the cloud: Privacy risks and approaches," in *7th International Conference on Risk and Security of Internet and Systems (CRiSIS)*, 2012, pp. 1 –9.

[64] R. Agrawal and R. Srikant, "Privacy-preserving data mining," in *ACM Sigmod Record*, vol. 29. ACM, 2000, pp. 439–450.

[65] Y. Lindell and B. Pinkas, "Privacy preserving data mining," in *Advances in Cryptology (CRYPTO)*. Springer, 2000, pp. 36–54.

[66] B. Pinkas, "Cryptographic techniques for privacy-preserving data mining," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 12–19, 2002.

[67] C. Clifton, M. Kantarcioglu, J. Vaidya, X. Lin, and M. Y. Zhu, "Tools for privacy preserving distributed data mining," *ACM SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 28–34, 2002.

[68] M. Kantarcioglu and C. Clifton, "Privately computing a distributed k-nn classifier," in *PKDD*. Springer-Verlag, 2004, pp. 279–290.

[69] L. Xiong, S. Chitti, and L. Liu, "K nearest neighbor classification across multiple private databases," in *CIKM*. ACM, 2006, pp. 840–841.

[70] C. C. Aggarwal and P. S. Yu, "A general survey of privacy-preserving data mining models and algorithms," *Privacy-preserving data mining*, pp. 11–52, 2008.

[71] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *ACM SIGMOD*, 2009, pp. 139–152.

[72] M. Upmanyu, A. Namboodiri, K. Srinathan, and C. Jawahar, "Efficient privacy preserving k-means clustering," in *Intelligence and Security Informatics*. Springer, 2010, vol. 6122, pp. 154–166.

[73] D. Liu, E. Bertino, and X. Yi, "Privacy of outsourced k-means clustering," in *ACM ASIACCS*, 2014. To appear.

[74] W. K. Wong, D. W. Cheung, E. Hung, B. Kao, and N. Mamoulis, "Security in outsourcing of association rule mining," in *VLDB*, 2007, pp. 111–122.

[75] I. Molloy, N. Li, and T. Li, "On the (in)security and (im)practicality of outsourcing precise association rule mining." in *ICDM*. IEEE Computer Society, 2009, pp. 872–877.

[76] C.-H. Tai, P. S. Yu, and M.-S. Chen, "k-support anonymity based on pseudo taxonomy for outsourcing of frequent itemset mining," in *ACM SIGKDD*, 2010, pp. 473–482.

[77] Q. Lu, Y. Xiong, X. Gong, and W. Huang, "Secure collaborative outsourced data mining with multi-owner in cloud computing," in *IEEE TRUSTCOM*, 2012, pp. 100–108.

[78] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "Gupt: Privacy preserving data analysis made easy," in *ACM SIGMOD*, 2012, pp. 349–360.

[79] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *ACM WPES*, 2005, pp. 61–70.

[80] X. Yi and E. Okamoto, "Practical internet voting system," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 378–387, Jan. 2013.

[81] X. Yi, F.-Y. Rao, and E. Bertino, "Privacy-preserving association rule mining in cloud computing environments," In Preparation.

[82] T. El Gamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *CRYPTO*. Springer, 1985, pp. 10–18.

[83] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *VLDB*. Morgan Kaufmann Publishers Inc., 1994, pp. 487–499.

[84] M. Jakobsson and A. Juels, "Mix and match: Secure function evaluation via ciphertexts," in *ASIACRYPT*. Springer, 2000, pp. 162–177.

[85] B. Schoenmakers and P. Tuyls, "Practical two-party computation based on the conditional gate," in *ASIACRYPT*. Springer, 2004, vol. 3329, pp. 119–136.

[86] D. J. Lilja and S. Sapatnekar, "Designing digital computer systems with verilog," Cambridge University Press, 2005.

[87] D. Harris and S. Harris, "Digital design and computer architecture," Morgan Kaufmann Publishers, 2007.

[88] Amazon Web Services, "Public Data Sets on AWS," Amazon, http://aws.amazon.com/publicdatasets/.