[Book Title], Edited by [Editor's Name].
ISBN 0-471-XXXXX-X

Chapter 3

Security and Privacy in Big Data

Lu Ou, Zheng Qin, Hui Yin, and Keqin Li

3.1 Introduction

The term big data refers to the massive amounts of digital information companies and governments collect about us and our surroundings. Human beings now create 2.5 quintillion bytes of data per day. The rate of the data creation has increased so much that 90% of the data in the world today has been created in the last two years alone. How to securely store these massive data and how to effectively process them have been more and more important and challenging tasks. An effective pattern is that institutions adopt the emerging cloud computing commercial service schema to outsource their massive data sets at remote cloud storage and computing centers. On the other hand, authorized data users can utilize the powerful computation capability of cloud sever to search and process data. For example, a hospital creates vast data sets of patients medical records every day such as EMR (Electronic Medical Records) including x-ray, electrocardiogram, clinical history and so on. Facing such huge amount of data files, hospitals have to resort to third-party data storage and management centers to maintain the massive data sets. Later, authorized physicians can search and obtain individual records through public communication channels.

Generally, big data is processed and queried on the cloud computing platform. Cloud computing, the new term for the long dreamed vision of computing as a utility, enables convenient, on-demand network access to a centralized pool of configurable computing resources that can be rapidly deployed with great efficiency and minimal management overhead. The

amazing advantages of cloud computing include: on-demand self-service, ubiquitous network access, location independent resource pooling, rapid resource elasticity, usage-based pricing, transference of risk, etc. Thus, cloud computing could easily benefit its users in avoiding large capital outlays in the deployment and management of both software and hardware. Undoubtedly, cloud computing brings unprecedented paradigm shifting and benefits in the history of IT, especially, in the big data era.

Cloud computing can provide various elastic and scalable IT services in a pay-as-you-go fashion, however, once the massive data sets are outsourced to a remote cloud, the data owners would lose directly control of these data, which bring a major concern of using cloud - privacy and security problem of data. Gartner listed seven security issues in cloud computing and claimed that 85% large companies refuse to adopt cloud computing just due to data security. For example, the hospital may be reluctant to outsource their data to the cloud because EMRs contain huge amount of confidential sensitive information that patients are reluctant to publish, such as ones life style, medical history, history of drug allergy, etc. To effectively solve data security problem and promote the development of cloud computing, the Gartner thinks that encryption scheme must be available to protect the outsourced cloud data, which should be encrypted before uploading them to the cloud. However, data encryption makes cloud data utilization a very challenging problem. Especially, the most important data search is a key challenging because the encrypted data disable the traditional plaintext keyword query. Thus, enabling an encrypted cloud data search service is of paramount importance.

In addition, attackers can also utilize correlation in big data and background knowledge to steal sensitive information. There are some existing technologies to prevent attackers stealing sensitive information of relational data. But the correlation in big data is quite different from traditional relationship between data stored in relational databases. Correlation in big data does not imply causation. Causality means A causes B, where correlation, on the other hand, means that A and B tend to be observed at the same time. These are very, very different things when it

comes to big data; however, the difference often gets glossed over or ignored, so sensitive information will be exposed in more danger.

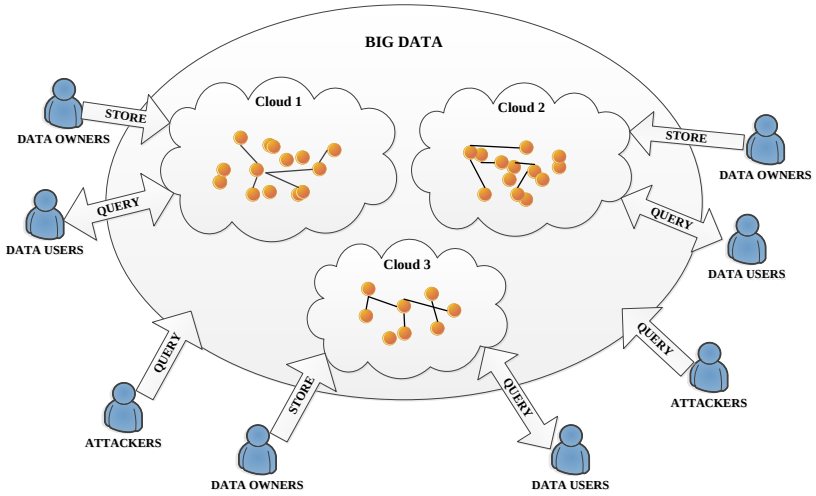

Figure 3.1 An overview of big data

Based on the above description, in this chapter, we will discuss two important security issues about big data under cloud computing environment. One is secure query over encrypted cloud data; the other is the security and privacy in correlative big data. The rest of this chapter is organized as follows. In Section 3.2.1, we introduce the related topics of secure query in the cloud computing, including system model, security model and some outstanding secure query techniques. In section 3.2.2, we comb the related researches about privacy protection on correlated big data, and these researches are divided into two categories, including anonymity and differential privacy. At last, we propose the future work in Section 3.3 and conclude this chapter in Section 3.4 respectively.

3.2 Related Work

3.2.1 Secure Query over Encrypted Big Data

3.2.1.1 System Model

The popular secure query architecture with privacy protection in the cloud computing

involves three entities: the data owner, data users, and the cloud, as shown in Figure 3.2. The system framework expresses the following application scenario: the data owner decide to employ the cloud storage service to store massive data files, which are encrypted for the guarantee that no actual data contents are leaked to the cloud. On the other hand, to enable the effective keyword query over encrypted data for data utilization, the data owner needs to construct confidential searchable indexes for outsourced data files. Both the encrypted file collection and index are outsourced to the cloud server. An authorized data user can obtain data files of interest by submitting specified keywords to the cloud. The query process includes the following four steps. First, the data user sends query keywords of interest to the data owner via secret communication channels. Second, upon receiving query keywords, the data owner encodes these keywords to generate query token in the form of ciphertexts and then returns it to his authorized user via the same secure channels. Third, the data user submits query token to the cloud server to request data files via the public communication channels such as internet. Fourth, upon receiving the query token submitted by the data user, the cloud server is responsible to execute the search algorithm over the secure index and returns the matched set of files to the user as the search result.

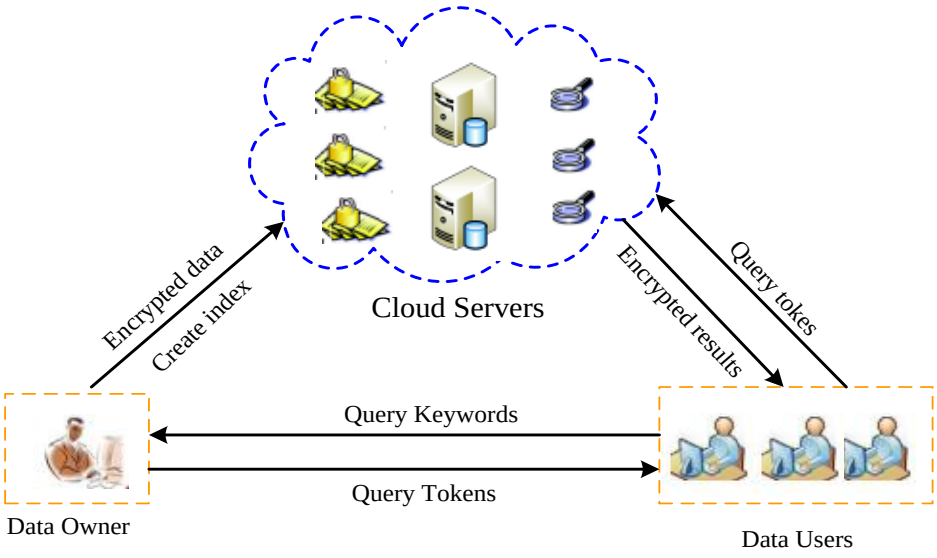

Figure 3.2 Architecture of the search over encrypted cloud data

A fatal drawback of this architecture is that the data owner must always keep online

to handle query requests of authorized users. If the data owner is offline, outsourced data cannot be retrieved in time. To improve the availability, an improved system architecture is proposed as shown in Figure 3.3.

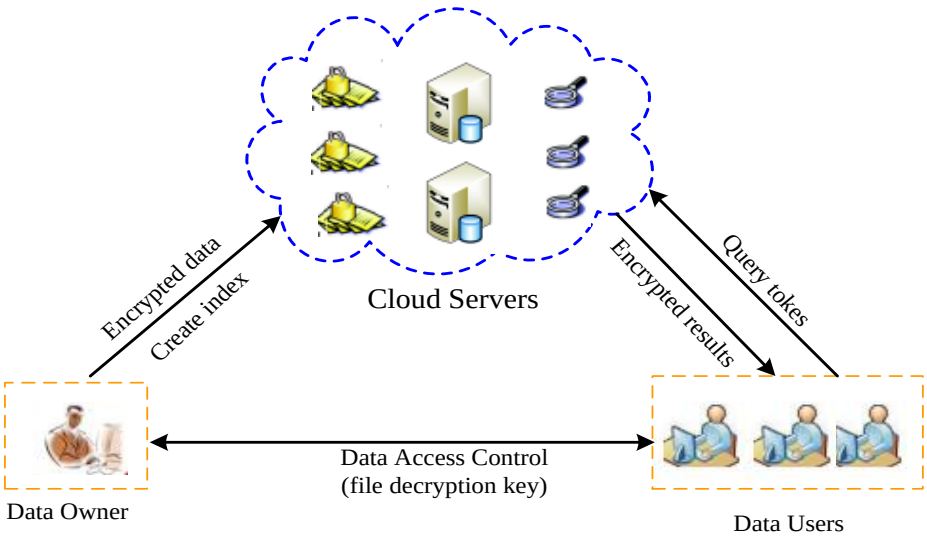

Figure 3.3 Improved architecture of the search over encrypted cloud data

In Figure 3.3, there is an important difference compared with Figure.3.2 that data users can encode keyword(s) to query token locally using some keys assigned by the data owner through data access control and queries are independently of the state of the data owner. Consequently, the improved system model more rigidly protects user query privacies, eliminates real-time communication overhead, and greatly enhances the availability and scalability of the whole system.

3.2.1.2 Threat Model and Attack Model

In this section, we introduce the threat model and attack model based on the above described secure query model for cloud computing.

1. Threat Model

Since the massive data set is stored at remote cloud center, the cloud may be not fully trusted due to the following some reasons. First, clouds may have corrupted employees who do not

follow data privacy policies. Second, cloud computing systems may be vulnerable to external malicious attacks, and when intrusions happen, cloud customers may not be fully informed about the potential implications on the security of their data. Third, clouds may base services on facilities in some foreign countries where privacy regulations are difficult to enforce. Based on the above facts, generally, the cloud server is considered as "honest-but-curious" semi-trusted threat model. More specifically, the cloud server acts in an "honest" fashion and correctly follows the designated protocol specification and promises that it always securely store data and perform computation. However, the cloud server may intentionally analyze contents of data files or curiously infer the underlying query keyword(s) submitted by the data user. Recently, besides the cloud, researchers begin to consider some internal attackers such as a maliciously data owner or a compromised data user in the multi-owner and multi-user cloud environments such that the cloud can collude with these internal attackers to analyze useful information of data files and query keyword(s). Figure 3.4 shows the two threat models in the cloud computing.

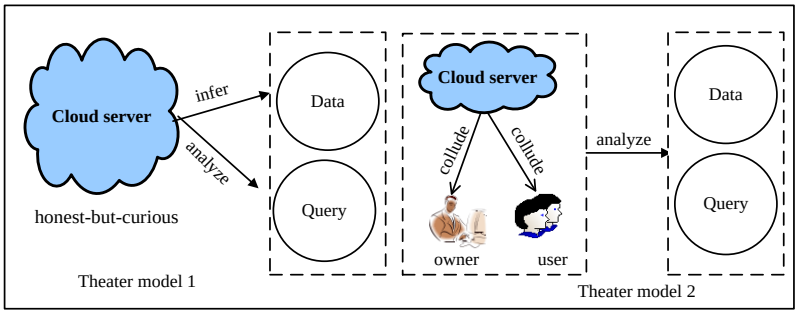

Figure 3.4 Threat model

2. Attack Model

In addition to the threat model, based on what information the cloud server knows, secure query schemes in the cloud environment mainly consider two attack models with different attack capabilities, they are *Known Ciphertext Model* and *Known Background Model* [7, 8] In the *Known Ciphertext Model*, the cloud is only permitted to access the encrypted data files and the searchable secure indexes. In the *Know Background Model*, as a stronger attack model, the cloud server is supposed to possess more background knowledge than what can be accessed in the

known cipertext model. These background knowledge information may include the keywords and their statistical information, Correlation relation of different data sets, as well as the correlation relationship of given search requests. For example, the cloud server could use the known query tokens information combined with previous query results to deduce the plaintext information of certain keywords in the query. Figure 3.5 shows the two attack models.

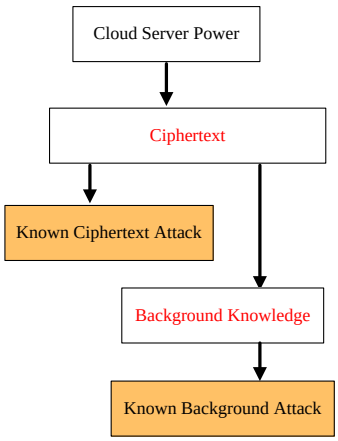

Figure 3.5 Attack model

3.2.1.3 Overview of Secure Query Scheme in the Cloud Computing Environment

In fact, as early as 2000, Song *et al* had set out to research the problem how to correctly perform search over the encrypted document via encrypted keyword and first introduced a practical technique that allows a user to securely perform search over encrypted documents through some specified keywords[3]. In 2006, Curtmola *et al* formally named this query technique as the searchable encryption (SE) and presented explicit security requirements and definitions [4]. Meanwhile, they adopted the inverted index data structure [9] (which is widely used in the plaintext information retrieval fields), symmetric encryption schemes [2], and hash table to design a novel searchable encryption scheme named SSE (Searchable Symmetric Encryption). The key goal of an SE scheme is to allow a server to search over encrypted data on

behalf of data users without learning any useful information of the data files as well as the query contents of data users other than encrypted query results.

Recently, with the rapid development of the cloud computing and the sharp increase of organization or enterprise business data, data outsourcing service model promotes the further study on secure search in the cloud computing environment. Researchers have proposed many secure schemes to improve the query efficiency, enrich query functionalities, and enhance security. In these secure query schemes, the encrypted searchable indexes are constructed by the data owner, though which an authorized data users can obtain qualified data by submitting encrypted keyword(s) to the cloud.

We will introduce secure query techniques based on the index construction in the symmetric encryption setting in this section. (Note that some public-key based secure query schemes are also proposed by researchers [10], however, which are not suitable for the massive data collection in the cloud environment due to the much more expensive computation overhead of the asymmetric encryption).

We assume that the data owner owns the file set $F=\{f_1, f_2, \ldots, f_n\}$ and extracts a set of keywords $W=\{w_1, w_2, \ldots, w_n\}$ from F. To realize an effective secure query scheme, the data owner constructs an encrypted searchable index for F by invoking an index generation algorithm called **BuildIndex** when taking as input a secret key K. In addition, the file set F should be encrypted for data confidentiality using an algorithm called **Enc** with the other key K'. Generally, the semantically secure symmetric encryption scheme DES or AES is a very good choice for **Enc.** Finally, the encrypted data files and encrypted indexes are stored on a remote cloud center in the following form:

$$\{I = \mathbf{BuildIndex}_k(F=\{f_1, f_2, \ldots, f_n\}; W=\{w_1, w_2, \ldots, w_n\}), C=\mathbf{Enc}_{k'}(f_1, f_2, \ldots, f_n)\}$$

When an authorized data user wants search data files that contain a certain keyword w, he/she invokes the algorithm **Trapdoor** under the same key K used to encrypted the index I to generate a so-called trapdoor (i.e., query token) $T = \mathbf{Trapdoor}_k(w)$ and submits T to the cloud via

public communication channels. With *T*, the cloud can search the index using an algorithm called **Search** and returns corresponding encrypted data files. Finally, the data user uses the algorithm **Dec** and takes as input the symmetric encryption key *K*' to decrypt query results. Figure 3.6 shows a general model of an index-based searchable encryption scheme in the cloud computing environment.

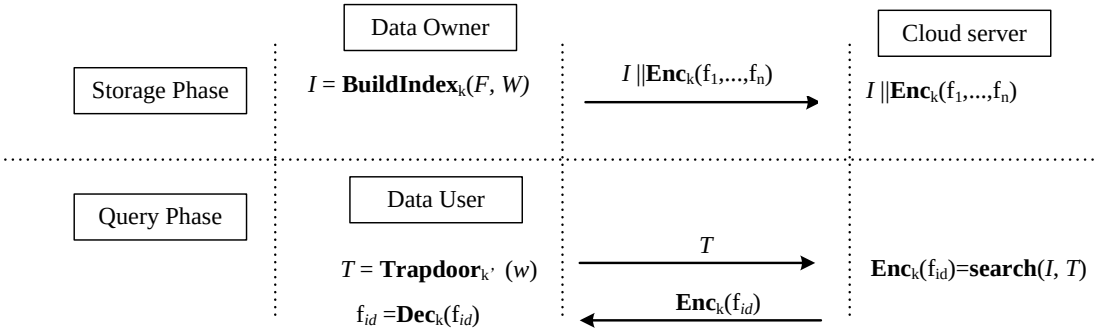

Figure 3.6 General model of an index-based searchable encryption scheme in the cloud computing environment

We formally give the definition of an index-based secure query scheme for cloud computing.

Definition 1 *An index-based secure query scheme consists of the following six polynomial time algorithms.*

- **Keygen**(1^k): The data owner takes as input a security parameter *k* and outputs two keys *K*, *K'*. The *K* is used to encrypt and decrypt the data file set and the *K'* is used to encrypt indexes and to generate query trapdoors.
- **Enc**(*K*, *F*): The data owner takes as input the secret key *K* and a data file set $F=\{f_1, ..., f_n\}$, outputs a sequence of ciphertexts $C=\{c_1, ..., c_n\}$.
- **BuildIndex**(*K'*, *F*, *W*): The data owner runs the algorithm and takes as input a key *K*', a data file set *F* and a set of keyword *W* extracted from *F*, outputs the secure index *I*.
- **Trapdoor**(*K'*, *w*): An authorized data user runs the algorithm and takes as input the key *K*' and a query keyword *w*, outputs the ciphertext form *T*(*w*) of *w*.
- **Search**(*T*(*w*), *I*): The cloud server runs the algorithm, which performs secure search on *I* according to the query trapdoor *T*(*w*) and outputs the corresponding encrypted data file

c that contains *w*.

- **Dec(*K*, *c*):** the data user invokes the algorithm to decrypt query result *c* returned by the cloud server to get the corresponding plaintext *f*.

3.2.1.4 The Security Definition of Index-based Secure Query Techniques

The definition of secure query scheme in Section 3.2.1.3 indicates that the security of outsourced data files itself can be easily achieved due to the semantically secure AES or DES encryption. However, the cloud server may infer actual contents of the data files by attacking searchable indexes, since indexes are embedded much useful information of data files. So, constructing semantically secure searchable indexes should be the most key security goal for an index-based secure query scheme. How to define the security of indexes? In [11], Eujin-Goh formally proposed the Semantic Security Against Adaptive Chosen Keyword Attack (IND-CKA) security definition, which has been widely adopted in prior secure query scheme.

Definition 2 *An index-based search scheme is semantically secure if the constructed searchable index satisfies the following two key security conditions: (1) index indistinguishability (IND); (2) security under chose keyword attacks (CKA).*

Simply speaking, when given two data files F_1, F_2 and their indexes I_1 and I_2, if the adversary (i.e., the cloud server) cannot distinguish which index is for which data file, we say the indexes I_1 and I_2 are IND-CKA secure. In other words, the IND-CKA security definition guarantees that an adversary cannot deduce and obtain any useful plaintext information of data files from the searchable index.

3.2.1.5 The Effective Implementations of Index-based Secure Query Techniques

In this section, we will introduce two effective and efficient query techniques over the encrypted data files, which have been widely used in cloud computing environments. One is a single-keyword secure query scheme based on the encrypted inverted index structure, the other is a multi-keyword secure query scheme based on the encrypted vector space model.

1. An Efficient Single Keywords Secure Query Scheme

In [4], Curtmola *et al* constructed an efficient single keywords secure query scheme based on encrypted inverted indexes. In practice, the inverted index is widely used to the keyword search in the plaintext retrieval information system due to the construction simplicity and the high search efficiency. To the best of our knowledge, almost of all modern search engines take advantage of the inverted index technique to design their internal index constructions. Figure 3.6 shows a simple inverted index structure. In this construction, a linked list L_i is used to represent a search index for a keyword item w_i. In each L_i, the header node HN_i stores the specified keyword information w_i and every intermediate node $N_{i,j}$ stores the identifier of a data file that contains w_i. For simplicity, Figure 3.7 shows a simple index structure that there is three keywords w_1, w_2, w_3 and eight data files F_1-F_8.

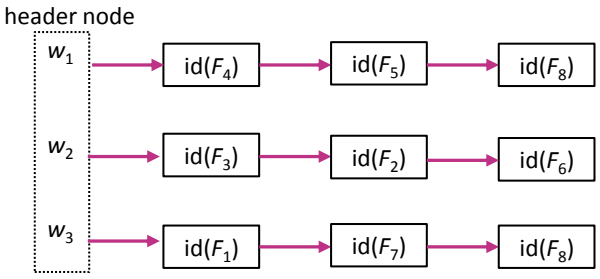

Figure. 3.7 A simple inverted index structure

Obviously, directly storing the inverted indexes to the cloud server in the plaintext form would cause the inevitable information leakages because the cloud can see all plaintexts. To avoid

gaining information from inverted indexes for any underlying attacker, Curtmola *et al* used the symmetric encryption technique to encrypted inverted indexes and made them confidential while keeping efficient searchable capabilities [4].

Now, we first introduce the proposed index encryption technique and make the indexes demonstrated in Figure 3.7 secure step-by-step and then illustrate how to perform search on these encrypted indexes by a simple example.

For ease of understanding, we introduce the implementation details of the index encryption technique by dividing the encryption processes into the following three steps.

(1) Encrypt Nodes Except for the Header Node and the First Node

For a linked list L_i of the distinct keyword w_i, except for the header node HN_i, each node $N_{i,j}$ consists of three fields: the identifier of data file containing the keyword w_i, a pointer to the next node, and a random key $K_{i,j}$ that is used to encrypt the next node. The pointer field and key field of the last node are set to be null and 0, respectively. Next, except for the first node $N_{i,1}$, each node $N_{i,j}$ is encrypted by the key $K_{i,j-1}$ using a semantically secure symmetric encryption scheme such as AES or DES. Figure 3.8 shows the above encryption processes.

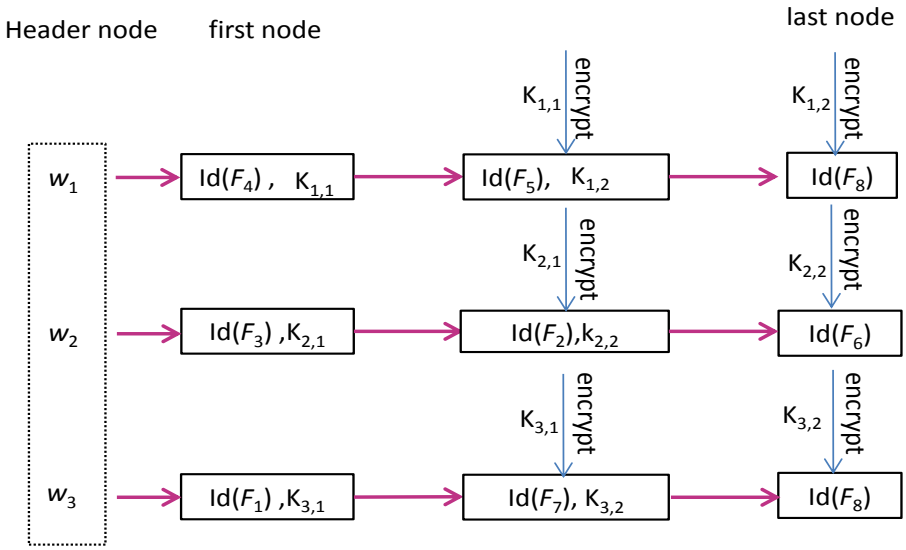

Figure. 3.8 Intermediate nodes encryption

(2) Construct the Header Node and Encrypt the First Node

The header node of each linked list L_i corresponding to the keyword w_i contains two fields, the first field stores a random key $K_{i,0}$ that is used to encrypted the first node $N_{i,1}$ and the second field is a pointer to the first node $N_{i,1}$ in L_i. Thus, the first node $N_{i,1}$ can now be encrypted by the key $K_{i,0}$. Now, except for the header node HN_i, each node in L_i is encrypted with different key stored at the corresponding previous node.

(3) Encrypt Header Node and Create Encrypted Look-Table

Obviously, the header node still discloses the key information $K_{i,0}$ for L_i, by which the cloud can decrypt all nodes of L_i and make the index very easy to crack. To hide the header node information while keeping the search ability, the proposed scheme uses an encrypted look-up table T to store the encrypted header node information. We use $HN_i = (K_{i,0}, \text{addr}(N_{i,1}))$ to denote the header node of L_i, where addr($N_{i,1}$) represents the pointer or address of first node $N_{i,1}$ in L_i. To make the header node secure to protect the key $K_{i,0}$, the scheme uses a pseudo-random function [2] f with the key k_2 to calculate the keyword w_i as $f(k_2, w_i)$, and then encrypt the header node $HN_i = (K_{i,0}, \text{addr}(N_{i,1}))$ by computing $f(k_2, w_i) \oplus HN_i$, where $\oplus$ is an exclusive or operation. Next, the scheme generates a look-up table T and uses another pseudo-random function π with the key k_1 to calculate the keyword w_i as $\pi_{k1}(w_i)$ and stores the encrypted header node $f(k_2, w_i) \oplus HN_i$ to the corresponding position $T[\pi(k_1, w_i)]$ in T.

So far, according to the above three steps, the linked list L_i of keyword w_i has been totally encrypted. In next section, we will give a simple example to illustrate how to perform query on encrypted indexes. Figure 3.9 shows a simple example of head node encryption and look-up table structure.

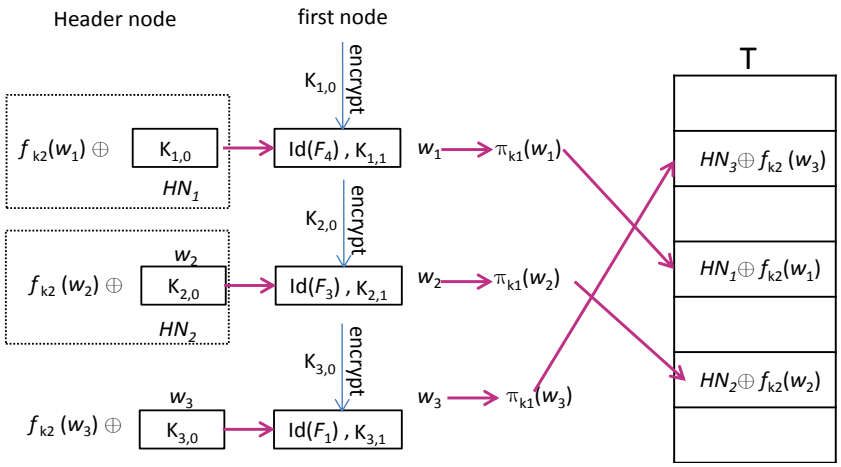

Figure 3.9 Head node encryption and look-up table structure

Lastly, we give a secure search example. Suppose that a data user wants search data files containing the keyword w_2, he/she first encrypts the query keyword w_2 using the keyed hash functions f and π to generate a query trapdoor as $trap=(\pi_{k1}(w_2), f_{k2}(w_2))$ and submits the *trap* to the cloud server. Upon receiving the *trap*, the cloud accesses the look-up table T and takes the encrypted value $val=HN_2 \oplus f_{k2}(w_2)$ from the position $T[\pi_{k1}(w_2)]$, and then decrypts the header node HN_2 by calculating $val \oplus f_{k2}(w_2)$. Last, the cloud decrypts the whole linked list L_2 starting with the header node HN_2 and outputs the list of data file identifiers in L_2.

2. An Efficient Multi-Keyword Secure Query Scheme

Obviously, the inverted index based secure query scheme efficiently support single keyword query, which needs to perform multiple rounds query to implement a multi-keyword conjunction query. In this section, we will introduce an efficient multi-keyword query scheme based on encrypted vector space model. The scheme can effectively perform multi-keyword ranked and top-*k* secure query over the encrypted cloud data.

The basic idea behind the query scheme is to use a n dimension binary data vector to represent a query index of a data file. In a multi-keyword query, the query is also denoted as a n dimension binary vector, meanwhile, the query operation is converted into the inner product

computation between the query vector and each file index vector. In this scheme, one can determine which data files are more relevant than others by sorting these inner product value in a descending order to implement a multi-keyword ranked and top-*k* query.

We first give a simple example to illustrate how to create the data file index vector, the query vector, and how to perform a multi-keyword ranked query in the plaintext setting. For ease of understanding, we assume that there is two documents D_1 (containing three keywords w_1, w_3, w_4) and D_2 (containing three keywords w_1, w_2, w_3), which are denoted as $D_1=\{w_1,w_3,w_4\}$ and $D_2=\{w_1,w_2,w_3\}$, respectively. To create the n dimensions binary vector indexes for D_1 and D_2, the scheme needs to predefine a keyword dictionary consisting of n keywords, denoted as $W = \{w_1,w_2,w_3,w_4,w_5\}$ (For simplicity, we set n=5). The binary index vector I of data file D can be created according to the dictionary W as follows: if w_i ($1 \le i \le 5$) in W appears in the data file D, the corresponding ith position is set to be 1, otherwise, the position is set to be 0. Thus, the 5 dimensions vector indexes of D_1 and D_2 can be denotes as $I_1=\{1,0,1,1,0\}$ and $I_{2=}\{1,1,1,0,0\}$, respectively. Suppose that a data user uses the keywords w_1 and w_2 to request data files, the query is also converted into a 5 dimensions query vector by adopting the above same way, denoted as $Q =\{1,1,0,0,0\}$. Now, the query is performed by computing inner product between query vector and each data index vector. In this example, $I_1 \cdot Q=1$ and $I_2 \cdot Q=2$, which indicates D_2 is more relevant than D_1. If this is a top-1 query, the D_2 should be regarded as the only query result.

Document	index vector
$D_1=\{w_1, w_3, w_4\}$	$I_1=\{1, 0, 1, 1, 0\}$
$D_2=\{w_1, w_2, w_3\}$	$I_2=\{1, 1, 1, 0, 0\}$

Query keywords	query vector
w_1, w_2	$Q=\{1, 1, 0, 0, 0\}$

Inner product
$I_1 \cdot Q= 1$
$I_2 \cdot Q= 2$

Figure 3.10 An example of multi-keyword query in the plaintext setting

We can observe from Figure 3.10, the scheme can achieve relevance ranked query

by comparing the inner product between the index vector and query vector. To implement the secure search over encrypted cloud data, the basic goal is to design an encryption function E to encrypt the file index vector and query vector while can still compare the inner product. We call the encryption function E as inner product preserving encryption function.

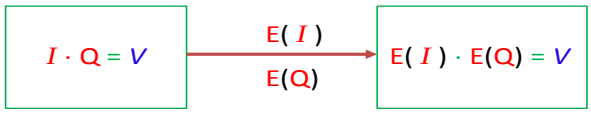

Figure 3.11 Secure inner product preserving encryption

In [5], based on the space vector model, Wong *et al* proposed a secure k-nearest neighbor (kNN) query scheme over encrypted database and Cao *et al* improved this technique and made it suitable for secure query over encrypted cloud data [6]. In this section, for simplicity, we introduce a simplest version of the encryption function E which can preserve the inner product between two encrypted vector. First, we need to define two encryption functions E_1 which is used to encrypt the n dimensions index vector I and E_2 which is used to encrypt the n dimensions query vector Q. The I and Q are represented by column vectors. Given a secret $n \times n$ invertible matrix M as the encryption key, encrypt the index vector I by computing $E_1(I)=M^T \times I$ and encrypt the query vector Q by computing $E_2(Q)=M^{-1} \times Q$. Given an encrypted index $E_1(I)$ and an encrypted query $E_2(Q)$, the search system can get the correct inner product value between I and Q by computing $[E_1(I)]^{-1} \times E_2(Q)$, this is because:

$$[E_1(I)]^{-1} \times E_2(Q) = I^{\mathrm{T}} M M^{-1} Q = I^{\mathrm{T}} \times Q = I \cdot \mathrm{Q}$$

Thus, the search system can correctly perform ranked query over the encrypted data index according to the inner product. Many security-enhanced versions based on this technique have been widely used in the multi-keyword secure query system for the cloud computing.

3.2.2 Privacy on Correlated Big Data

3.2.2.1 Correlated Data in Big Data

The data correlations in big data are very complex and quite different from traditional relationships between data stored in relational databases. There are some researchers having studied data correlations and their privacy problems.

Generally, data correlations are described by social networks graphs. Figure 3.12 shows that there is a simple data relationship in the social network. In Figure 3.12, nodes represent persons or other entities in social networks, and edges represent correlations between them. Adversaries may attack the privacy of persons who have high correlations in social networks by using external information (i.e. background knowledge) or the structural similarity between nodes and nodes or graphs and graphs. For example, adversaries can analyze target individuals' privacy according to the degree of target nodes, attributions of nodes, linking relationships between target individuals, neighborhood structures of target individuals, embedded sub-graphs, the features of graphs [12,13] and so on. Besides, adversaries can also utilize structural information of target individuals to steal their privacy [14].

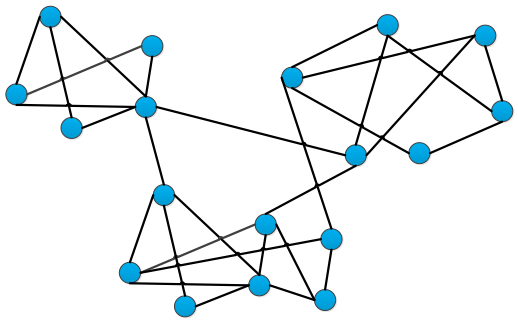

Figure 3.12 A simple social networks graph

In fact, adversaries can achieve data correlations easily from social networks, etc. For instance, the structure of the social network may be obtained when adversaries use a web crawler. Therefore, persons' correlations in social network could be ascertained by analyzing this structure.

An adversary may achieve some query results, such as "the number of persons whose hobbies are smoking and drinking", and then the adversary may try to analyze the hobby of one special person based on the fact of the high correlated persons with the same hobby [15]. Figure 3.13 shows that adversaries attack sensitive information through utilizing data correlation.

So that many attributes about correlated data in social networks may be used to analyze privacy of target individuals. And correlated data privacy will be leaked more easily, we should pay much more attention on this problem.

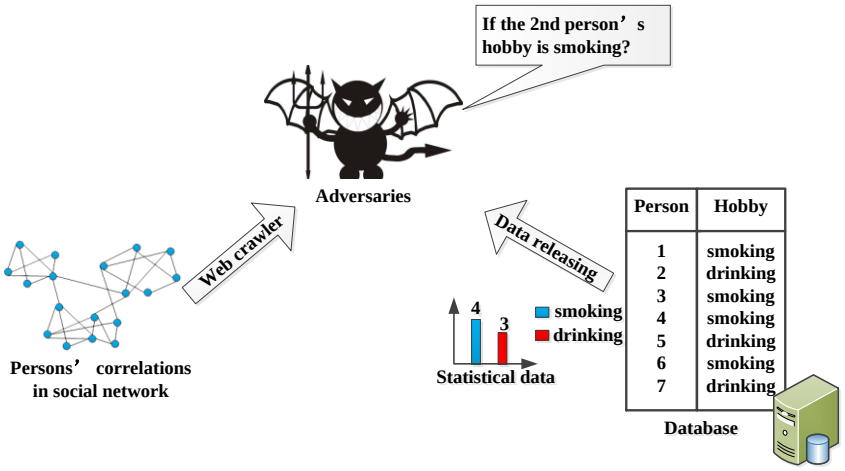

Figure 3.13 Sensitive information stolen through utilizing data correlation

However, there may be important, undetected relationships in big data. And Reshef *et al* proposed an approach called the maxim information coefficient (MIC) to measure a wide range of correlations both functional and not in large data sets **[**16**]**. But the amount of data sets in big data is huge, thus data correlations in big data are more complex. Besides, adversaries also can utilize the undiscovered data correlations to attack sensitive information. So correlated data privacy in big data is an important and growing challenge.

Furthermore, data correlations in big data does not mean that data A causes data B. On the other hand, these correlations imply that data A and B may be detected simultaneously. And these correlations are quite different from the data correlations that are in the time before big data coming. However, the difference often is neglected, so sensitive information will be exposed in

more danger.

Nowadays, some researchers are studying that how to protect sensitive information in big data. And the researches about data privacy protection focus on anonymity and differential privacy primarily.

3.2.2.2 Anonymity

Adversaries can utilize data correlation in big data to attack sensitive information. There are some existing technologies to prevent adversaries stealing sensitive information of relational data. In this section we will introduce the development of anonymity for data releasing, and data privacy protection for social networks.

1. Anonymity for Data Releasing

For the structured data in big data, the key technologies and basic methods achieving the anonymity for data releasing are still being developed and improved.

Take *k*-anonymity for example. In schemas which are proposed at the early age [17,18] and their optimization [19,20,21], data is processed by tuples generalization and restraint, thus quasi identifiers are divided into groups. Quasi identifiers in every group are converted to be the same and there are *k* tuples at least in every group. So every tuple cannot be identified by other *k*-1 tuples at least. Figure 3.14 shows the example of *k*-anonymity.

	Non-sensitive			Sensitive
No.	Zip Code	Age	Nationality	Hobby
1	410082	29	Chinese	Somking
2	410082	28	Chinese	Driking
3	410082	21	Chinese	Dancing
4	13068	49	American	Singing
5	13053	37	American	Somking
6	13053	41	American	Somking
7	12047	36	Japanese	Drinking
8	12035	35	Japanese	Dancing
9	12050	20	Japanese	Smoking
10	14850	27	Russian	Somking
11	14853	26	Russian	Drinking
12	14853	40	Russian	Smoking

Person’s Data in Social Networks

Anonymity

	Non-sensitive			Sensitive
No.	Zip Code	Age	Nationality	Hobby
1	4100**	<30	*	Somking
2	4100**	<30	*	Driking
3	4100**	<30	*	Dancing
4	130**	>30	*	Singing
5	130**	>30	*	Somking
6	130**	>30	*	Somking
7	120**	>20	*	Drinking
8	120**	>20	*	Dancing
9	120**	>20	*	Smoking
10	148**	>20	*	Somking
11	148**	>20	*	Drinking
12	148**	>20	*	Smoking

3-Anonymous Person’s Data in Social Networks

Figure 3.14 Example of k-anonymity, where k=3 and QI={ZIP, Age, Nationality}

Then *l*-diversity anonymity is proposed. The character of this method is that the diversity of

sensitive data should be more than l. Implementations of *l*-diversity contain a scheme based on anatomy and a scheme based on data substitution [22]. Figure 3.15 shows the example of *l*-diversity anonymity.

	Non-sensitive			Sensitive
No.	Zip Code	Age	Nationality	Hobby
1	4100**	<30	*	Somking
2	4100**	<30	*	Driking
3	4100**	<30	*	Dancing
4	130**	>30	*	Singing
5	130**	>30	*	Somking
6	130**	>30	*	Somking
7	120**	>20	*	Drinking
8	120**	>20	*	Dancing
9	120**	>20	*	Smoking
10	1485*	>20	*	Somking
11	1485*	>20	*	Drinking
12	1485*	>20	*	Smoking

Figure 3.15 Example of l-diversity anonymity, where l=2

Besides, there is another anonymity called *t*-closeness which requires that the sensitive data distribution of the equivalence class is the same as data distribution of the whole data table [23]. And there are the other works about anonymity include (*k*, *e*)-anonymity [24], (*X*, *Y*)-anonymity [25] and so on.

The anonymity for data releasing in big data is more complex. Adversaries can achieve data from multiple channels, not only the same data publishing source. For example, people discover that adversaries can compare the data with public imbd and then recognize the account of the object in Netflix. So this problem can be researched more deeply [26].

2. Anonymity for Social Networks

The data produced by social networks is one of big data sources, meanwhile, the data contains a mass of users privacy. Because of graph structure features of social networks, anonymity for social networks is quite different from structured data.

The typical requirements for anonymity in social networks are users identifiers

anonymity and attributes anonymity (also called nodes anonymity). When data is released, users identifiers and attributes information are hidden. Furthermore, relationship anonymity (also called edges anonymity) is the other typical requirement for anonymity in social networks. The relationships between users are hidden when data is published.

Currently, edges anonymity schemes are mostly based on edges deleting and adding. And edges anonymity can be achieved by adding and deleting exchange-edges randomly. In [27], in the process of anonymity, characteristic values of adjacent matrixes and the second characteristic values of the according Laplace matrixes are maintained constant. In [29], nodes are classified according the degree of nodes. Edges satisfied requirements are chosen from the nodes of the same degree. And the solutions in [30, 31] are similar to that in [28].

Besides, an important idea is that the graph structure is partitioned and clustered based on super nodes. For instance, anonymity based on nodes clustering, anonymity based on genetic algorithm, anonymity based on simulated annealing algorithm and a schema for filling in the splitted super nodes [31]. In [32], the concept of *k*-security is proposed. It achieves anonymity by *k* isomorphism subgraphs. Anonymization based on super nodes can achieve edges anonymity, but deposed social structured graphs are quite different from the original ones.

However, an important problem of the social networks anonymization is that adversaries can use the other public information to infer anonymous users, specially, there are connection relations between users. There are such problems as follows. Firstly, adversaries utilize weak connections to predict the possible connections between users [32]. This is appropriate for networks in which there are sparse user relations. Secondly,

attackers can regain and infer hierarchical relationships in the crowd, according to existing social networks structures [34]. Secondly, adversaries can analyze the complex social networks such as Weibo and infer relations [35]. At last but not the least, adversaries can infer the probability of different connection relations based on the restricted random-walk mechanism. Researches consider that the clustering feature of social network will make a significant impact on the correctness of sub-relationship prediction.

3.2.2.3 Differential Privacy

Differential privacy [36, 37] is a powerful tool for providing privacy-preserving noisy query answers over statistical databases. And it also is the most popular perturbation for correlated data privacy protection. It guarantees that the distribution of noisy query answers changes very little with the addition or deletion of any tuple.

1. Definitions of Differential Privacy

There are two typical definitions of differential privacy, which are classified into unbounded and bounded [38]. The formal definitions of Unbounded Differential Privacy [36] and Bounded Differential Privacy [37] are definition 3 and definition 4 respectively.

Definition 3 *Let A be a randomized algorithm, S be any set, D_1 and D_2 be any pairs of databases, A will be unbounded ε-differential privacy, if* $P(A(D_1) \in S) \leqslant e^{\varepsilon} P(A(D_2) \in S)$, *where D_1 could be achieved from deleting or adding one tuple of D_2.*

Definition 4 *Let A be a randomized algorithm, S be any set, D_1 and D_2 be any pairs of databases, A will be unbounded ε-differential privacy, if* $P(A(D_1) \in S) \leqslant e^{\varepsilon} P(A(D_2) \in S)$, *where D_1 can be obtained from changing the value of exactly one tuple of D_2.*

2. Differential Privacy Optimization

To improve the precision of inquiry response, there are lots of researches on differential

privacy.

Blum *et al* proposed a new approach which can response arbitrary "querying class" [39]. And the precise of the response relies on the VC dimension of the "querying class". In [39**]**, an efficient method for the querying of responsing domain，but the precision is worse than that proposed in [40,41].

Differential privacy for compound query, Hardt *et al* proposed a *k*-mod method based on linear query sets [42]. This method makes linear query mapped to L_1 sphere, and then the noise is decided that how much to be added. And this method is required uniformly sampling from high dimensional convex body. In [43], the cost of the algorithm is searching for the query schema mainly. This procedure is proceeded only once. Once the best schema is determined, its efficiency is the same as Laplace's.

Roth *et al* supposed that query will arrive in chronological sequence in an interactive environment [44]. In the condition that the query which will arrive is unknown, the query which will arrive must be responded immediately. And the traditional Laplace mechanism is improved. The query response is got by computing prior query results using a median approach.

(1) Query Response Optimization based on the Hierarchy Concept of the Matrix Mechanism

Relevant researches about the best response schema for the count query in the related domain almost aren't reported. Related researches are gotten down to research since 2010.

Xiao *et al* applied Haar wavelet transform (WT) to differential privacy [40]. Data is carried out wavelet transform before noise added, so that the correctness of the counting query can be improved. At the same time, the applied range is expanded to the attributions of the nouns and the high-dimensional Histogram query. This approach is equivalent to the H query no matter in theory or the result, and is the equivalent to a query tree. And the summaried data which is more fine-grained is on the every lay of this tree from up to down.

Hay *et al* proposed differential privacy based on layered summation and least squares [41]. The query sequence is divided into groups that are satisfied consistency constraints. And noises

are added group by group, so that the precision of the query is satisfied and the amount of noised should be added is reduced.

Li *et al* proposed the conception of the matrix mechanism [43], and paper [40,41] are brought into the category of the matrix mechanism. Some relationship between the queries is analyzed, and each query is regarded as linear combinations of basic count operations. Then the queries containing relationship are expressed as the matrixes. So that the amount of the added noise is reduced based on the existing Laplace noise distribution.

These three approaches above, are all based on the matrix mechanism. And special query questions are improved by them.

(2) The Precision Range of the Linear Query

Since differential privacy is proposed, the range of the precision of the query response that is satisfied differential privacy at the same time, becomes an important research field studied by researchers. And the most important is the range of the precision of the linear query response.

In the non-interactive mode, Blum *et al* proposed differential privacy for releasing synthetic data [39]. This algorithm obeys the error range of $O(n^{2/3})$, where n is the size of the data set, and the error range will be expand with the increasing of the data sets. Dwork *et al* improved the algorithm [45, 46**]** proposed in paper [39]. Gupta *et al* proposed a non-interactive query response based on multiplicative weights and agnostic learning [47]. Hardt *et al* implemented these algorithms above and evaluated them by experiments in [48]. The experiments' results show that this algorithm is not adjusted to the interactive mode, as well as when the number of the query times is many times as the size of the data sets, the efficiency of the algorithm is very low.

In the interactive mode, Roth *et al* [44] assumed that the query was executed online, and the query should be responded before the upcoming query arrived. So that the error range will be $n^{2/3}(logk)(log|X|)^{1/3}/\varepsilon^{1/3}$. Hardt *et al* [49] adapted an approach based on additive and multiplicative weights, and the error edge of the query response is reduced to $(n^{1/2}(logk)(log|X|)^{1/4})/n$. And the

meaning of the parameters is: under ε-differential privacy protection, n is the size of the data sets, k is the query times and X is the data domain.

Gupta *et al* composited the mid-value algorithm [44] and additive and multiplicative weights [49] in the interactive mode, and summarizes the multiplicative weights [47,48] in the non-interactive mode, as well as defines iterative database construction (IDC) algorithm abstractly [50]. IDC algorithm can respond queries under differential privacy protection either in the interactive mode or in the non-interactive mode. An assumed data structure is improved iteratively by IDC algorithm, so that this data structure can respond all the queries finally. At the same time, the algorithm improves the precision range of the mid-value algorithm and the multiplicative weights algorithm.

3. Key approaches for differential privacy

(1) Differential Privacy for Histogram

As the foundation of many data analysis approaches, such as contingency tables and marginals, the histogram query plays an important role on the differential privacy protections.

Chawla *et al* began to study privacy protections for histogram queries before the differential privacy proposed **[**52**].** Dwork *et al* analyzed the essential properties of the histogram query deeply, and pointed out that comparing with the existing methods, the biggest advantage of the differential privacy is that the computation of the sensitiveness is independent of the dimension [52]. With regard to the differential privacy for some high-dimensional output contingency table and the covariance matrix analysis, data privacy protections can be on the premise that noises is add a little. Dwork *et al* gave a complete summary about achieving the differential privacy for the histogram query, and pointed out that although the histogram containing k units can be regarded as k independent counting queries, a line of data adding or deleting just only effects the counting of that unit corresponding that line of data, and at most one count is affected [53]. Therefore the sensitiveness of a histogram query is 1. This conclusion makes a foundation of applying the differential-private histogram protection to various problems.

(2) The differential-private K-means Clustering

Blum *et al* proposed that the differential privacy was achieved by adding appropriate noises in the process of the K-means clustering, and summarized the sensitiveness computing method of the query function and the main steps for achieving ε-differential private K-means algorithm [54]. In K-means, privacy will be leaked by computing the central point which is nearest to every sample point. However, for an unknown set, computing the average value only needs the sum to divide by the number. Therefore, as long as the approximate value of the set S_j is released, and the set's own information is not needed.

Dwork *et al* replenished and improved the algorithm above, and analyzed the sensitiveness computing method of each query function in the differential-private K-means algorithm in detail, gave the total sensitiveness of the whole query sequence [53].

4. The PINQ Framework

PINQ (privacy integrated queries) is developed by Dwork *et al*, a framework of sensitive date privacy protection [55]. The PINQ framework is similar to the LINQ framework. And it provides a series of APIs, so that developers can develop the differential privacy protection system through using the simple and expressive language. There are rich application cases of the differential-private data analysis in this framework.

Figure 3.16 shows an application example of releasing data by the PINQ framework. In the framework, a privacy budget is set by the owner of the database which need be protected. When users query data from the database, some budgets will be consumed. Once the budget is exhausted, users cannot query by the framework. Supposing a record in database is "Jerry, male, pneumonia, 28-years old, address: yuelu mountain at Changsha of Hunan Province in China". For data users, in the range of the parameter ε, whether Jerry is in the database or not, the query results will not be affected. The query results can be shared between users freely. And users also can integrate the external information (such as the building owner information, the inhabitant's consuming data and the other data from the other database). The privacy will not be leaked by

these operations.

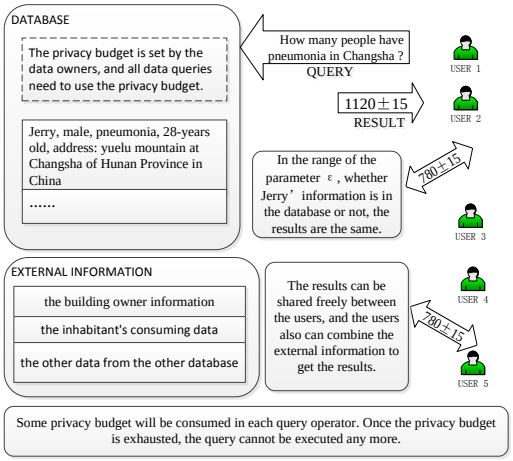

Figure 3.16 The PINQ framework

5. Differential Privacy for Correlated Data Publication

To protect correlated data, differential privacy is studied to solute this problem.

Chen *et al* proposed an approach to protect correlated network data while the data is released. This approach is differential privacy by adding an extra correlation parameter. And it is a non-interactive method to stop adversaries analyzing the presence of a direct link between two persons. Meanwhile, researchers gave the theorem of the ε-differentially private mechanism under correlations, see theorem 1[56].

Theorem 1 *Let $\mathcal{A}$ be private mechanism over a database, and k_1 be a correlation parameter which is the database with, besides k_2 is another correlation parameter and $1 \leqslant k_2 \leqslant k_1$. So $\mathcal{A}$ will be ε-differentially private for all databases with k_2.*

What's more, Zhang *et al* researched differentially privacy for privacy-preserving data releasing via Bayesian network and proposed a different solution called "*PRIVBAYES*" to deal with this problem [57]. To depict the correlations of the attribution in a dataset *D*, a Bayesian network is built. Then noise is added to a set of marginals of *D* for achieving differential privacy. At last, an approximate data distribution is constructed by utilizing the noisy maginals and the Bayesian network. And this data distribution is published for researching.

Yang *et al* studied a perturbation method on correlated data called "Bayesian differential privacy" to deal with the privacy protection of correlated data. The definition of Bayesian differential privacy is as follows [15].

Definition 6 *Let $\mathcal{A}$ be an adversary and $\mathcal{M}(x)$ be a randomized perturbation mechanism over a database D, $\mathcal{K}$ be the set of known tuples, $\mathcal{U}$ be the set of unknown tuples, i be the tuple adversaries want to attack, x be an instance of the database D, where an adversary with knowledge $\mathcal{K}$ to attack x_i, i.e. $\mathcal{A}=\mathcal{A}(i,\mathcal{K})$ and $\mathcal{K}\subseteq[n]\backslash\{i\}$,* and *$\mathcal{M}(x)=Pr(r\in S|x)$. The privacy leakage of $\mathcal{M}$ is $BDPL_{\mathcal{A}}(\mathcal{M}) := \underset{x_i,x'_i,x_{\mathcal{K}},t}{SUP} \log\frac{\Pr(r=t\mid x_i,x_{\mathcal{K}})}{\Pr(r=t\mid x'_i,x_{\mathcal{K}})}$, there are two conditions as follows.*

- *x* is continuous-valued: $Pr(r\in S\mid x_i,x_{\mathcal{K}})=\int Pr(r\in S\mid x)\,p(x_{\mathcal{U}}\mid x_i,x_{\mathcal{K}})d_{x_{\mathcal{U}}}$
- *x* is discrete-valued: $Pr(r\in S\mid x_i,x_{\mathcal{K}})=\sum_{x_{\mathcal{U}}} Pr(r\in S\mid x)\Pr(x_{\mathcal{U}}\mid x_i,x_{\mathcal{K}})$

If $\sup_{\mathcal{A}} BDPL_{\mathcal{A}}(\mathcal{M})\le\varepsilon$, $\mathcal{M}$ is Bayesian differential privacy.

Then to depict the data correlation, Gaussian Markov random field is adopted to build Gaussian Correlation Model. The definition of Gaussian Correlation Model is as follows.

Definition 7 *Let $W=(w_{ij})$ $(w_{ij}\ge 0)$ be the weighted adjacent matrix, G(x, W) be a weighted undirected graph, where each vertex x_i represents a tuple i in the dataset and $x_i\in x$, w_{ij} represents the correlation between tuples i and j. And let $DI=diag(w_1,w_2,\cdots,w_n)$ be the weighted degree matrix of G(x,W) where $w_i=\sum_{j\neq i}w_{ij}$, and L be the Laplacian matrix of G(x,W),*

$$L=DI-W=\begin{pmatrix} w_1 & -w_{12} & \cdots & -w_{1,n} \\ -w_{12} & w_2 & \cdots & -w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ -w_{1,n} & -w_{2,n} & \cdots & w_n \end{pmatrix}$$

A Gaussian correlation model $G(x, L)$ is the pair (x, L).

3.3 Future Research Directions

The existing researches about security and privacy in big data are not very perfect. There are some challenges to face. And there are future research directions as follows.

On the one hand, these secure query schemes which we introduce above allow cloud users to obtain encrypted files by submitting query keywords of interest to the cloud server. However, there are still many important techniques and functionalities of secure search to be worth further researching for continuously improving user query experience and perfecting query functionality, such as keyword fuzzy query, range query, subset query, and query expression relation operation, etc. In addition, recently, researchers propose new application requirements that a secure query scheme should provide some mechanisms that allow data users to verify the correctness and completeness of query results returned by the cloud, based on the assumption that cloud may intentional omit some qualified query results for saving computation resource and communication overhead. Verifiable secure query scheme will be research focus in the future.

On the other hand, with the increasing of the local link density in social network and the clustering coefficient, the link prediction algorithm between correlated data in social network should be further enhanced. Thus in the future, anonymity for correlated data in social network can prevent this kind of attacks by inferring. Besides, the amount of big data could be extremely high in the event of years of data collection. And the essence of the differential privacy is adding noise to protect sensitive information so that the utility of big data and the efficiency the differential privacy all will be affected more or less. Thus there is another future research that how to perform the differential privacy appropriately for big data.

3.4 Conclusion

With the growing popularity of cloud computing and sharp increase of data volumes, more and more encrypted data is stored at a cloud server, which promote the further development of secure query technique and make this technique vital and essential for data utilization in cloud

environments. At present, secure query schemes over encrypted data can be classified into two categories: the symmetric-key based solutions and the public-key based query schemes. Compared with the symmetric-key, due to the heavy computation cost and inefficiency of public-key cryptosystem, secure query schemes based on public-key setting are not suitable for the cloud applications with mass users and mass data and make those schemes based on symmetric-key predominant in all solutions.

We systemically illustrate the related concepts, models, definitions, and important implementation techniques of secure search over encrypted cloud data based on symmetric-key setting, including system framework, threat model, attack model, etc. We give the implementation principles and details of secure search technique over encrypted data by introducing two concrete secure query schemes proposed in [4] and [6].

What's more, correlated big data brings a new security and privacy problems, but it is an important solution for these problems by itself. In this chapter, we systematically comb the related key researches about correlated data privacy protection from different perspectives such as data releasing, social networks, correlated data publication and so on. And we group the relevant researches into two categories: anonymity and differential privacy, then we introduce these researches respectively. Besides, we look forward to the future works about correlated data privacy in big data. But existing researches about privacy on correlated big data are little. Thus we should pay more attention to study technical means for preserving privacy of correlated big data. The problem of privacy on correlated big data could be solved well only through a combination of technical means and relevant policies and regulations.

References

1. M. Li, S. Yu, Y. Zheng, K. Ren and W. Lou, Scalable and secure sharing of personal health records in cloud computing using attribute-based encryption, *IEEE TPDS*, 24(1): 131–143 (2013).
2. M. Bellare and P. Rogaway, Introduction to modern cryptography, Lecture Notes, 2001.
3. D. Song, D. Wagner and A. Perrig, Practical Techniques for Searches on Encrypted Data, *IEEE Symposium on Security & Privacy*, 0044 (2002)

4. R. Curtmola, J. Garay, S. Kamara and R. Ostrovsky, Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions, in *Proceeding of the 13th ACM Conference on Computer and Communications Security*, Alexandria, USA, October 30-November 3, 2006.
5. W. K. Wong, D. W. Cheung, B. Kao and N. Mamoulis, Secure kNN Computation on Encrypted Databases, in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, Providence , USA, June 29-July 2, 2009.
6. N. Cao, C. Wang, M. Li, K. Ren and W. Lou, Privacy-preserving multi-keyword ranked search over encrypted cloud data, in *Proceedings of the 30th IEEE International Conference on Computer Communications*, Shanghai, China, April 10-15, 2011.
7. W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou and H. Li, Privacy-preserving Multi-keyword Text Search in the Cloud Supporting Similarity based Ranking, in *Proceedings of the 8th ACM Symposium on Information, Computer and Communications Security*, Hangzhou, China, May 8-10, 2013.
8. C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, Secure ranked keyword search over encrypted cloud data, in *Proceedings of the 30th International Conference on Distributed Computing Systems*, Genoa, Italy, June 21-25, 2010.
9. A. Singhal, Modern information retrieval: A brief overview, *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering,* 24(4): 35-43 (2001).
10. D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, Public Key Encryption with Keyword Search, *Advances in Cryptology - EUROCRYPT* 2004, 2004: 506-522 (2004).
11. Eujin-Goh, Secure indexes, *Stanford University Technical Report*, 2004.
12. Zhou Bin, Pei Jian, Luk W S. A Brief Survey on Anonymization Techniques for Privacy Preserving Publishing of Social Network Data, *ACM Sigkdd Explorations Newsletter*, 2008, 10(2): 12-22 (2008).
13. Liu Kun and Terzi E, Towards Identity Anonymization on Graphs, in *Proceeding of ACM SIGMOD International Conference on Management of Data*, Columbia, Canada, June 9-12, 2008.
14. Backstrom L, Dwork C, Kleinberg J, Wherefore Art Thou R3579X? Anonymized Social Network, Hidden Patterns, and Structural Steganography, in *Proceedings of the 16th International Conference on World Wide Web Conference*, Banff, CANADA, May 8-12, 2007.
15. Bin Yang, Issei Sato, Hiroshi Nakagawa, Bayesian Differential Privacy on Correlated Data, in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, Melbourne, Australia, May 31-June 4, 2015.
16. David N Reshef, Yakir A Reshef, Hilary K Finucane and et al, Detecting Novel Associations in Large Data Sets, *Science*, 334 (6062): 1518-1524 (December 2011).
17. Sweeney L, k-anonymity: A model for protecting privacy, *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5): 557-570 (2002).
18. Sweeney L. k-anonymity: Achieving k-anonymity privacy protection using generalization and suppression. *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5): 571-588 (2002).
19. Bayardo R J, Agrawal R, Data Privacy Through Optimal k-anonymization, in *Proceedings of the 21st International Conference on Data Engineering*, Tokyo, Japan, April 5-8, 2005.
20. Kristen LeFevre, David J DeWitt, Raghu Ramakrishnan. Incognito: Efficient full-domain k-anonymity, in *Proceedings of the 2005 ACM SIGMOD International Conference on Management of data*, Baltimore, USA, June 14-16, 2005.
21. Kristen LeFevre, David J DeWitt, Raghu Ramakrishnan, Mondrian Multidimensional k-anonymity, in *Proceedings of the 22nd International Conference on Data Engineering*,

Georgia, USA, April 3-8, 2006.
22. Ashwin Machanavajjhala, Daniel Kifer, jOhannes Gehrke and Muthuranmakrishnan Venkitasubramaniam, L-diversity: Privacy beyong k-anonymity, *ACM Transactions on Knowledge Discover from Data*, 1(1): 1-52 (2007).
23. Ninghui Li, Tiancheng Li, Venkatasubramanian S, *t*-closeness: Privacy beyond k-anonymity and l-diversity, in *Proceedings of the IEEE 23rd International Conference on Data Engineering*, Istanbul, Turkey, April 15-20, 2007.
24. Zeng K, Publicly verifiable remote data integrity, in *Proceedings of the 10th International conference on information and communications security*, Birmingham, UK, October 20-22, 2008.
25. Ke wang and Benjamin C M Fung, Anonymizing Sequential Releases, in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Philadelphia, USA, August 20 – 23, 2006.
26. Nasayanan A and Shmatikov V, Robust De-anonymization of Large Sparse Datasets, in *Proceedings of the 2008 IEEE symposium on security and privacy*, Oakland, USA, May 18-21, 2008.
27. Ying X and Wu X, Randomizing Social Networks: A spectrum preserving approach, in *Proceedings of the 8th SIAM International Conference on Data Mining*, Atlanta, USA, April 24-26, 2008.
28. Lijie Zhang and Weining Zhang, Edge Anonymity in Social Network Graphs, in *Proceedings of the 12th International Conference on Computational Science and Engineering*, Vancouver, Canada, August 29-31, 2009.
29. Na Li and Sajal K Das, Applications of k-anonymity and l-diversity in Publishing Online Social Networks, *Security and Privacy in Social Networks*, 153-179 (2013).
30. Lei Zhou, Lei Chen and M Tamer Ozsu, K-automorphism: A general framework for privacy preserving network publication, in *Proceedings of the 35th International Conference on Very Large Databases*, Lyon, France, August 24-27, 2009.
31. Lijie Zhang and Weining Zhang, Efficient Edge Anonymization of Large Social Graph, *http://venom.cs.utsa.edu/dmz/techrep/2011/CS-TR-2011-004.pdf*, 2013,6,10
32. James Cheng, Ada Wai-Chee Fu and Jia Liu, k-isomorphism: Privacy preserving network publication against structural attackes, in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, Indianapolis, USA, June 6-11, 2010.
33. L Lu and T zhou, Link Prediction in Weighted Networks: The role of weak ties, *Europhysics Letter*s, 89(1): 18001-18006 (2010).
34. A Clauset, C Moore, M E J Newman, Hierarchical Structure and the Prediction of Missing Links in Networks, *Natrure*, 453: 98-101 (May 2008).
35. Dawei Yin, Liangjie Hong, Xiong Xiong and Brian D Davison, Link Formation Analysis in Microblogs, in *Proceedings of the 34th international ACM SIGIR Conference on Research and Development in Information Retrieval*, Beijing, China, July 24-28, 2011.
36. C Dwork, Differential privacy, *ICALP*, 26(2):1-12 (2006).
37. C Dwork, F McSherry, K Nissim and A Smith, Calibrating Noise to Sensitivity in Private Data Analysis, in *Proceedings of the 3rd Conference on Theory of Cryptography*, New York, USA, March 4-7, 2006.
38. Daniel Kifer, Ashwin Machanavajjhala, No Free Lunch in Data Privacy, in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*, Athens, Greece, June 12-16, 2011.
39. Blum A, Ligett K and Roth A, A Learning Theory Approach to Non-Interactive Database Privacy, in *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, Victoria, Canada, May 17-20.

40. Xiao Xiaohui, Wang Guozhang and Gehrke J, Differential Privacy Via Wavelet Transforms. *IEEE Transaction on Knowledge and Data Engineering*, 23(8): 1200-1214 (2011).
41. Hay M, Rastogi V, Miklau G and et al, Boosting the Accuracy of Differentially Private Histograms through Consistency, *Journal of the VLDB*, 3(1-2):1021-1032 (2010).
42. Hardt M, Talwar K, On the Geometry of Differential Privacy, in *Proceedings of the 42nd ACM Symposium on Theory of Computing*, Cambridge, MA, June 6-8, 2010.
43. Li Chao, Hay M, Rastogi V and et al, Optimizing Linear Counting Queries under Differential Privacy, in *Proceedings of the 29th ACM SIGMOD-SIGACT-SIGART Symposium on Priciples of Database Systems*, Indianapolis, USA, June 6-11, 2010.
44. Roth A and Rougharden T,. Interactive Privacy Via the Median Mechanism, in *Proceedings of the 42nd ACM Symposium on Theory of Computing*, Cambridge, MA, June 6-8, 2010.
45. C Dwork, M Naor, O Reingold and et al. On the Complexity of Differentially Private Data Release: Effcient algorithms and hardness results, in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, Bethesda, USA, May 31-June 2, 2009.
46. Dwork C, Rothblum G N and Vadhan S, Boosting and Differential Privacy, in *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, Las Vegas, USA, October 24-26, 2010.
47. Gupta A, Hardt M, Roth A and et al, Privately Releasing Conjunctions and the Statistical Query Barrier, in *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing*, San Jose, USA, June 6-8, 2011.
48. Hardt M, Ligett K and McSheery F, A Simple and Prictical Algorithm for Differentially Private Data Release, *http://www.cs.princeton,edu/~mhardt/pub/mwem.pdf*, 2012-03-15.
49. Hardt M and Rothblum G N, A Multiplicative Weights Mechanism for Privacy Preserving Data Analysis, in *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, Vegas, USA, October 23-26, 2010.
50. Gupta A, Roth A and Ullman J, Iterative Constructions and Private Data Release, in *Proceedings of the 9th International Conference on Theory of Cryptography*, March 19-21, Taormina, Italy , 2012.
51. Chawla S, Dwork C, McSheery F and et al, On the Utility of Privacy-Preserving Histograms, in *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*, Edinburgh, Scotland, July 26-July 29, 2005.
52. Dwork C, McSherry F, Nissim K and et al, Calibrating Noise to Sensitivity in Private Data Analysis, in *Proceedings of the 3rd International Conference on Theory of Cryptogrphy*, New York, USA, March 4-7, 2006.
53. Dwork C, A Firm Foundation for Private Data Analysis, *Communications of the ACM*, 54(1): 86-95 (2011).
54. Blum A, Dwork C, McSherry F and et al, Practial Privacy: The SuLQ framework, in *Proceedings of the 24th ACM SIGMOD international conference on Management of Data/Principles of Database Systems*, New York, USA, June 13-16, 2005.
55. McSherry F, Privacy integrated queries (PINQ), *http://research.microsoft.com/en-us/projects/PINQ/*, 2015-6-10.
56. Jun Zhang, Graham Cormode, Cecilia M Procopiuc and et al, PrivBayes: Private Data Release via Bayesian Networks, in *Proceedings of the Acm Sigmod International Conference on Management of Data, 2014*, Snowbird, USA, June 22-27, 2014.
57. Rui Chen, Benjamin C M Fung, Philip S Yu and et al, Correlated Network Data Publication via Differential Privacy, VLDB JOURNAL, 4(23): 653-676 (April 2014).