Ensuring Security and Privacy Preservation for Cloud Data Services

JUN TANG, Tsinghua University; Zhengzhou Institute of Information Science and Technology
YONG CUI and QI LI, Tsinghua University
KUI REN, State University of New York at Buffalo
JIANGCHUAN LIU, Simon Fraser University
RAJKUMAR BUYYA, University of Melbourne

13

With the rapid development of cloud computing, more and more enterprises/individuals are starting to outsource local data to the cloud servers. However, under open networks and not fully trusted cloud environments, they face enormous security and privacy risks (e.g., data leakage or disclosure, data corruption or loss, and user privacy breach) when outsourcing their data to a public cloud or using their outsourced data. Recently, several studies were conducted to address these risks, and a series of solutions were proposed to enable data and privacy protection in untrusted cloud environments. To fully understand the advances and discover the research trends of this area, this survey summarizes and analyzes the state-of-the-art protection technologies. We first present security threats and requirements of an outsourcing data service to a cloud, and follow that with a high-level overview of the corresponding security technologies. We then dwell on existing protection solutions to achieve secure, dependable, and privacy-assured cloud data services including data search, data computation, data sharing, data storage, and data access. Finally, we propose open challenges and potential research directions in each category of solutions.

CCS Concepts: • **Networks** → **Cloud computing**; • **Information systems** → **Data encryption**; • **Security and privacy** → *Public key encryption*; *Access control*; *Management and querying of encrypted data; Privacy protections;*

Additional Key Words and Phrases: Cloud computing, cloud data service, data security, privacy preservation

ACM Reference Format:
Jun Tang, Yong Cui, Qi Li, Kui Ren, Jiangchuan Liu, and Rajkumar Buyya. 2016. Ensuring security and privacy preservation for cloud data services. ACM Comput. Surv. 49, 1, Article 13 (June 2016), 39 pages.
DOI: http://dx.doi.org/10.1145/2906153

1. INTRODUCTION

Cloud computing, a new deployment and delivery model of computing resources, enables convenient network access to a virtualized pool of remote resources [Armbrust et al. 2010]. Its benefits include on-demand self-service, unlimited resource pooling,

This work is supported by the National Natural Science Foundation of China (under grants 61120106008, 61422206, and 61572278) and US National Science Foundation (under grant CNS-1262277).
Authors' addresses: J. Tang and Y. Cui (Corresponding Author), FIT Building, Tsinghua University, Beijing 100084, China; emails: tangj13@mails.tsinghua.edu.cn, cuiyong@tsinghua.edu.cn; J. Tang, Zhengzhou Institute of Information Science and Technology, Science Avenue, Zhengzhou 450001, China; email: tang_jun_76@163.com; Q. Li, Graduate School at Shenzhen, Tsinghua University, Lishui Road, Nanshan District, Shenzhen 518055, China; email: qi.li@sz.tsinghua.edu.cn; K. Ren, 317 Davis Hall, University at Buffalo, State University of New York, Buffalo, NY, USA; email: kuiren@buffalo.edu; J. Liu, TASC 9005, Simon Fraser University, Burnaby (Metro-Vancouver), BC, Canada; email: jcliu@cs.sfu.ca; R. Buyya, Department of Computing and Information Systems, University of Melbourne, Grattan Street, Parkville, Melbourne, VIC 3010, Australia; email: rbuyya@unimelb.edu.au.

DOI: http://dx.doi.org/10.1145/2906153

Table I. Security Threats to Cloud Computing

CSA [2013]	Ko et al. [2013]	Verizon[2015]
. Data Breaches	. Hardware Failure	. Point-of-Sale (POS) Intrusions
. Data Loss	. Natural Disasters	. Crimeware
. Account Hijacking	. Closure of Cloud Service	. Cyber-Espionage
. Insecure APIs	. Cloud-Related Malware	. Insider and Privilege Misuse
. Denial of Service	. Inadequate Infrastructure Design	. Web App Attacks
. Malicious Insiders	˜ and Planning	. Miscellaneous Errors
. Abuse of Cloud Services		. Physical Theft/Loss
. Insufficient Due Diligence		. Payment Card Skimmers
. Shared Technology Issues		. Denial of Service (DoS) Attacks
Note: These threats were identified by conducting a survey of industry experts in 2013.	*Note*: These threats were identified from cloud vulnerability incidents between January 2008 and February 2012.	*Note*: These threats were identified from data breach incidents that occurred in 2014.

broad network access, dynamic scalability, service-measured pricing, and alleviation of management risks [Xiao and Xiao 2013]. With all these benefits, cloud computing motivates individual and enterprise users to intentionally outsource their local data to remote servers hosted by a Cloud Server Provider (CSP), that is, data storage outsourcing. According to the data usage forecast by Gartner in 2012 [Verma 2012], users would store more than a third of their data to the cloud by 2016. Besides data storage service, users further expect to acquire more related services from the cloud, such as data search, data computation, data sharing, and data access. These cloud data services[1] could easily help users avoid large capital outlays and operational overheads for purchasing devices and managing them.

A cloud is referred to as an untrusted cloud environment when its resources and services are open for public use and communication is performed over a nontrusted network. Generally, a public cloud (e.g., Amazon AWS, Microsoft Azure, and Google Cloud Platform) is not fully trusted by users. Therefore, although the benefits of this new cloud service paradigm are tremendous, serious security risks and privacy challenges are raised under untrusted cloud environments. Particularly, when data owners outsource their data to a public cloud, they will lose tight control of the data as in their local storage systems. Curious cloud administrators and unauthorized users may deliberately access the outsourced data and obtain the sensitive information for various motivations. The investigation report of Verizon in 2015 [Verizon 2015] indicates that infamous data breach incidents occurred from time to time in recent years. Moreover, data corruption or loss could also happen in cloud servers because of failures incurred by improper configuration, software bugs, hardware errors, and power failures [Ko et al. 2013]. To save storage space and minimize costs, greedy CSPs may discard the data that are never or rarely accessed by users, which may impact data retrievability for users. The report from the Cloud Security Alliance (CSA) [CSA 2013] shows that the data security problems are among the top threats in the cloud. Table I lists the security threats to cloud computing identified by CSA and Verizon [CSA 2013; Ko et al. 2013; Verizon 2015]. Additionally, when users use cloud data services, privacy breaches often occur due to undesirable interference from internal and external adversaries; for example, a CSP can guess a user is ill by observing his or her access to certain medical data. Thus, it can be seen that cloud data services are intrinsically not secure from

[1]In this article, we take different data processing and management operations in the cloud as typical operations enabled by cloud data services, including data storage, data search, data computation, data sharing, and data access.

the viewpoint of cloud users. If there are no effective security and privacy protection measures, it would be hard to believe that cloud users will delegate a CSP to manage their data only based on cost reductions and flexible services.

To alleviate these concerns and accordingly prompt the widespread deployment of data outsourcing services, leading cloud service providers (e.g., Amazon, Google, and Microsoft) developed different security measures to prevent data exposure and illegal access. For example, they use AES-256 to encrypt user data at rest and leverage resource-based or user-based access policies to enforce data access control in use [Amazon 2015a; Google 2015; Amazon 2015b]. However, these approaches cannot support flexible and scalable data sharing, privacy-assured data search, and secure computation. Apart from removing the storage management on the user side, obtaining convenient data utilization services in the cloud is exactly what users want.

Aiming at further addressing these issues, novel cryptographic primitives and various security protection proposals for cloud data services have been presented recently. They can be broadly classified into four categories: confidentiality-assured cloud data service, owner-controlled cloud data sharing, integrity-guaranteed cloud data storage, and privacy-preserving cloud data access. More specifically, searchable encryption and homomorphic encryption techniques are proposed to enforce secure data search and data computation, respectively; selective encryption and attribute-based encryption techniques are introduced to achieve authorized access and secure data sharing; provable data possession and proof-of-retrievability techniques are presented to ensure data intactness and retrievability; and privacy preservation is enabled to protect multiple dimensions of private information (e.g., access pattern, query privacy, and identity information) when users access the data stored in the cloud. Note that we separate privacy preservation from data confidentiality protection though privacy protection probably can be achieved by certain data confidentiality techniques. Instead of pure data protection, we focus on specific privacy protection goals in privacy preservation.

In this article, we identify two broad categories of security threats to cloud data services, that is, threats from external attackers and threats from internal participants, and present four significant security requirements, that is, data confidentiality, data integrity, data access controllability, and privacy preservability. Following the requirements, we provide an overview of the data and privacy protection solutions in a high level, which will give readers an outline of protection technologies. In particular, we present recent research advances of confidentiality protection, access control, integrity guarantee, and privacy preservation for secure cloud data services. From this survey, a beginner or nonspecialist can easily follow this area to learn the problems. Moreover, we discuss some open challenges that need to be further explored, which provides future research directions for researchers in this area.

The rest of the survey is organized as follows. Section 2 presents a brief overview of security threats, security requirements, and the corresponding protection solutions for cloud data services. Sections 3 through 6 discuss different solutions of confidentiality-assured cloud data service, owner-controlled cloud data sharing, integrity-guaranteed cloud data storage, and privacy-preserving cloud data access, respectively. Section 7 presents a brief summary of existing solutions and some open issues that need future research efforts. Finally, the article is concluded in Section 8.

2. SECURE CLOUD DATA SERVICES: THREATS, REQUIREMENTS, AND SOLUTIONS

In this section, we first provide a system model of cloud data services. Then, we present several security threats, security requirements, and a high-level overview of the corresponding solutions for cloud data services.

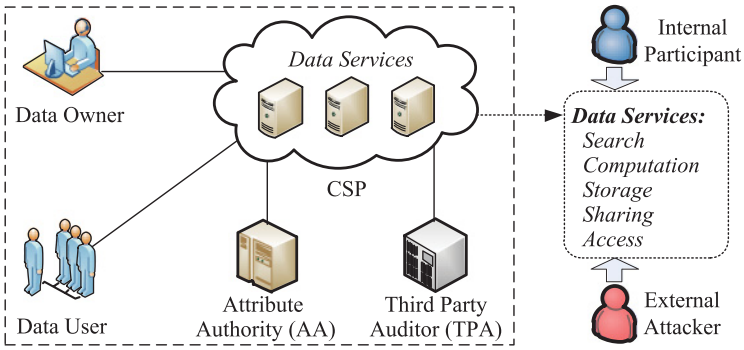

Fig. 1. System model of cloud data services.

2.1. System Model of Cloud Data Services

In general, typical entities of a cloud data service system include data owners (owners), data users (users), and a CSP, as shown in Figure 1. The owners upload their local data to a cloud and use its storage service. Beyond that, the owners expect that the cloud can offer more services such as data search, data computation, and data sharing. The user is the party that consumes the cloud data, for example, retrieving the specific data, getting the data computing results, and accessing the shared data. The CSP has significant storage space and computation capacities, and is responsible for providing all the aforementioned data services. Note that owners also act in the role of users when they utilize cloud data services.

A third party may be involved to provide security functionalities in the cloud, such as an Attribute Authority (AA) and Third Party Auditor (TPA). An AA is a trusted key authority in an attribute-based access control system [Li et al. 2012; Hur and Noh 2011; Yang et al. 2013a]. It takes charge of generating attribute keys for users according to their identity and updating or revoking users' attribute keys when their roles change. A TPA is a semitrusted party from the viewpoint of users in a public data verification scheme [Wang et al. 2013, 2015; Yuan and Yu 2013a]. It may be trusted to check the correctness of data stored in the cloud on behalf of users, but it has no privilege to access the actual content of data.

2.2. Threats to Data Security and User Privacy

There are a number of security threats associated with cloud data services, not only covering traditional security threats (e.g., network eavesdropping, illegal invasion, and denial-of-service attacks) but also including specific cloud computing threats (e.g., side-channel attacks, virtualization vulnerabilities, and abuse of cloud services). In this survey, we mainly consider data security and user privacy[2] under untrusted cloud environments, and survey user-controlled security schemes. Accordingly, we focus on the threats to data contents, data intactness, and user privacy in a cloud service system.

Normally, from the viewpoint of cloud users, two types of adversaries are considered to pose security threats to cloud data services. One is external attackers (e.g., hackers), while the other is internal participants (e.g., CSP, TPA). The threats from these adversaries may raise various security issues, such as data leakage or disclosure,

[2]Privacy refers to sensitive information about users that is expected to be secluded from others. It is a closely related concept with security. It involves more aspects than confidentiality (e.g., financial privacy, medical privacy, political privacy, and data privacy). In our article, we focus on user privacy, which covers identity information, query privacy, and access pattern of cloud users.

unauthorized access, data corruption or loss, and user privacy breach [CSA 2013; Xiao and Xiao 2013].

2.2.1. Threats from External Attackers. In general, external attackers are *malicious*. They utilize a variety of attack techniques (e.g., network eavesdropping, vulnerability scanning, and malware attacking) to attempt to get access privileges and unauthorized access to the data outsourced to cloud. Attackers may intentionally tamper with or delete the data outsourced to the cloud for their unfavored motivations, which will make the data incorrect or unavailable. Moreover, they may maliciously infer the users' private behaviors out of curiosity or for other sordid purposes.

2.2.2. Threats from Internal Participants. In a cloud service system, the CSP and TPA are not fully trusted internal participants. The CSP is considered as an *honest* but *curious*, *vulnerable*, or *greedy* provider, while the TPA is often regarded as an *honest* but *curious* party.

Honest but curious is a popular cloud threat model in most existing schemes [Samarati 2014; Li et al. 2014; Sun et al. 2013; Yuan and Yu 2013a]. In other words, the CSP can honestly execute the system protocols and faithfully provide data services, but it is curious about the outsourced data and user access. It desires to learn the information of data stored or processed in the cloud. Furthermore, in order to increase commercial interests, the CSP may attempt to infer the users' personal information (e.g., identity, preferences, and habits) according to the users' data queries or data accesses. These undesirable inferences clearly compromise the privacy of cloud users.

The CSP is also regarded as a *vulnerable* provider in some proposed schemes [Bowers et al. 2011; Wang et al. 2013], which means the data stored in the cloud is possibly corrupted or erased as a result of hardware or software failures (e.g., configuration errors and misoperation of server management, service software bugs, and improper software updates), as well as physical storage hardware failures. Additionally, it is *vulnerable* to facing natural or man-made disasters like earthquakes, fires, and power failures, which may cause data loss in a cloud storage system as well. Sometimes, the CSP is *greedy* for reducing its costs [Wang et al. 2010, 2011]. For instance, to save storage costs, it is likely to discard rarely accessed data by cloud users, which will incur data loss [Yang and Jia 2012].

Honest but curious is also considered as a threat model for the TPA in the proposed audit schemes [Yang and Jia 2012]. The TPA can loyally perform the audit protocol during the whole checking procedure, but it is curious about the audited data and eager to know the actual data content.

2.3. Security Requirements

To throttle the threats from malicious attackers, curious CSPs/TPAs, and vulnerable or greedy CSPs, the following security requirements are to be met in a cloud data service.

2.3.1. Data Confidentiality. Data confidentiality is the property by which data contents are not made available or disclosed to unauthorized users. Outsourced data is stored in a cloud and out of the owners' direct control. Only authorized users can access the sensitive data, while others, including CSPs, should not gain any information about the data. Meanwhile, data owners expect to fully utilize cloud data services (e.g., data search, data computation, and data sharing) without leakage of the data contents to CSPs or other adversaries.

2.3.2. Data Access Controllability. Access controllability means that a data owner can perform the selective restriction of access to his or her data outsourced to the cloud. Some users can be authorized by the owner to access the data, while others cannot access it without permission. Further, it is desirable to enforce fine-grained access

Table II. Security Threats, Requirements, and Solutions for Cloud Data Services

Threats		Requirements		Solutions
Malicious attackers Curious CSP Vulnerable or greedy CSP Curious TPA	Data leakage or disclosure	Data confidentiality	Secure data search	Searchable Encryption (SE) [Wang et al. 2010]
			Secure data computation	Homomorphic Encryption (HE) [Gentry 2009b]
	Illegal access	Data access controllability	Secure data sharing	Selective Encryption [De Capitani di Vimercati et al. 2010]
				Attribute-Based Encryption (ABE) [Sahai and Waters 2005]
	Data corruption or loss	Data integrity	Secure data storage	Provable Data Possession (PDP) [Ateniese et al. 2007]
				Proof of Retrievability (POR) [Juels and Kaliski 2007]
	User privacy breach	Privacy preservability	Privacy preservation data access	Access Pattern Protection [Chor et al. 1998; Ding et al. 2011; Yang et al. 2011]
				Query Privacy Protection [Sun et al. 2013; Cao et al. 2014]
				Identity Privacy Protection [Wang et al. 2012a, 2012b; Nabeel et al. 2011]

control to the outsourced data; that is, different users should be granted different access privileges with regard to different data pieces. The access authorization must be controlled only by the owner in untrusted cloud environments.

2.3.3. Data Integrity. Data integrity demands maintaining and ensuring the accuracy and completeness of data. A data owner always expects that his or her data in a cloud can be stored correctly and trustworthily. This means that the data should not be tampered with, fabricated, or maliciously deleted. If any undesirable operations corrupt or delete the data, the owner should be able to detect the corruption or loss. Further, when a portion of the outsourced data is corrupted or lost, the rest of the data should be retrievable.

2.3.4. Privacy Preservability. Many users pay more attention to their privacy protection when they access cloud data or use cloud services. In particular, they expect to hide their identity while using cloud data services. Some users also want their operations on the data and the information retrieved from a cloud to be properly protected. For instance, the keywords queried over the outsourced data and the query results returned by a cloud should not be exposed to others. Moreover, it is expected that users' access behaviors and habits should not be inferred by any other parties in cloud.

2.4. Overview of Security Solutions

According to the security requirements for cloud data services, the research community has developed different data and privacy protection solutions, shown in Table II.

2.4.1. Confidentiality-Assured Cloud Data Service. In public cloud environments, it is necessary to keep data secret while providing normal data utilization services. Encryption is a basic mechanism to enable data confidentiality. However, traditional encryption hampers the data utilization services (e.g., data search and data computation). Intuitively, all the outsourced data may be downloaded and decrypted locally to perform data search or data computation. However, this naive solution is obviously impractical for the reason of prohibitive communication overheads. To achieve confidentiality-assured and effective data services in the cloud, new cryptographic primitives and data encryption proposals have been presented. Specifically, Searchable Encryption (SE) [Wang

et al. 2010] and Homomorphic Encryption (HE) [Gentry 2009b] are proposed to offer secure search and computation services.

2.4.2. Owner-Controlled Cloud Data Sharing. In untrusted cloud environments, a challenging problem is to enable fine-grained enforcements of data access and achieve secure data sharing among large-scale users. Obviously, since an owner does not trust the cloud, traditional access control mechanisms, normally depending on a trusted server, are not suitable for cloud data sharing. To address this challenge and enforce owner-controlled access control, data should be encrypted by the owner before outsourcing it to the cloud, and then the owner can perform fine-grained access control over the encrypted data by securely distributing keys. Based on this idea, access control based on encryption is proposed. Two typical mechanisms in this direction are access control based on selective encryption [De Capitani di Vimercati et al. 2010] and Attribute-Based Encryption (ABE) [Sahai and Waters 2005], respectively.

2.4.3. Integrity-Guaranteed Cloud Data Storage. Since outsourcing data to the cloud implies that owners no longer locally possess their data, owners are concerned whether the outsourced data is correctly stored in the cloud. Traditional approaches of data correctness guarantees cannot be adopted directly, because a local data copy is generally required to verify the data integrity. Furthermore, the limited computing resources of the owners and the vast amount of cloud data make the task of checking data intactness expensive and even unbearable. Aiming at efficiently verifying data integrity in a cloud, blockless verification and probabilistic check have been adopted. Blockless verification allows a user to verify data authenticity without accessing the actual data, while probabilistic check enables a user to verify data integrity by launching a random challenge-response protocol. Provable Data Possession (PDP) [Ateniese et al. 2007] and Proof of Retrievability (POR) [Juels and Kaliski 2007] are two popular integrity verification approaches enabling blockless verification and probabilistic check. Compared with PDP, which only checks whether outsourced data is intact, POR enables a user to retrieve all of the data from cloud servers with high probability, even though a part of the outsourced data is corrupted or lost.

2.4.4. Privacy-Preserving Cloud Data Access. User privacy in cloud services includes identity privacy, query privacy, and access pattern privacy. Different privacy issues can be addressed with different protection techniques and need some specific considerations. To achieve access pattern privacy, three typical protection approaches are proposed: Private Information Retrieval (PIR) [Chor et al. 1998], Oblivious RAM (ORAM) [Ding et al. 2011], and dynamically allocated data structure [Yang et al. 2011]. Query privacy can be effectively implemented by using secure indexes [De Capitani di Vimercati et al. 2011], virtual/dummy keywords [Sun et al. 2013], or random trapdoors [Cao et al. 2014]. When data is shared within a group, ring or group signature techniques are used to keep user identity information secret [Wang et al. 2012a, 2012b]. Anonymous access techniques are adopted to protect user identity in order to avoid identity leakage when data access control is enforced [Nabeel et al. 2011].

3. CONFIDENTIALITY-ASSURED CLOUD DATA SERVICE

Encryption is the simplest and most common way to provide data confidentiality before data is outsourced to a cloud. However, traditional encryption techniques make deployment of data services, such as data search services and data computation services, a difficult task. In addition to relieving the burdensome work of data management, outsourcing data to a cloud serves no purpose from the perspective of users if these data services cannot be conveniently utilized. Recently, new encryption primitives (i.e., searchable encryption [Boneh et al. 2004; Wang et al. 2010] and homomorphic

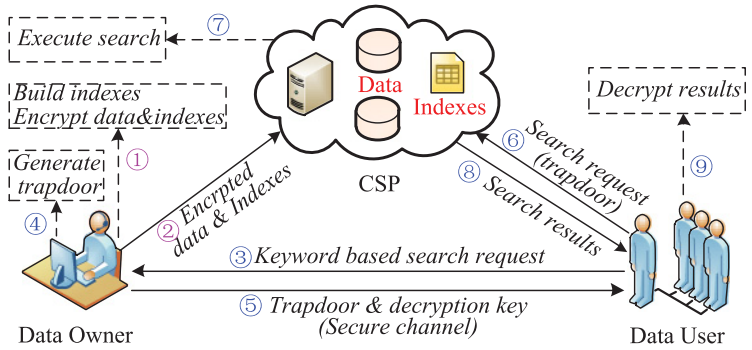

Fig. 2. Architecture of search over encrypted cloud data.

encryption [Gentry 2009b; Smart and Vercauteren 2010]) have been proposed to enable secure data search and data computation.

3.1. Encryption for Secure Data Search

To conveniently access the related data stored in the public cloud, enabling a secure data search service for users is of paramount importance. SE is a helpful technique to achieve this goal. It is a cryptographic primitive that offers secure search functions over encrypted data. In order to improve search efficiency, an SE solution generally builds keyword indexes to securely perform user queries. Existing SE schemes can be classified into two categories: SE based on symmetric-key cryptography [Kamara et al. 2012; Cao et al. 2014; Wang et al. 2010; Sun et al. 2013] and SE based on public-key cryptography [Boneh et al. 2004; Sun et al. 2014; Yanguo et al. 2014]. The former is efficient in performing data encryption and data search. A main disadvantage of this scheme is that keys must be transmitted over a secure channel, and it only offers limited search functionality. In contrast, the latter is able to generate more flexible and expressive queries. Nevertheless, it is inefficient since complex computations of public-key cryptography are involved.

3.1.1. Search Design. A general SE design is illustrated in Figure 2 [Li et al. 2013; Cao et al. 2014; Li et al. 2014; Wang et al. 2014]. To provide search capabilities on ciphertexts, search indexes are first built, which maintain the information of a set of keywords contained in original data, and both original data and the indexes are encrypted (step 1) and outsourced to the cloud together (step 2). When a user intends to perform a search on outsourced data, he or she needs to send the search keywords to the owner (step 3). The owner generates a trapdoor based on the keywords using the related secret key (step 4) and returns the trapdoor and the key to the user over a secure channel (step 5). Then, the user may submit a search request with the trapdoor to the CSP (step 6). The CSP performs a matching search process based on the trapdoor (step 7) and transmits the research results back to the user (step 8). Finally, the user obtains the plaintexts by decrypting the received results with the secret key (step 9). In this search process, there is no cleartext information exposure to the CSP, since all operations performed by the CSP are executed on encrypted data only by using the secure trapdoor.

3.1.2. Search Index. There are two basic types of index structures that can be used to construct a search index of SE schemes:

—Index organized by keywords for all documents [Curtmola et al. 2011; Wang et al. 2012; Kamara et al. 2012]: The index constructed upon keywords is called an inverted

index. In such a data structure, each keyword is followed by a file list that consists of all files containing the keywords. The structure enables an extremely fast search process. However, updating the index is a problem. When a new file is appended, it needs to rebuild the structure and update all indexes containing the keywords included in this file. Another issue with this keyword-based index structure is that searches can only be performed with a single keyword query.

—Index constructed per document [Li et al. 2014; Cao et al. 2014; Wang et al. 2014; Örencik and Savaş 2014]: This structure adopted by many ciphertext search schemes constructs a specific index, which consists of all of the keywords in one target document. The advantage of this index structure is that one file update usually affects only one corresponding index. However, when a keyword query is executed, a search algorithm needs to traverse full indexes of all files.

Keyword indexes can be built based on tree structures [Lu 2012; Sun et al. 2013]. This kind of index structure offers a more efficient search than the one organized by the list.

3.1.3. Search Functionality. Recently, many efforts have focused on improving query expression of SE and enriching its search functions. In addition to single-keyword search [Boneh et al. 2004] and multikeyword search [Li et al. 2014; Wang et al. 2014; Örencik and Savaş 2014], most of the existing SE schemes can fall into fuzzy-keyword search [Li et al. 2010; Wang et al. 2012; Kuzu et al. 2012; Wang et al. 2014], search on dynamic data [Kamara et al. 2012; Naveed et al. 2014; Cash et al. 2014], ranked search [Li et al. 2014; Sun et al. 2013; Cao et al. 2014; Örencik and Savaş 2014], authorized search [Zheng et al. 2014; Sun et al. 2014], and verifiable search [Zheng et al. 2014].

Fuzzy-Keyword Search. The fuzzy-keyword search schemes adopt fuzzy-keyword search techniques for tolerating spelling mistakes. A similarity measurement is introduced to specify some predefined tolerance range. For this purpose, Li et al. [2010] adopt the metric of edit distance to measure keyword similarity and exploit a wildcard-based technique to construct similarity keyword sets. The proposed search scheme can return a match result if the intended keyword is within the constructed similarity keyword sets. Wang et al. [2012] solve a more general problem of similarity search. They develop a suppressing method to construct fuzzy-keyword sets with efficient storage and then build a private trie-traverse searching index. These techniques can achieve the fuzzy search results with a constant time complexity. In order to further improve the efficiency of fuzzy-keyword search, Kuzu et al. [2012] also propose a similarity keyword search scheme. This scheme builds a secure index based on locality-sensitive hashing (LSH) and BF, and utilizes an approximation near-neighbor search algorithm to search the index in high-dimensional spaces. To tackle the challenges of multikeyword fuzzy search on encrypted data, Wang et al. [2014] leverage LSH functions in the Bloom filter to construct the file index, and adopt the Euclidean distance to capture the similarity between the keywords. This scheme efficiently implements secure fuzzy search with multiple keywords.

Search on Dynamic Data. Users may append new data or modify or remove the uploaded data in a cloud. Search on dynamic data in SE is capable of executing search on data that are frequently updated or deleted. Kamara et al. [2012] present a construction of dynamic searchable symmetric encryption based on an improved inverted index structure. The scheme can effectively support file-updating search and defend against chosen-keyword attacks. Naveed et al. [2014] introduce a new cryptographic primitive, named Blind Storage. It allows a user to store a set of files on a cloud, while the cloud does not need to know how many files are stored and what the lengths of each file

are. By using Blind Storage, they propose a simpler and more efficient SE scheme that offers dynamic file operations. To achieve dynamic search, Cash et al. [2014] adopt an optimized index structure in a very large database. It effectively supports additions to the data, as well as deletions via revocation lists. MKQE [Li et al. 2014] introduces partitioned matrices. A keyword dictionary in MKQE can be expanded dynamically without operating the contents in the original dictionary. Hence, the overhead of the dictionary reconstruction and the file index re-encryption is greatly reduced.

Ranked Search. Relevance ranking search returns the most relevant set of files, rather than directly returning undifferentiated search results to users. A match metric is used to realize the ranking functionality of an SE scheme. Some common match metrics are as follows: (1) keyword frequency or term frequency (TF), (2) inverse document frequency (IDF), (3) coordinate matching, and (4) cosine measure. Wang et al. [2010] resort to order-preserving symmetric encryption to construct a secure ranked search scheme. The construction uses the keyword frequency metric to provide the ranked search functionality. MRSE [Cao et al. 2014] focuses on addressing the multikeyword ranked search over ciphertexts by employing the similarity measure of coordinate matching. The schemes use secure inner product computation to solve the challenges of identifying multikeyword semantics without privacy leakage and adopt the metric TF$\times$IDF to enhance ranked search accuracy. For achieving more accurate multikeyword ranked search results, Sun et al. [2013] adopt a cosine measure in the vector space model, and further incorporate the metric TF$\times$IDF. Örencik and Savaş [2014] assign relevancy levels based on the weights of search terms. The proposed search scheme can return the top matching results to users. Similarly, MRQE [Li et al. 2014] takes keyword weights and user access history as the metric. In this scheme, files get higher ranks in the matching result if the files have higher access frequencies and match closer to the users' access history. Thus, the users have a high probability of obtaining the files that they really want according to the file ranks.

Authorized Search and Verifiable Search. Zheng et al. [2014] propose a verifiable attribute-based keyword search scheme to check whether the search operations are indeed executed. The scheme allows a data owner to control the search of its outsourced data according to an access control policy. A cloud user can search on the outsourced data only when he or she possesses the credential that satisfies the owner's access control policy. In the scheme, an authorized user is also able to check the cheating search behavior performed by a cloud. Sun et al. [2014] present a scalable authorized keyword search scheme in the presence of multiple data owners by exploiting the ciphertext policy ABE technique. It offers fine-grained owner-enforced search authorization at the file level.

Table III gives a detailed comparison of search functionalities, security goals, security models, and efficiency (index storage, trapdoor size, and search time) of the different searchable encryption schemes discussed earlier. The legends and notations can be found in Table IV.

3.2. Encryption for Secure Data Computation

In many application scenarios, the cloud users expect to get a secure data computing service provided by powerful cloud servers. HE can meet this requirement. It allows computation operations to be directly performed on ciphertexts by mirroring the corresponding operations on the plaintexts; that is, the decrypted value of computation results on ciphertexts is identical to the value of operations on the plaintexts. Early HE schemes only offer homomorphic addition, multiplication, or a limited combination of the two operations (e.g., computation of quadratic formulas [Boneh et al. 2005]). For more complicated homomorphic operations, Gentry [2009a, 2009b] proposes and constructs the first Fully Homomorphic Encryption (FHE) solution. The scheme

Table III. Comparison of Different Searchable Encryption Schemes

Scheme	Search Func.	Security Goal	Efficiency			Security Model
			Index Storage	Trapdoor Size	Search Time	
Li et al. [2010]	Single, Fuzzy	File, Index	$O(l^d m)$	$O(l^d)$	$O(l^d m)$	IND-CKA1
Wang et al. [2012]	Single, Fuzzy	File, Index, Keyword length	$O(\tau m)$	$O(\tau)$	$O(\tau)$	IND-CKA1
Kuzu et al. [2012]	Single, Fuzzy	File, Index, Similarity pattern	$O(\lambda m)$	$O(\lambda)$	$O(\lambda)+O(t)$	IND-CKA2
Kamara et al. [2012]	Single, Dynamic	File, Index	$O(\sum_w n_w+m)$	$O(1)$	$O(n_w)$	IND-CKA2
Naveed et al. [2014]	Single, Dynamic	File, Index, Access pattern	$O[\alpha(\sum_w n_w+m)]$	$O(1)$	$O(t_{Br} n_w)+O(t_{Bu})+O(t_{Cu})$	IND-CKA2
Cash et al. [2014]	Single, Dynamic	File, Index	$O(\sum_w n_w)$	$O(1)$	$O[(n_w+u_w)/p]$	IND-CKA2
Wang et al. [2014]	Multi, Fuzzy, Dynamic	File, Index, Trap-unlink	$O(n)$	$O(1)$	$O(n)$	IND-CKA1
Örencik and Savaş [2014]	Multi, Ranked	File, Index, Trap-unlink, Search pattern	$O[(q+2)m]$	$O(r)$	$O[(q+2)n\eta]$	IND-CKA2
Sun et al. [2014]	Multi, Authorized	File, Index, Trap-unlink	$O[(N_a+2)m]$	$O(2N_a+2)$	$O[(2N_a+4)m]$	IND-CKA2
Li et al. [2014]	Multi, Ranked, Dynamic	File, Index, Trap-unlink	$O[2(m+m_d+1)n]$	$O[2(m+m_d+1)]$	$O[(m+m_d+1)nk]$	KBM
MRSE-I [Örencik and Savaş 2014]	Multi, Ranked	File, Index, Trap-unlink	$O[2(m+2)n]$	$O[2(m+2)]$	$O[(m+2)nk]$	KCM
MRSE-II [Örencik and Savaş 2014]			$O[2(m+m_d+1)n]$	$O[2(m+m_d+1)]$	$O[(m+m_d+1)nk]$	KBM
BMTS [Sun et al. 2013]	Multi, Ranked	File, Index, Trap-unlink	$O(2mn)$	$O(2m)$	$O(mnk)$	KCM
EMTS [Sun et al. 2013]			$O[2(m+m_d)n]$	$O[2(m+m_d)]$	$O[(m+m_d)nk]$	KBM
KP-ABKS [Zheng et al. 2014]	Single, Authorized, Verifiable	File, Index, Keywords	$O[(N_a+3)m]$	$O(2N_a+2)$	$O[(3N_a+2)m]$	KCM
CP-ABKS [Zheng et al. 2014]			$O[(2N_a+3)m]$	$O(2N_a+3)$	$O[(3N_a+3)m]$	KBM

Column "Search Func." means search functionalities achieved by the proposed schemes.

theoretically supports any operation on ciphertexts, but it cannot be applied due to high computational overhead. Therefore, FHE constructions and optimizations for its practicability have become hot research topics in recent years.

3.2.1. Construction of FHE. The FHE scheme can be divided into pure FHE and leveled FHE. A pure FHE scheme can ideally support calculations of any depth circuit, namely, infinite operations on ciphertexts. Instead, a leveled FHE scheme only offers evaluations of any given polynomial size circuit, that is, limited calculations. According to the mathematical basis of FHE construction, there are the following three types of schemes [Cheon et al. 2013]: schemes based on ideal lattices, schemes over the integers, and schemes based on Learning with Errors (LWE) or Ring Learning with Errors (RLWE).

The ideal lattice-based schemes [Gentry 2009b; Smart and Vercauteren 2010; Gentry and Halevi 2011] make use of hard problems on ideal lattices, while the schemes over the integers [Van Dijk et al. 2010; Cheon et al. 2013] depend on the hardness of the approximate-gcd problem. All of them generally follow the construction framework presented in Gentry [2009a]. The framework consists of several steps [Gentry and Halevi 2011]. First, a somewhat homomorphic encryption (SwHE) scheme is

Table IV. Legends and Notations for Table III

Legends for Search Functionalities		Legends for Security Goals/Models	
Single	single-keyword search	Trap-unlink	trapdoor unlinkability
Multi	multikeyword search	IND-CKA1	nonadaptive indistinguishability
Fuzzy	fuzzy-keyword search	IND-CKA2	adaptive indistinguishability
Dynamic	search on dynamic data	KCM	known ciphertext model
Ranked	ranked search	KBM	known background model
Authorized	authorized search		
Verifiable	verifiable search		

Notations for Efficiency			
n	number of files	m_d	number of dummy keywords inserted into a file
m	number of keywords	N_a	number of universal attributes
k	number of top-k related search results	η	number of ranking levels
l	length of a keyword in characters	τ	maximum size of the similarity keyword set
d	edit distance	λ	number of LSH functions
p	number of search processors	α	constant factor of Blind Storage
q	factor of added fake index entries	t_{Br}	read time of Blind Storage
r	bit number of indexes or trapdoors	t_{Bu}	update time of Blind Storage
t	number of top related research results	t_{Cu}	update time of Clear Storage
n_w	number of files that contain keyword w	u_w	number of times the searched-for keyword has been added/deleted

constructed to evaluate low-order polynomials. Then, the decryption process associated with an arbitrary ciphertext is squashed; as a result, it can be described as a low-order polynomial in the secret key bits. Finally, a bootstrapping transformation is applied to produce a fully homomorphic scheme by repeatedly refreshing ciphertexts. There are two paramount points here. One is to design a scheme that can support the evaluation of high-enough-order polynomials. The other is to obtain a squashed decryption process that can be described as low-enough-order polynomials. When the order of the decryption polynomial is less than the order of polynomials that the designed scheme can evaluate, the scheme can be transformed into a fully homomorphic scheme through recursive self-embedding.

The construction process of the LWE- and RLWE-based FHE schemes [Brakerski et al. 2012; Brakerski and Vaikuntanathan 2014] is similar. The first step is to construct an SwHE scheme from LWE, and the second step is to use a key-switching technique for dimension reduction that controls the dimension expansion of the new ciphertext vector and to adopt a modulus-switching technique for noise management that keeps noise in check. To achieve full homomorphic encryption, the schemes finally iterate the aforementioned operations of dimension reduction and noise management. Since they do not use bootstrapping, the schemes constructed by this method are more efficient than the ones built by Gentry's framework [Gentry 2009a]. Nevertheless, these construction schemes can only achieve leveled FHE. Besides, the adopted key-switching technique affects the efficiency of operations because each operation of key switching needs to multiply a matrix. Therefore, the efficiency in key-switching operations is the crucial issue for this FHE construction.

Recently, Gentry et al. [2013] proposed an approximate eigenvector method to eliminate dimension reduction and noise control. In the designed LWE-based FHE scheme, homomorphic addition is just matrix addition, and homomorphic multiplication directly corresponds to matrix multiplication. Hence, the scheme constructed by this method is more efficient than the prior.

3.2.2. Optimization and Implementation of FHE. The efficiency of FHE schemes can be improved by controlling ciphertext size, optimizing key generation, or reducing calculation dimensions. According to the elementary theory of algebraic number fields, Smart and Vercauteren [2010] present an FHE scheme that has relatively small ciphertext and key size. Their FHE construction is still produced from an SwHE scheme. In comparison with Gentry's original scheme [Gentry 2009b], this construction has smaller message expansion and key size. To optimize the key generation in Gentry's scheme [Gentry 2009b], Gentry and Halevi [2011] design a new faster algorithm for computing secret keys. They also present many simplifications and optimizations to improve the FHE efficiency, including a batching technique for encryption and a tradeoff between space and time overhead. Moreover, a ring-switching operation is used to lower ring dimensions in order to speed up further homomorphic computation in FHE over polynomial rings [Gentry et al. 2012c].

SwHE schemes are building blocks for FHE schemes; hence, it is crucial to optimally realize an SwHE scheme. Naehrig et al. [2011] implement the SwHE scheme based on the RLWE assumption [Brakerski and Vaikuntanathan 2011, 2014] in the computer algebra system MAGMA and further optimize the implementation by adopting the specific message-encoding techniques. In the FHE system, homomorphic evaluation of arithmetic circuits can be accomplished with only polylogarithmic overhead [Gentry et al. 2012b]. To reduce the overhead, the scheme uses a batch homomorphic evaluation technique to pack ciphertexts. Moreover, it introduces permutating/routing techniques to move data elements in plaintext slots arbitrarily. Therefore, the scheme implements a general arithmetic circuit in a batched mode without ever unpacking plaintext vectors.

The Single Instruction Multiple Data (SIMD) parallel processing technique can be used to improve the efficiency of FHE systems. Gentry et al. [2012a] and Smart and Vercauteren [2014] attempt to enable the SIMD operations for speeding up the homomorphic computation. To achieve efficient bootstrapping, Gentry et al. [2012a] perform homomorphic operations in an SIMD fashion. Smart and Vercauteren [2014] show how to select parameter settings to obtain the advantage of SIMD operations and get a SwHE scheme supporting SIMD operations. Their SwHE system can perform re-encryption operations in parallel, thus resulting in substantial speedup.

4. OWNER-CONTROLLED CLOUD DATA SHARING

Data sharing is another fundamental data service in the cloud. This service should support secure and efficient access to the shared data among a large number of users with differentiated access privileges. Traditionally, the primary method of enforcing such selective data access is by utilizing a trusted storage server to enforce access policies and manage data users. Nevertheless, the traditional access control mechanism depending on the trusted server is no longer applicable in the open cloud environment, because cloud servers are not in the trusted domain of data owners. To address this issue and achieve owner-controlled cloud data sharing, access control based on selective encryption and ABE are proposed [De Capitani di Vimercati et al. 2014; De Capitani di Vimercati et al. 2013; Yu et al. 2010; Li et al. 2013].

4.1. Selective Encryption-Based Access Control

Access control based on selective encryption is a novel approach that uses cryptography for authorizations. It allows selective access to encrypted outsourced data by effective key management. Different keys can be distributed to different users who have visibility on the corresponding resources. The crucial issues in these schemes are how to successfully translate an authorization policy into an equivalent encryption policy and how to efficiently manage keys used to encrypt different data.

4.1.1. Privilege Control. There are two basic access privileges on outsourced data, that is, read privilege and write privilege. To enforce read privilege, De Capitani di Vimercati et al. [2010] present an authorization policy model by using an equivalent encryption policy based on graph theory. They present a proof that obtaining an optimal encryption policy is an NP-hard problem and further introduce a heuristic algorithm to solve it. They also adopt a key derivation algorithm based on symmetric cryptography for efficient key management. Aiming at the scenario where the data is controlled by multiple owners, De Capitani di Vimercati et al. [2010] exploit two cryptographic techniques to control data access. One is a key agreement algorithm that is based on the Diffie-Hellman key agreement. The other is a key derivation algorithm that enables a key to be derived from another key and a public token. The combination of these two algorithms is able to correctly convert access policies defined by data owners into encryption policies.

Enforcing write access control is more challenging. Raykova et al. [2012] present an access control scheme that supports both read and write privileges. The scheme proposes a combination of a coarse-grained access control and a fine-grained access control. The former is enforced by the data owners, while the latter is enforced by the CSP. To differentiate read and write privileges, a public-private key pair for each document is provided at the fine-grained level. Further, two token trees are built to distribute the private and public keys, respectively used to enforce read and write privileges. De Capitani di Vimercati et al. [2012b] present another approach to allowing the write access to encrypted cloud data. The basic idea of this approach is to associate each resource with a write tag. The cloud server allows a user to perform a write operation on a file if he or she can correctly show the corresponding write tag. The crucial point of this scheme is that the keys used to encrypt write tags have to be shared between authorized users and the server. To achieve efficient management of the shared keys, a new key derivation structure is presented based on the set-based key derivation graph. As an extension to their previous scheme, De Capitani di Vimercati et al. [2013] present a proposal that enables write privilege updates (i.e., grant and revoke write operations). In this proposal, these operations can be implemented by inserting tokens and keys into the key derivation structure. A main advantage of the proposal is that updates on write access control policies are delegated to cloud servers, thus relieving the burden of the data owner.

4.1.2. Policy Update and Management. To enable an encryption policy update, the following operations can be applied: (1) user insertion or deletion, (2) resource insertion or deletion, and (3) permission grant or revocation. Inserting or deleting a user means that the involved user is granted or revoked all the access permissions, respectively. Similarly, inserting or deleting a resource means that the involved resource is granted or revoked all the authorizations, respectively. It can be seen that the first two policy updates are in essence the same as the third one. Based on this result, De Capitani di Vimercati et al. [2010] concentrate on the permission grant and revocation operations. They translate these authorization policy operations into proper update operations on an encryption policy graph. Moreover, policy change can be implemented by only updating parts of the graph associated with the grant or revocation operations.

If a data owner enforces the Access Control Policy (ACP) by him- or herself, high communication and computation costs will be incurred for data re-encrypting and re-uploading when user credentials are changed. To address this issue, ACP enforcement delegation to a cloud could be an efficient approach. De Capitani di Vimercati et al. [2010] propose a two-layer encryption approach to enabling ACP enforcement delegation. With this approach, a cloud can directly enforce policy changes requested by the data owners, which avoids resource transfer and re-encryption. A similar approach

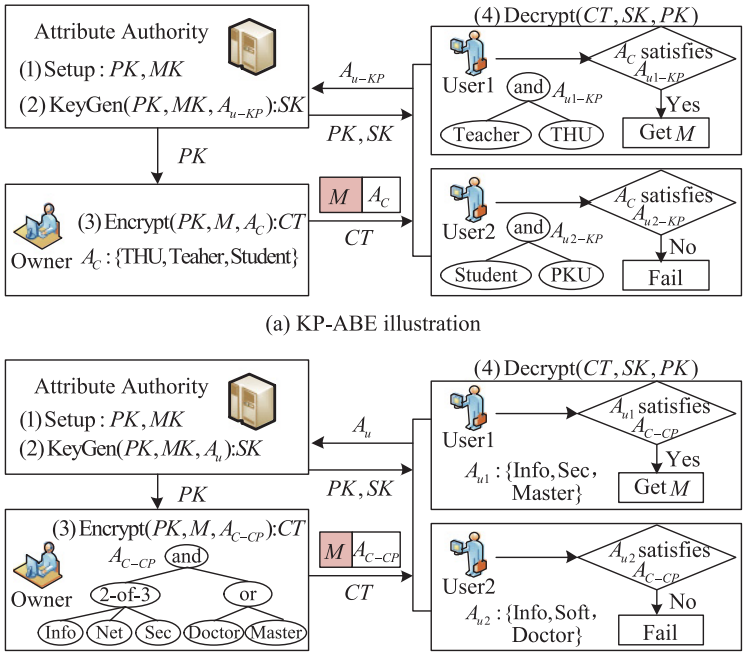

Fig. 3. KP-ABE and CP-ABE illustration.

is presented in De Capitani di Vimercati et al. [2010], which supports delegation of ACP enforcement to a semitrusted cloud so that it offers an efficient policy update performed by the cloud. Nabeel and Bertino [2014] and Raykova et al. [2012] borrow the idea of two-layer encryption and construct delegated access control schemes. In their schemes, each ACP is decomposed into two sub-ACPs, which are enforced by the two-layer encryption; that is, data owners first encrypt the data based on one sub-ACP and the cloud re-encrypts the encrypted data using the other sub-ACP.

4.2. ABE-Based Access Control

ABE [Sahai and Waters 2005] is a cryptographic technique that can offer flexible expressions of user access policies and support fine-grained access control on outsourced data in a cloud. ABE can be divided into Key-Policy Attribute-Based Encryption (KP-ABE) [Goyal et al. 2006] and Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [Bethencourt et al. 2007].

Normally, an ABE scheme has four core algorithms: *Setup*, *KeyGen*, *Encrypt*, and *Decrypt*, as shown in Figure 3. First, the AA runs *Setup* to produce public parameters PK and a master key MK (step 1), and then generates a private key SK for a user by operating *KeyGen* (step 2). Next, an owner encrypts his or her data M with the algorithm *Encrypt* and outputs a ciphertext CT (step 3). Finally, a user uses *Decrypt* to decrypt the ciphertext CT with his or her private key SK and attempts to get the plaintext M (step 4). In the process of running these algorithms, there are differences between KP-ABE and CP-ABE. In a KP-ABE scheme (see Figure 3(a)), the private key SK of a user is created with a specific access structure A_{u-KP}. The ciphertext CT is produced based on a set of attributes A_c. The ciphertext CT can be decrypted by a user as long as the attributes A_c embedded in the CT meet the access structure

A_{u-KP} specified in his or her private key SK. On the contrary, in a CP-ABE scheme (see Figure 3(b)), the attributes A_u of a user are used to compute his or her private key, and an access structure A_{c-CP} is attached to the ciphertext CT. Analogously, as long as the attributes A_u in the user's private key meet the access structure A_{c-CP} implied in the CT, he or she can achieve successful decryption.

4.2.1. Access Structure. The power of access control policy expression mainly depends on the access structures of ABE. There are two typical access structures: tree-based structure [Goyal et al. 2006; Bethencourt et al. 2007; Ostrovsky et al. 2007; Goyal et al. 2008; Ibraimi et al. 2009] and LSSS-matrix-based structure [Waters 2011; Green et al. 2011; Lewko et al. 2010].

Goyal et al. [2006] present a tree-based access structure in their KP-ABE scheme. The tree-based structure is monotonic. In the tree, each leaf is labeled with an attribute, and all the interior nodes are threshold gates. This design supports "AND" and "OR" gates. Ostrovsky et al. [2007] further enrich the expressibility of an access tree by adding the Boolean operation of "NOT." Similarly, the CP-ABE systems [Bethencourt et al. 2007; Ibraimi et al. 2009] also build a monotonic access tree with enabling "AND," "OR," and general threshold operations. By transforming a big access tree into smaller access subtrees, they further enforce more expressive access policies, such as access control based on numeric ranges. Goyal et al. [2008] further propose a bounded access tree structure, which allows a data owner to choose a threshold value, the depth of the access tree, and the cardinality of each node in the tree for data encryption. The access structure can express nonmonotonic access policies and support any access formula with a bounded-width polynomial size, including nonmonotonic access policies.

Unlike the aforementioned access structures, access policies in a CP-ABE system [Waters 2011] are expressed by an LSSS matrix [Beimel 1996] over attributes. This technique enables a more succinct representation of access policies than tree-based structures, while not losing operation efficiency. In the constructed system, both the ciphertext size and the time of encrypting/decrypting operations are in proportion to the complexity of the access expressions. Lewko et al. [2010] and Green et al. [2011] leverage the LSSS matrix to present an access structure as well and construct several secure ABE systems, respectively.

4.2.2. User/Attribute Revocation. Generally, there are two levels of revocation in an ABE-based access control system: user revocation and attribute revocation. User revocation means all of the attributes owned by revoked users are invalid in the system, while attribute revocation revokes access of all users who hold the attribute. Either data owners or attribute authorities can perform the revocation operations.

In Yu's scheme [Yu et al. 2010], user revocation is efficiently achieved based on the proxy re-encryption (PRE) [Blaze et al. 1998] technique. When the revocation of a user is required, the data owner specifies the minimal set of attributes that can disable the access permissions of the user. Then, he or she regenerates a master key and a public key for the involved attributes and computes the corresponding PRE keys. After that, the data owner sends the PRE keys and a revocation message to the cloud server. In the end, the revoked user is removed from the system user list and secret key elements of the remaining users are updated, and the cloud data are also updated by re-encrypting them with the new PRE keys.

To offer automatic attribute revocation, a time-based approach is presented in Bethencourt et al. [2007]. In this approach, an authority appends an expiration date to each attribute. Meanwhile, each ciphertext also associates a validity period. Only if the valid time of a ciphertext is before the expiration date of a user's attributes can the user decrypt the ciphertext. However, this mechanism requires interaction between users and the authority for revocation, and thus does not realize immediate attribute

revocation. Hur and Noh [2011] take advantage of the CP-ABE and the group key distribution mechanism to construct an access control scheme, which efficiently supports both attribute revocation and user revocation. In their construction of ABE, a trusted attribute authority takes charge of the release, revocation, and update of users' attribute keys, while the cloud server conducts the operations of group user revocation and the data re-encryption after revocation. Wu [2014] proposes a CP-ABE construction supporting the attribute revocation. In this scheme, the revocation list is embedded in the ciphertexts, and none of the users' private keys will be affected by the revocation process.

Yang et al. [2013a] design an access control framework for cloud storage systems based on a CP-ABE scheme. In the framework, each attribute is assigned with a version number. When an attribute of a user is disabled, the authority generates a new version key and an update key for this revoked attribute such that all the remaining users update their secret keys with the update key. The components associated with the revoked attribute are also updated to the latest version. Since privilege revocation can be performed efficiently at the attribute level, the scheme achieves fine-grained access control and allows dynamic changes of user access privileges. A similar revocation mechanism for the multiauthority cloud storage scenario is presented in Yang et al. [2013b].

4.2.3. System Efficiency. In essence, an ABE system is a public-key cryptography system. The security of ABE constructions still depends on a variety of mathematical problems. Most schemes based on mathematical assumptions involve complex computation of bilinear maps, resulting in high computation costs. In addition, ciphertext size and encryption/decryption time in these schemes increase with the increase of access structure complexity. In order to improve the efficiency of ABE in cloud storage, researchers started utilizing the idea of outsourcing computation operations [Green et al. 2011; Yang et al. 2013b; Li et al. 2012, 2013; Yu et al. 2010; Sahai et al. 2012; Yang et al. 2013a, 2013b]. In these schemes, large amounts of calculations, such as encryption/decryption, key generation, and attribute/user revocation operations, are delegated to external clouds.

Green et al. [2011] construct an ABE scheme based on an outsourcing decryption technique. The core idea behind the scheme is to use the PRE technique, in which two keys are produced to achieve the goal. The first one is a short key that has to be kept secret by the user, and the second one, called the transformation key, is shared with a proxy. When decrypting data, the proxy uses the transformation key to transform the original ciphertext into a compact El Gamal ciphertext, which can be easily decrypted by the user with his or her private key. In this transformation way, the decryption time of users is significantly reduced. In DAC-MACS [Yang et al. 2013b], a token-based decryption outsourcing approach is proposed. During data decryption, a user sends his or her secret keys to a cloud server and requests the server to generate a token for the decryption. Next, the ciphertext can be decrypted by the user with the decryption token retrieved from the cloud server and the global secret key. This outsourcing decryption technique greatly reduces the communication and decryption overhead.

Li et al. [2012] propose an outsourced ABE construction with delegated encryption. The construction is able to delegate encryption for any access policy, instead of a special hybrid access policy, such that the computation overhead at the user side for data encryption is sharply reduced. Further, Li et al. [2013] present an ABE-based access control scheme that adopts outsourcing techniques for key generation and data encryption. More specifically, two CSPs are introduced in this scheme. One is the key generation CSP that is in charge of performing key distribution on behalf of the attribute authority. The other is the decryption CSP that is delegated by users to conduct

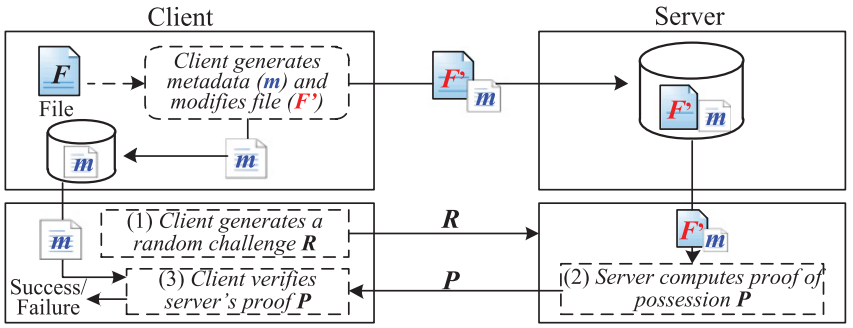

Fig. 4. Protocol for Provable Data Possession.

data decryption operations. With the two CSPs' help, it is obvious that the computation cost of the attribute authority and users is relieved.

Moreover, user revocation introduces a large amount of computational cost to a data owner, since it inevitably requires re-encrypting outsourced data to prevent the revoked users from unauthorized access of them. To address this challenging issue, Yu et al. [2010] present an access control proposal based on the PRE technique, which enables most of the intensive computation tasks to be outsourced to the cloud. Furthermore, the proposal makes full use of the lazy re-encryption technique to improve the system efficiency. The ciphertext delegation technique [Sahai et al. 2012], which allows a cloud server to update the ciphertexts periodically, is introduced to support dynamic revocation of users. The techniques of PRE and lazy re-encryption are similarly used to deal with the expensive computation issue resulting from attribute revocation [Yang et al. 2013a, 2013b].

5. INTEGRITY-GUARANTEED CLOUD DATA STORAGE

Although outsourcing data storage alleviates the owners' burden of heavy operational management, it also deprives owners' physical control of storage dependability. Under untrusted cloud environments, it is necessary to guarantee that the outsourced data are correctly stored in the cloud servers. Traditional data integrity protection schemes combine message authentication code and digital signature techniques. In these schemes, a local data duplication is required for the integrity checking, which means that a user needs to retrieve all outsourced data back before verifying if the data are corrupted. Nevertheless, with limited network bandwidth and storage on the user side, it is painful for a user to download a huge amount of cloud data. Therefore, these schemes are not suitable for integrity verification in cloud scenarios. To cope with this issue, Ateniese et al. [2007] propose blockless verifiability, that can remotely verify data integrity in a cloud. This approach allows users to check whether a server faithfully possesses the specified data even when they have no access to the data content. Different improvements [Erway et al. 2009; Chen and Curtmola 2012; Xiao et al. 2012; Wang et al. 2013, 2015; Dodis et al. 2009; Shacham and Waters 2013; Stefanov et al. 2012; Cash et al. 2013] have been proposed for better performance and usability of data integrity verification based on the blockless verifiability technique. Generally, these schemes can be divided into two classifications: PDP and POR.

5.1. Provable Data Possession

Provable Data Possession enables a user who outsources his or her data storage into a cloud to check if the cloud servers really store the correct data. The primary process of PDP is depicted in Figure 4 [Ateniese et al. 2007]. A user computes tags for each data

block in advance and generates a metadata file to specify the data block information. Then, the user sends the data block along with its tags to a server and stores the metadata file locally. When the user needs to verify the data in the cloud, he or she generates a random challenge that corresponds to a set of arbitrarily selected data blocks and submits it to the cloud. The server generates a proof of possession as a response with the queried blocks and the corresponding tags. Finally, the user can verify data possession according to the response returned from the server.

In a PDP scheme, only a small amount of metadata is stored on the user side, and the server only accesses small portions of the data to compute the verification proof. Moreover, only a limited amount of data is transferred over the network during the running of the challenge-response protocol. Therefore, the scheme significantly relieves storage overhead on the user side, I/O and computing cost on the server side, and communication overhead between of them.

5.1.1. Data Dynamics. To allow users to frequently insert new data or modify outsourced data, a PDP scheme should be able to handle data integrity verification on changed data, which is quite challenging [Wang et al. 2010, 2013].

Ateniese et al. [2008] construct a scalable PDP scheme based on symmetric key cryptography. In contrast to the basic PDP scheme [Ateniese et al. 2007], this construction algorithm supports the operations of modifying, deleting, and appending on a file block level. However, the scheme only provides limited dynamism, since the number of update operations and verification challenges is preset and limiting, and only allows append operations while not enabling insert operations. Erway et al. [2009] extend the PDP model to support update operations, and present two construction algorithms for dynamic provable data possession (DPDP) based on public-key cryptography. By using a rank-based authenticated dictionary that is constructed upon a skip list or an RSA tree, the schemes realize general update operations on data (e.g., inserting a new block, modifying an existing block, and deleting any existing data block). Its drawback is that involved RSA modular computation is very expensive, which will introduce a significant processing delay. Tan et al. [2014] propose a scheme to support fully dynamics operations based on an improved index hash table. The hash index table would no longer be stored on the verifier side, but stored on the cloud side so as to decrease the burden of the verifier.

To mitigate the impact by data corruption, particularly against a small part of data corruption, Chen and Curtmola [2012] propose a robust dynamic PDP scheme with Reed-Solomon codes [Reed and Solomon 1960] that can dynamically update any data and only requires small and constant storage on the user side. Another approach is to leverage an algebraic signature in Chen [2013]. The algebraic property of signatures is able to efficiently support random update operations in a large-scale cloud storage system. The drawback of the scheme lies in that it only provides a probabilistic security; that is, it may not be able to verify if data is corrupted without accessing all blocks.

According to Xiao's observation [Xiao et al. 2012], typical data updates to cloud storage are not inserting, modifying, or deleting operations on partial content in a file, but adding or removing an entire file. In other words, a file on a cloud is relatively stable, while the number of files on the cloud is often changing. To address data verification with this update pattern in cloud storage, Xiao et al. [2012] propose a remote data checking method, that is, Multiple-File Remote Data Checking (MF-RDC), and present two specific MF-RDC constructions. The proposed constructions utilize virtual block indices and homomorphic authenticators to enable ensured data possession in a cloud.

5.1.2. Public Verifiability. Public verifiability in cloud storage systems is important for users who have constrained computing resources [Wang et al. 2010, 2013]. It allows users to delegate data audit tasks to an independent TPA. This TPA-based mode can

provide an efficient way to guarantee the correctness of users' outsourced data, since a TPA possesses enough resources and professional skills. Moreover, audit results from a TPA can serve independent arbitration purposes, which would also be beneficial for CSPs to improve their cloud-based service platform. Wang et al. [2013] develop a privacy-preserving public auditing scheme by integrating a public key-based homomorphic linear authenticator and random masking. The scheme resorts to a TPA to perform audit operations without disclosing any knowledge about data contents to the TPA. It can achieve batch auditing tasks that are requested by multiple users, and thus, the auditing efficiency for multiuser scenarios is improved.

To handle the case when a user cannot access a cloud but wants to verify data, data possession checking can be delegated to a proxy. Wang [2013] introduces the concept of proxy-provable data possession (PPDP) and design a PPDP protocol to enable verification delegation by using proxy cryptography. In the design, a user can delegate this task to a proxy to perform remote data possession checking by warrant. The proxy can enforce verification on behalf of the user when it satisfies the conditions of the warrant.

5.1.3. Multiuser Modification and User Revocation. In order to allow a group of users to modify shared data, Wang et al. [2012b] propose a privacy-preserving public auditing scheme by constructing a homomorphic authenticator with ring signatures. The scheme adopts index hash tables to manage different data blocks. An identifier in the index hash table consists of two elements: one is a virtual index of a data block, which ensures that each block of shared data is in the right order, and the other is a random number generated by a collision-resistance hash function, which ensures that each block has a unique identifier. In this sense, the scheme allows a user to efficiently perform a dynamic operation on a single data block and avoid recomputation on other blocks. However, this scheme does not consider the case of user revocation. To overcome this shortcoming, Panda [Wang et al. 2014] is proposed to ensure shared data integrity in the presence of user revocation. In Panda, besides a block identifier and a signature, each block is also encoded with a signer identifier. Signer identifiers in data blocks allow a verifier to select correct keys for data verification and enable a cloud to determine which keys are needed for user revocation. Nevertheless, Panda suffers from heavy computational overhead that is linear to the group size and the number of checking tasks. Moreover, user revocation relies on the assumption that no collusion occurs between revoked users and cloud servers. However, in real practice, the assumption may not always hold. Yuan and Yu [2013a] try to solve these problems by using two novel designs. One is polynomial-based authentication tags, which allow aggregation of tags of different data blocks, and the other is secure proxy tag update operations, which can resist cloud servers' collusion with unauthorized users.

5.1.4. Multireplica Verification. Verification of multireplicas can be handled by extending the cryptographic primitives of PDP. Curtmola et al. [2008] first proposed an MR-PDP protocol. Establishing the existence of different replicas is crucial to making PDP usable. In the solution, they suggest making a replica unique and differentiable by first encrypting the file and then masking the encrypted version with some randomness generated from a pseudo-random function (PRF). This solution removes the server's ability to cheat when a client verifies the copies of an outsourced file. As a consequent improvement, Barsoum and Hasan [2010] propose two Efficient Multi-Copy PDP (EMC-PDP) protocols: deterministic EMC-PDP and probabilistic EMC-PDP. In the first version, the CSP has to access all the blocks of the data file, while the second version depends on spot checking by validating a random subset of the file blocks. Both of them resort to the diffusion property of any secure encryption scheme for generating distinct copies

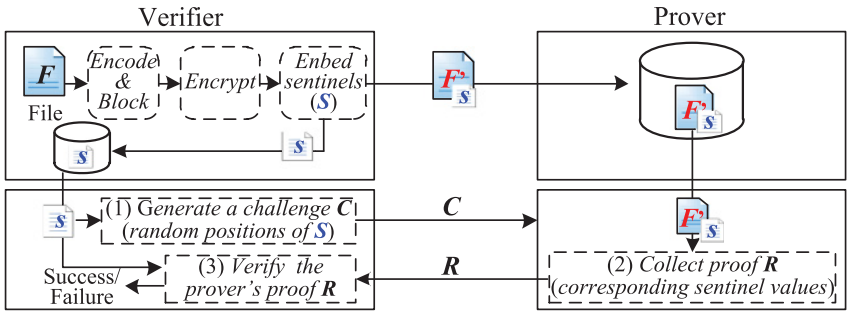

Fig. 5. Protocol for Proof of Retrievability.

and utilize the BLS homomorphic linear authenticators to offer public verifiability and unforgeable verification.

Open architectures and interfaces of the cloud storage environment enable multiple CSPs to store and manage the users' data in a cooperative way. To check the integrity of data outsourced to multiple clouds, Zhu et al. [2012] propose a cooperative verification framework. In this proposal, two fundamental techniques are applied: hash index hierarchy and homomorphic verifiable response. Wang [2015] also focuses on data integrity checking in multicloud storage and proposes an Identity-Based Distributed PDP (ID-DPDP), which eliminates the complicated certificate management. The proposed ID-DPDP scheme can achieve private auditing, delegated auditing, and public auditing under the user's authorization. The DMRDPC scheme [He et al. 2012] highlights how to reduce the communication time for multiple-replica data possession checking in geographically dispersed clouds. The idea for addressing this issue is to choose a better route to check replicas in multiple clouds instead of reducing the size of transmitted data. He et al. [2012] observe that network link capacities have geographical diversity on different links of different replicas and the bandwidths are asymmetric between two replicas. They use a Complete Bidirectional Directed Graph (CBDG) to describe the observation. The efficiency goal of multireplica verification is achieved by finding an optimal spanning tree to define the partial order of scheduling multiple-replica data possession checking.

5.2. Proof of Retrievability

Proof of Retrievability is a challenge-response audit protocol first introduced in Juels and Kaliski [2007] that can prove for a user whether a target file is intact. Using POR, a user can retrieve all file blocks from the server with high probability.

5.2.1. Basic POR Scheme. Juels and Kaliski [2007] construct a sentinel-based POR scheme, as shown in Figure 5. In the scheme, a file is carved into multiple blocks with error-correcting code, and then the encoded file is encrypted by applying a symmetric-key cipher. A set of check blocks with random values called sentinels are randomly embedded into the file. When needing data verification, a verifier challenges the prover by randomly selecting some positions of the sentinels. As the response, the prover must correctly return the corresponding sentinels back to the verifier. Finally, the verifier checks the correctness of the response. If the file is maliciously corrupted or deleted, a number of sentinels will be suppressed with high probability. In this case, it is impossible to respond correctly to the verifier. A defect of this construction is that the verification capability is limited, since the number of responses is determined by the number of sentinels.

5.2.2. Dynamic and Public Verification. With redundant coding in POR, even a single-bit modification in original data will result in a large fraction of changes in file blocks stored in a cloud server. However, this feature makes updating files more difficult. Moreover, a POR scheme should provide public verifiability in order to allow data verification delegation to a TPA.

Wang et al. [2011] propose an efficient BLS-based [Boneh et al. 2004] construction for remote data integrity verification. It supports both public auditability and data dynamics. This construction algorithm utilizes the classic Merkle hash tree technique [Merkle 1980] to achieve efficient data dynamics and offers secure public auditability with a TPA. Furthermore, the authors explore the technique of bilinear aggregate signature to enable multiple auditing requests of multiple users to be handled simultaneously. Shacham and Waters [2013] also provide a POR solution with public verification by using homomorphic authenticators and BLS signatures.

To prevent a dishonest user from accusing an honest cloud storage server of manipulating his or her data, Zheng and Xu [2011] introduce the concept of fair and dynamic proof of retrievability (FDPOR). The FDPOR scheme is built upon two new building blocks. The first building block is a novel authenticated data structure called range-based 2-3 tree. Dynamic maintenance of this tree only incurs logarithmic complexity; thus, it efficiently supports verification of dynamic data. The other one is a new incremental signature scheme named hash-compress-and-sign. It ensures the fairness of the proposed scheme. The lack of public verifiability is a drawback of this scheme; hence, it does not resort to a TPA to verify the retrievability of data.

Stefanov et al. [2012] present a dynamic POR construction algorithm. The proposed scheme employs a two-layer authentication architecture. In the lower layer, a message-authentication code (MAC) and a version of every file block are used to enable data integrity verification and ensure freshness, respectively. The upper layer is a Merkle tree structure that can be used to verify the integrity of file-block version counters. Cash et al. [2013] split data into small blocks and redundantly encode each block of data individually in order to support dynamic data. In the scheme, a user is allowed to perform arbitrary read and write operations on data at any location. The user can also initiate an audit protocol to ensure that the server maintains the latest version of data.

To further enhance efficiency and practicability of dynamic POR schemes, Shi et al. [2013] propose a lightweight dynamic POR construction scheme based on special erasure coding. Bandwidth overhead, client-side storage, and computation overhead in this scheme are comparable with the scheme using Merkle hash trees. Aiming to reduce the communication cost incurred by public verification, the POR scheme developed by Yuan and Yu [2013b] achieves public verifiability with constant communication overhead. By tailoring a specific polynomial commitment and designing a novel proof generation algorithm, proof information can be aggregated into a polynomial. The size of the proof is constant and independent of the number of elements in data blocks. Moreover, the message of verification request is also reduced to a constant level with the idea of homomorphic linear authenticators.

6. PRIVACY-PRESERVING CLOUD DATA ACCESS

Privacy refers to sensitive information about one person or a group that is expected to be secluded or hidden from others (e.g., identity, address, health, and hobbies). When users access cloud data and use cloud data services, it is necessary to preserve their privacy. In cloud data services, expectation of privacy protection exists everywhere (e.g., in sensitive outsourced data, authorized certificate, query keywords, and user access pattern). Direct sensitive data protection is normally treated as confidentiality protection, which is achieved by utilizing various encryption techniques. This section discusses privacy preservation techniques and focuses on protection of other privacy

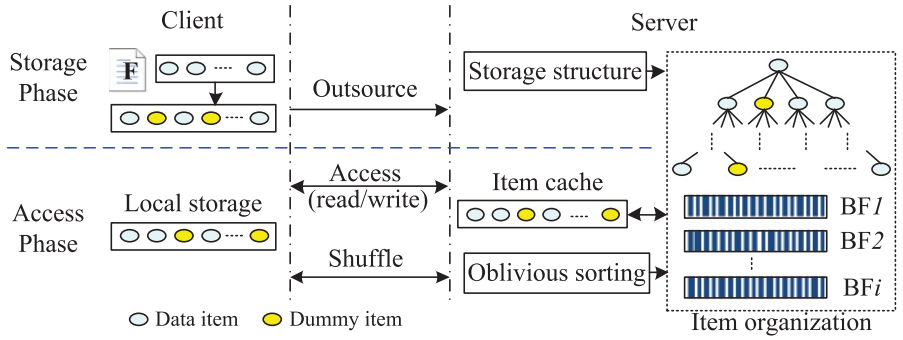

Fig. 6. ORAM mechanism illustration.

issues, including access pattern protection, query privacy protection, and user identity protection.

6.1. Access Pattern Protection

The access pattern is the sequence of accesses to a cloud system [Pinkas and Reinman 2010]. By observing the access pattern, an adversary possibly deduces various kinds of sensitive user information, including but not limited to access privilege, access frequency, and visiting habits. Access pattern protection can prevent CSPs or other malicious adversaries from inferring user access behavior according to user access requests and the corresponding server responses. To hide access patterns, different types of accesses (e.g., read and write) would be indistinguishable. In addition, adversaries cannot distinguish accesses to different data locations. Private Information Retrieval (PIR) [Chor et al. 1998; Yekhanin 2010], Oblivious RAM (ORAM) [Williams et al. 2008], and dynamically allocated data structure [De Capitani di Vimercati et al. 2011; Yang et al. 2011] are proposed to achieve these goals.

6.1.1. Private Information Retrieval. PIR is a protection approach for user access behavior in a cloud [Chor et al. 1998]. This technique primarily protects retrieved content and retrieval entrance. There are two kinds of PIR techniques: information theoretic PIR and computational PIR [Yekhanin 2010].

Information theoretic PIR depends on data copies that are stored in different distributed servers that cannot communicate with each other. It provides an absolute guarantee that each server participating in PIR protocol execution gets no information about what users access. It can be implemented by constructing locally decodable codes and converting the codes to PIRs. Shah et al. [2013] propose an erasure-coded PIR that reduces replica storage and communication overhead. Different from the information theoretic PIR approach, computational PIR is based on a certain computationally hard problem to protect user access privacy. Computational PIR does not require complete data copies so that it saves the communication bandwidth for data retrieval. This approach can effectively prevent collusion attacks among multiple servers. Unfortunately, it introduces high computational overhead [Yekhanin 2010].

6.1.2. Oblivious RAM. ORAM stores user data with some dummy data and changes its physical allocation after each visit. This approach implements independence between data and physical blocks, and achieves the goal of hiding the true access provider and access pattern [Samarati 2014]. The process of ORAM consists of two phases, *Storage* and *Access*, as shown in Figure 6. In the *Storage* phase, data is divided into several data items of equal size, mixed together with the same size of dummy items. All the items are encrypted on a client side and then are sent to a remote server. The server

stores them in a hierarchical data structure. In the *Access* phase, when the client needs to access a data item, he or she will run a multiround ORAM protocol with the server. To hide access types, a data access, whether reading or writing, is always treated as a read-and-then-write operation. After receiving a *read* request, the server will read some data items in each level of the hierarchical structure to respond. But only one data item is the target data, and others may be dumb data items. The target data item is added to the client's local storage and then is re-encrypted and placed back to the server's cache. Note that the first level functions as the cache in some schemes [Pinkas and Reinman 2010]. The *write* is performed identically to the *read*, with the exception that the new value is encrypted and inserted into the server's cache. Once the cache is full, it will be emptied into the hierarchical structure from the top to the bottom. This process is called *shuffle*, which is a complex operation and involves oblivious sorting algorithms. After the *shuffle* operation, data items will be dynamically altered in the server's storage locations. It means that the real storage location may vary corresponding to multiple accesses to the same data [Williams and Sion 2012].

Williams et al. [2008] first used ORAM for access pattern privacy. They construct an ORAM protocol that offers access pattern protection by combining a pyramid-shaped ORAM layout and Bloom filters. To reduce access costs, the scheme uses an improved sort algorithm when a higher level is emptied into the adjacent level below. Nevertheless, the response time of a query is still linear with the increase in the size of the dataset during the reordering of the bottom level. Many approaches are proposed to improve the ORAM efficiency, such as limiting shuffle operations on a portion of data (e.g., partial-shuffle ORAM [Ding et al. 2011]), reducing the number of user-server round trips (e.g., constant-round-trip ORAM [Goodrich et al. 2012a] and single-round-trip ORAM [Williams and Sion 2012]), and creating multiple threads to operate data simultaneously (e.g., parallel ORAM [Williams et al. 2012; Goodrich et al. 2012b]).

Partial-Shuffle ORAM. The shuffle operation is one of the efficiency bottlenecks of an ORAM algorithm. Reducing the number of shuffle operations can greatly improve the performance of an ORAM scheme. Ding et al. [2011] propose a partial shuffle algorithm that only shuffles and re-encrypts all *black* records when the cache is full, with no operation on any *white* records. In this way, partial shuffles can remove the correlations across query sessions and also reduce the number of re-encryption operations. Stefanov et al. [2013] introduce a path ORAM with required small client storage. The implementation of each ORAM access does not need to perform experienced deamortized oblivious sorting and construct an oblivious cuckoo hash table, but only requires simply fetching and storing a single path in the tree. Path ORAM achieves asymptotic efficiency due to smaller client-side storage and no shuttle operations.

Constant/Single-Round-Trip ORAM. Generally, an ORAM interactive protocol requires lots of client-server round trips. Nevertheless, these interactions result in great communication costs. In particular, with network delays, a large online query will incur significant latency. Goodrich et al. [2012a] address the issue by using constant amortized rounds of communications between users and servers for each data access. In this protocol, the number of round trips is decided by the amount of client-side storage. Moreover, with the help of a Bloom filter and a randomized shell sort, SR-ORAM [Williams and Sion 2012] enables a user to fold an entire interactive sequence of ORAM requests into a single query object without sacrificing privacy. It can achieve better efficiency for a storage-free user even without interactivity between users and servers.

Parallel ORAM. To further improve ORAM performance, Williams et al. [2012] propose a new parallel ORAM scheme called PrivateFS. It enables multiple users to access

remote storage simultaneously, while preventing both data information and user access patterns from reveal. In PrivateFS, a new deamortization construction is introduced to process concurrent queries with data reshuffling. This scheme loosens multiquery synchronization and guarantees low client latencies. By leveraging probabilistic encryption and stateless ORAM simulation, Goodrich et al. [2012b] construct another parallel ORAM scheme that ensures access pattern privacy under arbitrary operations on the outsourced data triggered by multiple users in a group.

6.1.3. Dynamically Allocated Data Structure. This line of work [De Capitani di Vimercati et al. 2013; De Capitani di Vimercati et al. 2011, 2013; Yang et al. 2011; Pang et al. 2013] provides access pattern confidentiality by exploiting dynamic data allocation. These approaches aim at destroying the correlation of physical blocks and the real contents. In general, they are implemented on a dynamically allocated index structure (e.g., a B^+-tree, a hash table, or a flat index) that guarantees private and efficient access to cloud data.

A shuffle index is introduced as a dynamic data allocation structure [De Capitani di Vimercati et al. 2011]. This approach provides pattern confidentiality by combining cover searches, cached searches, and data shuffling: (1) cover searches hide the target of an access within a set of other potential targets, (2) cached searches enable repeated access to node content to be indistinguishable from nonrepeated access for a server; and (3) data shuffling breaks the correlation of tree nodes and data blocks. The scheme is extended in De Capitani di Vimercati et al. [2013] to support concurrent accesses to data and searches over multiple indexes. Several distinguishing versions of the data index are used to achieve the concurrency. These versions are coordinated and applied to the main data structure at regular intervals.

Multiple indexes are supported by defining secondary shuffle indexes that are combined with the primary index in a single data structure. Using such differential versions and a combined index structure, the extended solution guarantees access pattern privacy. Another extension of the shuffle index [De Capitani di Vimercati et al. 2013] is given for addressing the access confidentiality in a distributed scenario. The proposal introduces a new protection technique, namely, shadow. This technique provides each server with the view that it is the sole data store server. Compared with the use of a single server, the protection measure of the distributed index will be stronger.

Yang et al. [2011] propose a lightweight scheme to preserve the privacy of a long-term data access pattern. The goal is achieved by combining three protection techniques: adopting dummy data items, swapping, and repeated patterns. The combination makes each data access indistinguishable from servers. Pang et al. [2013] propose a privacy-enhanced PB^+-tree index that ensures there is high uncertainty in accessed data. The scheme hides the order of leaves in an encrypted B^+-tree. Particularly, it binds all the tree nodes and puts them into buckets, and exploits homomorphic encryption techniques to stop the malicious users from locating the correct tree nodes found out by range queries.

6.2. Query Privacy Protection

Recent research focuses on query privacy preservation over ciphertexts outsourced to a cloud. Query privacy includes index privacy, keyword privacy, and trapdoor unlinkability [Sun et al. 2013; Li et al. 2013; Cao et al. 2014; Wang et al. 2014]. Index privacy demands that index information of original sensitive data is not leaked; keyword privacy means that any query keyword cannot be deduced from trapdoors of query requests; and trapdoor unlinkability means that relationships of trapdoor cannot be deduced from multiple query trapdoors, which implies that it is not decidable

no matter whether different trapdoors of query requests are originated from the same keywords [Wang et al. 2014].

6.2.1. Index Privacy. It is essential to construct a secure indexing function to map indexes to data. With a secure indexing function, relationships between original data and indexes could be hidden such that sensitive data leakage is prevented during data search and outsourcing. There are three main mapping modes [De Capitani di Vimercati et al. 2011, 2013]: (1) one-to-one mapping mode, that is, direct index, in which each plaintext is mapped to a different index and each index corresponds to a different plaintext; (2) one-to-many mapping mode, that is, flattened index [Wang and Lakshmanan 2006], in which each plaintext is mapped into a set of index values, while each index is mapped to a unique plaintext; and (3) many-to-one mapping mode, that is, bucket index [Ceselli et al. 2005], in which different plaintexts are mapped to the same index value, but each plaintext is mapped to only one index. Unfortunately, these three index modes still face potential leakage problems by exploiting additional horizontal knowledge and vertical knowledge [De Capitani di Vimercati et al. 2013]. Obviously, the more accurate indexes are, the greater the risks of private information disclosure are. Therefore, in order to completely prevent information leakage raised by indexes, a combination of the flattened index and bucket index may be helpful.

6.2.2. Keywords Protection and Trapdoor Unlinkability. In general, a trapdoor generation function is required to transform query keywords into special trapdoors before searching on encrypted data. To enable keyword privacy protection and trapdoor unlinkability, a large quantity of schemes are proposed, for example, using virtual keywords and random numbers during keyword search [Sun et al. 2013; Örencik and Savaş 2014], expanding query scopes, and randomizing query results.

In order to effectively protect query keywords and trapdoor, Sun et al. [2013] adopts a variety of techniques, for example, adding virtual keywords in queries, using a separated query vector, and introducing random factors to calculate the similarity of query results. By introducing a random number into the trapdoor function, the same query keywords will produce two different query trapdoors [Cao et al. 2014; Sun et al. 2014]. Moreover, optional parameters used in the search will generate confusing results so as to prevent clouds from learning relationships between trapdoors and keywords [Cao et al. 2014]. A combined approach with kNN encryption technology and random numbers is proposed to guarantee the unlinkability between index privacy and query trapdoors [Wang et al. 2014]. Other confusion methods leverage the dumb keyword query and response randomization method [Örencik and Savaş 2014].

For secure search on key-value stores, Hu et al. [2014] propose an oblivious index traversal framework and integrate various techniques in the framework, for example, database indexing techniques, conditional oblivious transfer, and homomorphic encryption. Within this framework, a secure search protocol is developed based on B^+ tree and R tree indexes. The protocol can resist traceablility from service providers during query processing, and hence it provides keyword privacy preservation.

6.3. User Identity Protection

When using a cloud data service, a user always hopes to become authorized while keeping his or her identity secret from other parties, including CSPs and TPAs. To achieve this goal, several signature and anonymity schemes are proposed [Wang et al. 2012a, 2012b, 2013; Li and Li 2006; Nabeel and Bertino 2014; Nabeel et al. 2013; Shang et al. 2010; Jung et al. 2013].

6.3.1. Ring/Group Signature. Ring signature and group signature techniques can be adopted to conceal user identity information. In the proposed integrity verification

schemes [Wang et al. 2012a, 2012b], the authors use the two signature techniques to prevent a TPA from obtaining any user information during signature verification. However, these schemes do not scale well if users in a group frequently change or a user group becomes larger. These schemes adopt public-key cryptography. As long as a group member changes (e.g., joining or leaving the group), the private key to generate signatures has to be reproduced and delivered to each authorized member of the group, and the public key needs to be updated accordingly. To address this problem, Wang et al. [2013] propose a privacy-preserving public verification scheme for shared cloud data. In the scheme, a shared group private key is used to facilitate cloud data sharing. It utilizes dynamic broadcast encryption [Delerablée et al. 2007] to securely distribute the group private key to existing users, and outsources the recomputation of signatures on shared data to the cloud by exploiting proxy resignatures [Blaze et al. 1998]. Therefore, the scheme achieves privacy-preserving updates and is efficient to handle secure updates in a large-scale group.

6.3.2. Anonymous Access. It is more challenging to enable identity protection when enforcing data access control by an untrusted server. Oblivious attribute certificates (OACerts) are proposed to implement a server access control policy [Li and Li 2006]. The main idea is to use the attribute value in an OACert certification instead of a user identity to perform access control, while not disclosing any attribute values to a CSP. Oblivious Commitment-Based Envelope (OCBE) [Li and Li 2006] is a new cryptographic protocol that provides a way to obliviously deliver a message to users. Based on the OCBE protocol, a secure cloud storage scheme [Nabeel et al. 2011] implements fine-grained and flexible access control, while ensuring that the CSP learns nothing about user identity attributes. Similarly, Nabeel and Bertino [2014] and Nabeel et al. [2013] adopt Pedersen commitments [Pedersen 1992] and the OCBE protocol to enable user identity privacy protection. The proposed schemes use user tokens, instead of user identity, to enforce access control. To preserve privacy during data upload, Shang et al. [2010] design a selective content distribution scheme that adopts ABE-based fine-grained access control policies and OCBE protocols. In the scheme, the data publisher will not learn anything about user identity attributes and the choice of access control policies. Jung et al. [2013] present an anonymous access control scheme based on ABE. By decomposing a center attribute authority to multiple ones while preserving tolerance to compromise attacks on the authorities, it effectively solves the problem of identity privacy preservation.

7. SUMMARY AND OPEN ISSUES

In this section, we will briefly summarize the research advances of data and privacy protection solutions in the cloud and further point out a few challenging issues for more secure and efficient cloud data services. Also, several important issues are discussed.

7.1. Confidentiality-Assured Cloud Data Service

SE and HE techniques can efficiently enable secure searching and computing over outsourced data in the cloud, respectively. The primary concern of an SE scheme is how to build secure keyword indexes and achieve various search functionalities on encrypted data, such as multikeyword search, fuzzy-keyword search, search on dynamic data, ranked search, authorized search, and verifiable search. The hot research aspects of HE include FHE construction, optimization, and implementation. Table V summarizes the research topics of SE and HE surveyed in previous sections.

A fewopen issues of this field that remain for future research are as follows:

Flexible and Verifiable Search over Ciphertexts. Users always expect that a cloud system is able to perform flexible search over encrypted data just like it does over

Table V. A Summary of Confidentiality-Assured Cloud Data Services

Techniques	Research Aspects		Representative Schemes
Searchable encryption	Secure index	Keyword-based index	Wang et al. [2012] and Kamara et al. [2012]
		Document-based index	Wang et al. [2014] and Örencik and Savaş [2014]
		Tree-based index	Lu et al. [2012] and Sun et al. [2013]
	Search functionality	Single-keyword search	Boneh et al. [2004]
		Multikeyword search	Wang et al. [2014] and Örencik and Savaş [2014]
		Fuzzy-keyword search	Kuzu et al. [2012] and Wang et al. [2014]
		Search on dynamic data	Naveed et al. [2014] and Cash et al. [2014]
		Ranked search	Cao et al. [2014] and Örencik and Savaş [2014]
		Authorized search	Zheng et al. [2014] and Sun et al. [2014]
		Verifiable search	Zheng et al. [2014]
Homomorphic encryption	FHE construction	Based on ideal lattices	Gentry [2009b] and Smart and Vercauteren [2010]
		Over the integers	Van Dijk et al. [2010] and Cheon et al. [2013]
		Based on LWE or RLWE	Brakerski et al. [2012] and Brakerski and Vaikuntanathan [2014]
	FHE optimization	Ciphertext-size control	Smart and Vercauteren [2010]
		Key-generation optimization	Gentry and Halevi [2011]
		Calculation-dimension reduction	Gentry et al. [2012c]
	FHE implementation	Message encoding	Naehrig et al. [2011]
		Batch mode	Gentry et al. [2012b]
		SIMD	Gentry et al. [2012a] and Smart and Vercauteren [2014]

plain data. It not only supports complex queries of multikeyword Boolean calculation but also provides range queries of a subset or an interval. Moreover, it should even offer arithmetic queries of summation or averaging. The current SE solutions [Li et al. 2014; Cao et al. 2014; Wang et al. 2014; Örencik and Savaş 2014] cannot meet these requirements. Moreover, ensuring the correctness, completeness, and freshness [Samarati 2014] of the query results is another important issue.

Efficient Design and Implementation of Homomorphic Encryption. Gentry's bootstrapping technique [Gentry 2009a] is the only known scheme to obtain a pure FHE scheme [Gentry et al. 2012a]. This construction algorithm requires homomorphic decryption to control noise growth. Nevertheless, the computational complexity is very high. To enhance the practicability of this FHE scheme, it is essential to construct more efficient homomorphic decryption circuits. SIMD enables an efficient way to implement an FHE scheme. However, most of the schemes cannot be directly implemented with SIMD. Moreover, in some scenarios (e.g., medical or financial applications [Naehrig et al. 2011]), the number of operation types is limited. Therefore, the operation efficiency could be greatly improved if an SwHE implementation only enables some special operation on ciphertexts.

7.2. Owner-Controlled Cloud Data Sharing

Access control based on selective encryption and ABE can effectively achieve owner-controlled data sharing in an untrusted cloud environment. Nonetheless, their access

Table VI. A Summary of Owner-Controlled Cloud Data Sharing

Techniques	Research Aspects		Representative Schemes
Selective-encryption-based access control	Privilege control	Read access	De Capitani di Vimercati et al. [2010] and De Capitani di Vimercati et al. [2010]
		Write access	Raykova et al. [2012] and De Capitani di Vimercati et al. [2013]
	Policy update	Two-layer encryption	De Capitani di Vimercati et al. [2010] and Nabeel and Bertino [2014]
		Proxy technique	De Capitani di Vimercati et al. [2010]
ABE-based access control	Access structure	Tree-based structure	Goyal et al. [2008] and Ibraimi et al. [2009]
		LSSS matrix	Green et al. [2011] and Lewko et al. [2010]
	Outsourcing computation	Outsourcing encryption	Li et al. [2012]
		Outsourcing decryption	Yang et al. [2013b] and Li et al. [2013]
		Outsourcing key generation	Li et al. [2013]
		Outsourcing ciphertext update	Yang et al. [2013a] and Sahai et al. [2012]
	Revocation	User revocation	Yu et al. [2010]
		Attribute revocation	Yang et al. [2013a, 2013b]

control mechanisms are different. Selective-encryption-based access control is enforced by using equivalent encryption policies and key derivation techniques. Authorized access in an ABE-based scheme depends on whether the attributes attached to a user's key (ciphertexts) meet the access policy embedded into ciphertexts (a user's key). As summarized in Table VI, a few selective-encryption-based schemes address *read* and *write* privilege control and access policy update issues. Existing ABE-based schemes attempt to achieve expressive access structure, efficient user/attribute revocation, and secure outsourcing computation.

Several issues related to access control in untrusted cloud environments that need to be further explored are as follows:

Access Policy Enforcement and Update. Different applications have different requirements for access authorization. For instance, in an online resource subscription scenario [De Capitani di Vimercati et al. 2012a], subscribers' access privileges should be automatically controlled according to their subscribed periods. Another instance is in multiowner situations [De Capitani di Vimercati et al. 2010], where users' access permissions depend on different owners' authorization policies. Besides, some applications need not only to authorize the read operation but also to control the write privilege. Nevertheless, none of the existing schemes [Nabeel and Bertino 2014; De Capitani di Vimercati et al. 2013; De Capitani di Vimercati et al. 2010; De Capitani di Vimercati et al. 2010] can fully achieve the aforementioned goals for these applications. Hence, it is helpful to have an access control scheme based on selective encryption such that flexible access control policies and efficient policy updates can be enabled according to diversified application requirements.

Secure Outsourcing Computation and Efficient Revocation in ABE. The main drawback in ABE is that the enforcement of complex access policies is extremely expensive [Li et al. 2012, 2013] and policy enforcement becomes a performance bottleneck in an ABE-based access control system. Some ABE-based solutions introduce outsourcing techniques to address this challenge. However, most of the existing solutions are built upon certain security assumptions, which may not hold in real practice. Therefore, it will be interesting to design a secure ABE-based scheme that relaxes these

Table VII. A Summary of Integrity-Guaranteed Cloud Data Storage

Techniques	Research Aspects		Representative Schemes
PDP	Data dynamics	Skip list	Erway et al. [2009]
		Hash table	Wang et al. [2012b]
	Public verifiability	TPA-based pattern	Wang et al. [2013]
		Proxy PDP	Wang [2013]
	User revocation	Proxy resignature	Wang et al. [2015]
		Proxy tag update	Yuan and Yu [2013a]
	Multireplica verification	MR-PDP	Curtmola et al. [2008]
		EMC-PDP	Barsoum and Hasan [2010]
		ID-DPDP	Wang [2015]
		DMRDPC	He et al. [2012]
POR	Data dynamics	Merkle hash tree	Wang et al. [2011]
		Range-based 2-3 tree	Zheng and Xu [2011]
		Balanced Merkle tree	Stefanov et al. [2012]
	Public verifiability	BLS signature	Shacham and Waters [2013] and Wang et al. [2011]
		Polynomial commitment	Yuan and Yu [2013b]

assumptions. Another issue in ABE is how to efficiently revoke attribute and/or user. Existing schemes may not be able to achieve efficiency, flexibility, and security during revocation. For instance, some schemes need users to interact online with the authority [Bethencourt et al. 2007], and some cannot support instant revocation [Yu et al. 2010]. In particular, a multiauthority case [Jung et al. 2013] cannot resist the collusion attacks between the revoked users and the authority.

7.3. Integrity-Guaranteed Cloud Data Storage

PDP and POR are two popular data verification techniques in cloud storage systems. Both of them can enable a user to understand whether outsourced data is lost or corrupted by leveraging a challenge-response protocol. The main differences between them lie in two aspects [Cash et al. 2013]. First, a POR audit guarantees that a server maintains knowledge of all of the client data, while a PDP audit only ensures that a server stores most of the client data. With PDP, a server may lose a small portion of data but it can still have a high probability of passing an audit. Second, a POR scheme enables data retrievability from a cloud by redundantly encoding the data stored on servers. However, a PDP scheme only checks if data is intact and does not offer data recoverability. The main research aspects and representative schemes of PDP and POR are shown in Table VII.

In order to achieve practical verification schemes, some issues still need to be further investigated:

Efficiency and Scalability of PDP Schemes. An integrity verification scheme should be able not only to enable an efficient data audit with data modification from multiple clients but also to provide user revocation and public verifiability. The costs of computation complexity, communication bandwidth, and storage overhead with existing schemes [Wang et al. 2015; Yuan and Yu 2013a] are still very high. Although the TPA-based schemes [Wang et al. 2013, 2015] enable dynamic verification with data change or user revocation in a large-scale cloud, it increases the system complexity because it involves more operations being performed by a TPA in a cloud storage system and requires a more sophisticated audit protocol. The construction schemes based on public-key cryptography also bring significant computation costs. Therefore, it is worth

Table VIII. A Summary of Privacy-Preserving Cloud Data Access

Privacy	Research Aspects		Representative Schemes
Access pattern	PIR	Information-theoretic PIR	Shah et al. [2013]
		Computational PIR	Yekhanin [2010]
	ORAM	Partial shuffle ORAM	Ding et al. [2011]
		Path ORAM	Stefanov et al. [2013]
		Parallel ORAM	Williams et al. [2012] and Goodrich et al. [2012b]
		Constant/single-round ORAM	Goodrich et al. [2012a] and Williams and Sion [2012]
	Dynamically allocated data structure	Shuffle index with B^+-tree	De Capitani di Vimercati et al. [2013] and Pang et al. [2013]
Query privacy	Index privacy	Flattened index	Wang and Lakshmanan [2006]
		Bucket index	Ceselli et al. [2005]
	Keyword privacy	Virtual keywords	Sun et al. [2013]
		Dummy keywords	Örencik and Savaş [2014]
	Trapdoors unlinkability	Nondeterministic trapdoor	Cao et al. [2014] and Örencik and Savaş [2014]
User identity	Ring/group signature	Ring signature	Wang et al. [2012b]
		Group signature	Wang et al. [2012a]
	Anonymous access	OCBE	Nabeel et al. [2013] and Shang et al. [2010]
		Attributes decomposed	Jung et al. [2013]

exploring new verification techniques to build more efficient and scalable PDP schemes with public verification.

Efficiency of POR Schemes with Dynamic Data. A POR scheme should support data integrity verification for dynamic data. Nevertheless, the redundant coding mechanism in a POR scheme makes data updating difficult. In addition, a practical POR scheme should enable data verification with the help of a third party and incur a constant communication cost during the verification process. Unfortunately, most of the existing schemes cannot achieve this goal. Some schemes [Juels and Kaliski 2007; Bowers et al. 2009] constructed by symmetric cryptography do not support public verification, while others [Wang et al. 2011; Shacham and Waters 2013; Yuan and Yu 2013b], which built upon public-key cryptography (e.g., BLS signature) introduce high computational costs. Hence, there is still a long way to go to construct more efficient POR schemes with dynamic data and public verification.

Cost Reduction by Jointly Considering Data Verification and Repair. Remote integrity verification allows users to periodically check if outsourced data is damaged, while data coding aims to efficiently achieve data retrieval by introducing data redundancy or inexpensively repairing corrupt data blocks. POR enables both remote data verification and data retrieval with a high probability. Unfortunately, it incurs a high cost to repair corrupted data. In a practical cloud storage system, it would reduce the overall cost by jointly considering the expense in data verification, repair, and retrieval operations. However, how to design an efficient scheme to achieve both data integrity and availability with a low cost is still quite challenging.

7.4. Privacy-Preserving Cloud Data Access

A few protection approaches are proposed to provide specific privacies, that is, access pattern, query privacy, and user identity. As shown in Table VIII, PIR, ORAM, and a dynamically allocated data structure are used for access pattern protection. Flattened/bucket index, virtual/dummy keyword, and nondeterministic trapdoor

techniques are adopted to provide query privacy. Ring/group signature and anonymous access techniques are utilized to protect user identity.

Different cloud users have different privacy protection requirements. When considering privacy protection for a cloud service, the following tradeoffs need to be well addressed:

Tradeoff Between Privacy and Accountability. To some extent, privacy and accountability are two opposite issues. On one hand, a cloud user is always reluctant to disclose his or her private information when using a cloud storage service. On the other hand, to prevent outsourced data abuse, data owners should trace unauthorized access by using operation logs. It is not easy to achieve both goals for a cloud. Most existing schemes only concentrate on privacy protection. An attribute-based anonymous access approach [Jung et al. 2013] is usually used to hide the user identities. However, it makes accountability more difficult. Group signatures [Wang et al. 2012a] provide anonymity for signers from the same group while simultaneously enabling the group manager to reveal the identity of a signer when needed. Unfortunately, if group members frequently change, the efficiency of these schemes will be very low because of key updating and resigning. It is necessary to explore an efficient approach to achieve both privacy and accountability goals.

Tradeoff Between Privacy and Query Accuracy. In general, search over ciphertexts is implemented through a secure index that is previously built upon plaintext keywords. The more accurate the provided indexes are, the more efficient the search execution is. However, exposure to possible privacy violations will occur [Samarati and De Capitani di Vimercati 2010]. On the contrary, more confusing indexes enable fewer leakage risks of sensitive information, which suffers from higher query overhead and inaccurate query results. Existing research [Sun et al. 2013; Örencik and Savaş 2014] enhances query privacy by adding virtual keywords or dummy keywords to query requests, which will incur more undesired results returned to a user. Hence, it is quite challenging to make a tradeoff between privacy and accuracy.

Tradeoff Between Privacy and Access Efficiency. Compared to protecting user static information (e.g., user identity, query keywords, query response), it is more challenging to protect users' dynamic information (e.g., access pattern). The existing approaches defend against privacy leakage by using PIR, ORAM, and a dynamically allocated data structure. However, these techniques result in inefficient data access. For example, to access data a user wants, the user needs to touch a whole dataset outsourced to the servers [Sun et al. 2013; Cao et al. 2014] or bear expensive computation [Williams et al. 2008; De Capitani di Vimercati et al. 2011]. Many schemes [Sun et al. 2013; Cao et al. 2014] don't protect access patterns for efficiency concerns. Consequently, some challenges will be faced en route to effectively addressing the dynamic access pattern protection.

7.5. Discussions

Key Management. To ensure data security for cloud data services, various encryption schemes are proposed, as we have surveyed in Section 3 and 4. These new cryptographic schemes can fall into two broad categories: symmetric cryptography and asymmetric cryptography. When an encryption scheme is based on asymmetric cryptography, its key management is comparatively simple. The keys can be generated through an existing Public Key Infrastructure (PKI) or a trusted party, for example, a trusted AA in an ABE scheme. Key distribution also can be easily achieved, because only a private key should be protected while a public key can be published. By contrast, the key in a symmetric cryptosystem is usually produced by users, and it must be kept secret all the time;

thus, its distribution is more difficult and challenging. Fortunately, this issue can be addressed by using traditional key distribution methods, for example, key sharing, proxy re-encryption [Blaze et al. 1998], and broadcast encryption [Fiat and Naor 1993]. In particular, the key derivation technique [De Capitani di Vimercati et al. 2010] may be utilized to achieve efficient key distribution for large-scale cloud users in a selective encryption scheme. In addition, both the private key of asymmetric cryptography and secret key of symmetric cryptography must be securely saved on the user side.

Comprehensive Security. As discussed in previous sections, different schemes have been developed to achieve data confidentiality, data integrity, and privacy preservability for various cloud services. Unfortunately, most of them only aim at part of the security properties. As far as we know, there is no one comprehensive solution that can hold all desired security properties simultaneously. We can integrate existing techniques to achieve a comprehensive security scheme in the cloud. However, simple integration will incur usability and efficiency problems since different techniques may interfere with each other. Hence, a comprehensive security scheme for efficient cloud data services needs to be further explored.

Security and Privacy Versus Efficiency. Users expect that they can get secure and efficient cloud data services. However, security and privacy are never free. A secure scheme always affects the performance of cloud services. Moreover, the more secure a scheme, the lower its performance. For instance, in order to protect data, it is enough to use cryptographic techniques. This only adds some encryption/decryption time overhead of data services. If the access pattern needs to be hidden, another protection measure such as ORAM should be taken, and more overheads will be introduced, for example, storage costs of dummy data, additional communication overheads, and time consumption of shuffle operations. It will increase the response time of cloud services and deteriorate the service quality and user experience. Thus, in a real cloud service, a secure scheme should be constructed with a tradeoff between security, privacy, and efficiency according to actual requirements.

8. CONCLUSIONS

Security and privacy are some of the most important issues of cloud data services. Several state-of-the-art security solutions have been proposed for protecting outsourced data and user privacy. In this survey, we systematically review these solutions in the literature. In particular, we discuss prominent schemes that provide data confidentiality, access control, data integrity, and privacy preservability for cloud data services.

We find that most schemes aim at achieving a tradeoff between security and functionality. We also notice the following trends in constructing a secure cloud data service. First, a security scheme gradually shifts from offering a single security attribute to providing multiple-dimension protection. Second, the state-of-the-art schemes for secure cloud data services enable rich application functions, which are more powerful to satisfy the requirements than earlier proposed solutions. Lastly, existing schemes aim to improve the efficiency by reducing the computation complexity, bandwidth cost, and storage overhead so as to enhance their practicability.

Furthermore, we proposed a few open issues in each category of solutions that need future research efforts. We hope our study will help to shape future research directions in this promising area of security and privacy preservation for cloud data services.

REFERENCES

Amazon. 2015a. AWS Security Center. (2015). http://aws.amazon.com/security/.

Amazon. 2015b. Microsoft Azure Trust Center. (2015). http://azure.microsoft.com/en-us/support/trust-center/.

Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. A view of cloud computing. *Communications of the ACM* 53, 4 (2010), 50–58.

Giuseppe Ateniese, Randal Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, and Dawn Song. 2007. Provable data possession at untrusted stores. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. ACM, 598–609.

Giuseppe Ateniese, Roberto Di Pietro, Luigi V. Mancini, and Gene Tsudik. 2008. Scalable and efficient provable data possession. In *Proceedings of the 4th International Conference on Security and Privacy in Communication Netowrks (SecureComm'08)*. ACM, 9.

Ayad F. Barsoum and M. Anwar Hasan. 2010. Provable possession and replication of data over cloud servers. *CACR, University of Waterloo* (March 2010).

Amos Beimel. 1996. *Secure Schemes for Secret Sharing and Key Distribution*. Ph.D. Dissertation. Technion-Israel Institute of Technology, Faculty of Computer Science.

John Bethencourt, Amit Sahai, and Brent Waters. 2007. Ciphertext-policy attribute-based encryption. In *Proceedings of IEEE Symposium on Security and Privacy (S&P)*. IEEE, 321–334.

Matt Blaze, Gerrit Bleumer, and Martin Strauss. 1998. Divertible protocols and atomic proxy cryptography. In *Advances in Cryptology (EUROCRYPT'98)*. Springer, 127–144.

Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. 2004. Public key encryption with keyword search. In *Advances in Cryptology (Eurocrypt'04)*. Springer, 506–522.

Dan Boneh, Eu-Jin Goh, and Kobbi Nissim. 2005. Evaluating 2-DNF formulas on ciphertexts. In *Theory of Cryptography*. Springer, 325–341.

Dan Boneh, Ben Lynn, and Hovav Shacham. 2004. Short signatures from the Weil pairing. *Journal of Cryptology* 17, 4 (2004), 297–319.

Kevin D. Bowers, Ari Juels, and Alina Oprea. 2009. Proofs of retrievability: Theory and implementation. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security (SCC'09)*. ACM, 43–54.

Kevin D. Bowers, Marten van Dijk, Ari Juels, Alina Oprea, and Ronald L. Rivest. 2011. How to tell if your cloud files are vulnerable to drive crashes. In *Proceedings of the 18th ACM Conference on Computer and Communications Security (CCS'11)*. ACM, 501–514.

Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2012. (Leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS'12)*. ACM, 309–325.

Zvika Brakerski and Vinod Vaikuntanathan. 2011. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *Advances in Cryptology (CRYPTO'11)*. Springer, 505–524.

Zvika Brakerski and Vinod Vaikuntanathan. 2014. Efficient fully homomorphic encryption from (standard) LWE. *SIAM Journal on Computing* 43, 2 (2014), 831–871.

Ning Cao, Cong Wang, Ming Li, Kui Ren, and Wenjing Lou. 2014. Privacy-preserving multi-keyword ranked search over encrypted cloud data. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 25, 1 (2014), 222–233.

David Cash, Joseph Jaeger, Stanislaw Jarecki, Charanjit Jutla, Hugo Krawczyk, Marcel-Catalin Rosu, and Michael Steiner. 2014. Dynamic searchable encryption in very large databases: Data structures and implementation. In *Proceedings of Network and Distributed System Security Symposium (NDSS'14)*. ISOC.

David Cash, Alptekin Küpçü, and Daniel Wichs. 2013. Dynamic proofs of retrievability via oblivious ram. In *Advances in Cryptology (EUROCRYPT'13)*. Springer, 279–295.

Alberto Ceselli, Ernesto Damiani, Sabrina De Capitani Di Vimercati, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. 2005. Modeling and assessing inference exposure in encrypted databases. *ACM Transactions on Information and System Security (TISSEC)* 8, 1 (2005), 119–152.

Bo Chen and Reza Curtmola. 2012. Robust dynamic provable data possession. In *Proceedings of 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW'12)*. IEEE, 515–525.

Lanxiang Chen. 2013. Using algebraic signatures to check data possession in cloud storage. *Future Generation Computer Systems* 29, 7 (2013), 1709–1715.

Jung Hee Cheon, Jean-Sébastien Coron, Jinsu Kim, Moon Sung Lee, Tancrede Lepoint, Mehdi Tibouchi, and Aaram Yun. 2013. Batch fully homomorphic encryption over the integers. In *Advances in Cryptology (EUROCRYPT'13)*. Springer, 315–335.

Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. 1998. Private information retrieval. *Journal of the ACM (JACM)* 45, 6 (1998), 965–981.

CSA. 2013. The Notorious Nine: Cloud Computing Top Threats in 2013. https://downloads.cloudsecurityalliance.org/initiatives/top_threats/The_Notorious_Nine_Cloud_Computing_Top_Threats_in_2013.pdf. (Feb. 2013).

Reza Curtmola, Juan Garay, Seny Kamara, and Rafail Ostrovsky. 2011. Searchable symmetric encryption: Improved definitions and efficient constructions. *Journal of Computer Security* 19, 5 (2011), 895–934.

Reza Curtmola, Osama Khan, Randal Burns, and Giuseppe Ateniese. 2008. MR-PDP: Multiple-replica provable data possession. In *Proceedings of the 28th International Conference on Distributed Computing Systems (ICDCS'08)*. IEEE, 411–420.

Sabrina De Capitani Di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. 2010. Encryption policies for regulating access to outsourced data. *ACM Transactions on Database Systems (TODS)* 35, 2 (2010), 12.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Giovanni Livraga, Stefano Paraboschi, and Pierangela Samarati. 2013. Enforcing dynamic write privileges in data outsourcing. *Computers & Security* 39 (2013), 47–63.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, Gerardo Pelosi, and Pierangela Samarati. 2010. Encryption-based policy enforcement for cloud storage. In *Proceedings of IEEE 30th International Conference on Distributed Computing Systems Workshops (ICDCSW'10)*. IEEE, 42–51.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. 2011. Private data indexes for selective access to outsourced data. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society (WPES'11)*. ACM, 69–80.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. 2013. On information leakage by indexes over data fragments. In *Proceedings of the 29th IEEE International Conference on Data Engineering Workshops (ICDEW'13)*. IEEE, 94–98.

Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, Gerardo Pelosi, and Pierangela Samarati. 2011. Efficient and private access to outsourced data. In *Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS'11)*. IEEE, 710–719.

Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, Gerardo Pelosi, and Pierangela Samarati. 2013. Supporting concurrency and multiple indexes in private access to outsourced data. *Journal of Computer Security* 21, 3 (2013), 425–461.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, and Giovanni Livraga. 2012a. Enforcing subscription-based authorization policies in cloud scenarios. In *Data and Applications Security and Privacy XXVI*. Springer, 314–329.

Sabrina De Capitani di Vimercati, Sara Foresti, Sushil Jajodia, Stefano Paraboschi, and Pierangela Samarati. 2012b. Support for write privileges on outsourced data. In *Information Security and Privacy Research*. Springer, 199–210.

Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, Gerardo Pelosi, and Pierangela Samarati. 2013. Distributed shuffling for preserving access confidentiality. In *Computer Security (ESORICS'13)*. Springer, 628–645.

Sabrina De Capitani di Vimercati, Sara Foresti, and Pierangela Samarati. 2014. Selective and fine-grained access to data in the cloud. In *Secure Cloud Computing*. Springer, 123–148.

Cécile Delerablée, Pascal Paillier, and David Pointcheval. 2007. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *Pairing-Based Cryptography (Pairing'07)*. Springer, 39–59.

Xuhua Ding, Yanjiang Yang, and Robert H. Deng. 2011. Database access pattern protection without full-shuffles. *IEEE Transactions on Information Forensics and Security (TIFS)* 6, 1 (2011), 189–201.

Yevgeniy Dodis, Salil Vadhan, and Daniel Wichs. 2009. Proofs of retrievability via hardness amplification. In *Theory of Cryptography*. Springer, 109–127.

Chris Erway, Alptekin Küpçü, Charalampos Papamanthou, and Roberto Tamassia. 2009. Dynamic provable data possession. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09)*. ACM, 213–222.

Amos Fiat and Moni Naor. 1993. Broadcast encryption. In *Advances in Cryptology (CRYPTO'93)*. Springer, 480–491.

Craig Gentry. 2009a. *A Fully Homomorphic Encryption Scheme*. Ph.D. Dissertation. Stanford University.

Craig Gentry. 2009b. Fully homomorphic encryption using ideal lattices. In *Proceedings of ACM Symposium on Theory of Computing (STOC'09)*, Vol. 9. ACM, 169–178.

Craig Gentry and Shai Halevi. 2011. Implementing gentry's fully-homomorphic encryption scheme. In *Advances in Cryptology (EUROCRYPT'11)*. Springer, 129–148.

Craig Gentry, Shai Halevi, Chris Peikert, and Nigel P. Smart. 2012c. Ring switching in BGV-style homomorphic encryption. In *Security and Cryptography for Networks*. Springer, 19–37.

Craig Gentry, Shai Halevi, and Nigel P. Smart. 2012a. Better bootstrapping in fully homomorphic encryption. In *Public Key Cryptography (PKC'12)*. Springer, 1–16.

Craig Gentry, Shai Halevi, and Nigel P. Smart. 2012b. Fully homomorphic encryption with polylog overhead. In *Advances in Cryptology (EUROCRYPT'12)*. Springer, 465–482.

Craig Gentry, Amit Sahai, and Brent Waters. 2013. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology (CRYPTO'13)*. Springer, 75–92.

Michael T. Goodrich, Michael Mitzenmacher, Olga Ohrimenko, and Roberto Tamassia. 2012a. Practical oblivious storage. In *Proceedings of the 2nd ACM Conference on Data and Application Security and Privacy (CODASPY'12)*. ACM, 13–24.

Michael T. Goodrich, Michael Mitzenmacher, Olga Ohrimenko, and Roberto Tamassia. 2012b. Privacy-preserving group data access via stateless oblivious RAM simulation. In *Proceedings of the 23th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'12)*. SIAM, 157–167.

Google. 2015. Syncdocs Security and Privacy. (2015). http://www.syncdocs.com/help/syncdocs-security-and-privacy/.

Vipul Goyal, Abhishek Jain, Omkant Pandey, and Amit Sahai. 2008. Bounded ciphertext policy attribute based encryption. In *Automata, Languages and Programming*. Springer, 579–591.

Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. 2006. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06)*. ACM, 89–98.

Matthew Green, Susan Hohenberger, and Brent Waters. 2011. Outsourcing the decryption of ABE ciphertexts. In *Proceedings of USENIX Security Symposium*, Vol. 2011. USENIX Association.

Jing He, Yanchun Zhang, Guangyan Huang, Yong Shi, and Jie Cao. 2012. Distributed data possession checking for securing multiple replicas in geographically-dispersed clouds. *Journal of Computer and System Sciences* 78, 5 (2012), 1345–1358.

Haibo Hu, Jianliang Xu, Xizhong Xu, Kexin Pei, Byron Choi, and Shuigeng Zhou. 2014. Private search on key-value stores with hierarchical indexes. In *Proceedings of the 30th IEEE International Conference on Data Engineering (ICDE'14)*. IEEE, 628–639.

Junbeom Hur and Dong Kun Noh. 2011. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 22, 7 (2011), 1214–1221.

Luan Ibraimi, Qiang Tang, Pieter Hartel, and Willem Jonker. 2009. Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In *Information Security Practice and Experience*. Springer, 1–12.

Ari Juels and Burton S. Kaliski Jr. 2007. PORs: Proofs of retrievability for large files. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. ACM, 584–597.

Taeho Jung, Xiang-Yang Li, Zhiguo Wan, and Meng Wan. 2013. Privacy preserving cloud data access with multi-authorities. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'13)*. IEEE, 2625–2633.

Seny Kamara, Charalampos Papamanthou, and Tom Roeder. 2012. Dynamic searchable symmetric encryption. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*. IEEE, 965–976.

Ryan Ko, Stephen Lee, and Veerappa Rajan. 2013. Cloud computing vulnerability incidents: A statistical overview. https://downloads.cloudsecurityalliance.org/initiatives/cvwg/CSA_Whitepaper_Cloud_Computing_Vulnerability_Incidents.zip. (March 2013).

Mehmet Kuzu, Mohammad Saiful Islam, and Murat Kantarcioglu. 2012. Efficient similarity search over encrypted data. In *Proceedings of IEEE 28th International Conference on Data Engineering (ICDE'12)*. IEEE, 1156–1167.

Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. 2010. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *Advances in Cryptology (EUROCRYPT'10)*. Springer, 62–91.

Jin Li, Xiaofeng Chen, Jingwei Li, Chunfu Jia, Jianfeng Ma, and Wenjing Lou. 2013. Fine-grained access control system based on outsourced attribute-based encryption. In *Computer Security (ESORICS'13)*. Springer, 592–609.

Jingwei Li, Chunfu Jia, Jin Li, and Xiaofeng Chen. 2012. Outsourcing encryption of attribute-based encryption with mapreduce. In *Information and Communications Security*. Springer, 191–201.

Jiangtao Li and Ninghui Li. 2006. OACerts: Oblivious attribute certificates. *IEEE Transactions on Dependable and Secure Computing (TDSC)* 3, 4 (2006), 340–352.

Jin Li, Qian Wang, Cong Wang, Ning Cao, Kui Ren, and Wenjing Lou. 2010. Fuzzy keyword search over encrypted data in cloud computing. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'10)*. IEEE, 1–5.

Ming Li, Shucheng Yu, Kui Ren, Wenjing Lou, and Y. Thomas Hou. 2013. Toward privacy-assured and searchable cloud data storage services. *IEEE Network* 27, 4 (2013), 56–62.

Ruixuan Li, Zhiyong Xu, Wanshang Kang, Kin Choong Yow, and Cheng-Zhong Xu. 2014. Efficient multi-keyword ranked query over encrypted data in cloud computing. *Future Generation Computer Systems* 30 (2014), 179–190.

Yanbin Lu. 2012. Privacy-preserving logarithmic-time search on encrypted data in cloud. In *Proceedings of Network and Distributed System Security Symposium (NDSS'12)*. ISOC.

Ralph C. Merkle. 1980. Protocols for public key cryptosystems. In *Proceedings of IEEE Symposium on Security and Privacy (S&P'80)*. IEEE, 122–134.

Mohamed Nabeel and Elisa Bertino. 2014. Privacy preserving delegated access control in public clouds. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 26, 9 (2014), 2268–2280.

Mohamed Nabeel, Elisa Bertino, Murat Kantarcioglu, and Bhavani Thuraisingham. 2011. Towards privacy preserving access control in the cloud. In *Proceedings of the 7th International Conference on Collaborative Computing: Networking, Applications and Worksharing*. IEEE, 172–180.

Mohamed Nabeel, Ning Shang, and Elisa Bertino. 2013. Privacy preserving policy-based content sharing in public clouds. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 11 (2013), 2602–2614.

Michael Naehrig, Kristin Lauter, and Vinod Vaikuntanathan. 2011. Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM Cloud Computing Security Workshop (CCSW'11)*. ACM, 113–124.

Muhammad Naveed, Manoj Prabhakaran, and Carl A. Gunter. 2014. Dynamic searchable encryption via blind storage. In *Proceedings of IEEE Symposium on Security and Privacy (S&P'14)*. IEEE, 639–654.

Cengiz Örencik and Erkay Savaş. 2014. An efficient privacy-preserving multi-keyword search over encrypted cloud data with ranking. *Distributed and Parallel Databases (DPD)* 32, 1 (2014), 119–160.

Rafail Ostrovsky, Amit Sahai, and Brent Waters. 2007. Attribute-based encryption with non-monotonic access structures. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*. ACM, 195–203.

HweeHwa Pang, Jilian Zhang, and Kyriakos Mouratidis. 2013. Enhancing access privacy of range retrievals over b^+-trees. *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 25, 7 (2013), 1533–1547.

Torben Pryds Pedersen. 1992. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology (CRYPTO'91)*. Springer, 129–140.

Benny Pinkas and Tzachy Reinman. 2010. Oblivious RAM revisited. In *Advances in Cryptology (EUROCRYPT'10)*. Springer, 502–519.

Mariana Raykova, Hang Zhao, and Steven M. Bellovin. 2012. Privacy enhanced access control for outsourced data sharing. In *Financial Cryptography and Data Security*. Springer, 223–238.

Irving S. Reed and Gustave Solomon. 1960. Polynomial codes over certain finite fields. *Journal of the Society for Industrial & Applied Mathematics* 8, 2 (1960), 300–304.

Amit Sahai, Hakan Seyalioglu, and Brent Waters. 2012. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *Advances in Cryptology (CRYPTO'12)*. Springer, 199–217.

Amit Sahai and Brent Waters. 2005. Fuzzy identity-based encryption. In *Advances in Cryptology (EUROCRYPT'05)*. Springer, 457–473.

Pierangela Samarati. 2014. Data security and privacy in the cloud. In *Information Security Practice and Experience*. Springer, 28–41.

Pierangela Samarati and Sabrina De Capitani di Vimercati. 2010. Data protection in outsourcing scenarios: Issues and directions. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS'10)*. ACM, 1–14.

Hovav Shacham and Brent Waters. 2013. Compact proofs of retrievability. *Journal of Cryptology* 26, 3 (2013), 442–483.

Nihar B. Shah, K. V. Rashmi, Kannan Ramchandran, and P. Vijay Kumar. 2013. *Privacy-Preserving and Secure Distributed Storage Codes*. http://www.eecs.berkeley.edu/ñihar/publications/privacy_security. pdf.

Ning Shang, Mohamed Nabeel, Federica Paci, and Elisa Bertino. 2010. A privacy-preserving approach to policy-based content dissemination. In *Proceedings of the 26th International Conference on Data Engineering (ICDE'10)*. IEEE, 944–955.

Elaine Shi, Emil Stefanov, and Charalampos Papamanthou. 2013. Practical dynamic proofs of retrievability. In *Proceedings of the 2013 ACM Conference on Computer and Communications Security (CCS'13)*. ACM, 325–336.

Nigel P. Smart and Frederik Vercauteren. 2010. Fully homomorphic encryption with relatively small key and ciphertext sizes. In *Public Key Cryptography (PKC'10)*. Springer, 420–443.

Nigel P. Smart and Frederik Vercauteren. 2014. Fully homomorphic SIMD operations. *Designs, Codes and Cryptography* 71, 1 (2014), 57–81.

Emil Stefanov, Marten van Dijk, Ari Juels, and Alina Oprea. 2012. Iris: A scalable cloud file system with efficient integrity checks. In *Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC'12)*. IEEE, 229–238.

Emil Stefanov, Marten Van Dijk, Elaine Shi, Christopher Fletcher, Ling Ren, Xiangyao Yu, and Srinivas Devadas. 2013. Path oram: An extremely simple oblivious ram protocol. In *Proceedings of the 2013 ACM Conference on Computer and Communications Security (CCS'13)*. ACM, 299–310.

Wenhai Sun, Bing Wang, Ning Cao, Ming Li, Wenjing Lou, Y. Thomas Hou, and Hui Li. 2013. 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 (ASIACCS'13)*. ACM, 71–82.

Wenhai Sun, Shucheng Yu, Wenjing Lou, Y. Thomas Hou, and Hui Li. 2014. Protecting your right: Attribute-based keyword search with fine-grained owner-enforced search authorization in the cloud. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'14)*. IEEE.

Shuang Tan, Lin Tan, Xiaoling Li, and Yan Jia. 2014. An efficient method for checking the integrity of data in the cloud. *China Communications* 11, 9 (2014), 68–81.

Marten Van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. 2010. Fully homomorphic encryption over the integers. In *Advances in Cryptology (EUROCRYPT'10)*. Springer, 24–43.

Verizon. 2015. 2015 Data Breach Investigations Report. http://www.verizonenterprise.com/DBIR/2015/. (2015).

Shalini Verma. 2012. Forecast: Consumer Digital Storage Needs, 2010–2016. http://www.gartner.com/newsroom/id/2060215. (2012).

Boyang Wang, Baochun Li, and Hui Li. 2012a. Knox: Privacy-preserving auditing for shared data with large groups in the cloud. In *Applied Cryptography and Network Security*. Springer, 507–525.

Boyang Wang, Baochun Li, and Hui Li. 2012b. Oruta: Privacy-preserving public auditing for shared data in the cloud. In *Proceedings of the 5th International Conference on Cloud Computing (CLOUD'12)*. IEEE, 295–302.

Boyang Wang, Baochun Li, and Hui Li. 2015. Panda: Public auditing for shared data with efficient user revocation in the cloud. *IEEE Transactions on Services Computing (TSC)* 8, 1 (2015), 92–106.

Boyang Wang, Hui Li, and Ming Li. 2013. Privacy-preserving public auditing for shared cloud data supporting group dynamics. In *Proceedings of IEEE International Conference on Communications (ICC'13)*. IEEE, 1946–1950.

Bing Wang, Shucheng Yu, Wenjing Lou, and Y. Thomas Hou. 2014. Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'14)*. IEEE.

Cong Wang, Ning Cao, Jin Li, Kui Ren, and Wenjing Lou. 2010. Secure ranked keyword search over encrypted cloud data. In *Proceedings of IEEE 30th International Conference on Distributed Computing Systems (ICDCS'10)*. 253–262.

Cong Wang, Ning Cao, Kui Ren, and Wenjing Lou. 2012. Enabling secure and efficient ranked keyword search over outsourced cloud data. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 23, 8 (2012), 1467–1479.

Cong Wang, Sherman S. M. Chow, Qian Wang, Kui Ren, and Wenjing Lou. 2013. Privacy-preserving public auditing for secure cloud storage. *IEEE Transactions on Computers (TOC)* 62, 2 (2013), 362–375.

Cong Wang, Kui Ren, Wenjing Lou, and Jin Li. 2010. Toward publicly auditable secure cloud data storage services. *IEEE Network* 24, 4 (2010), 19–24.

Cong Wang, Kui Ren, Shucheng Yu, and Karthik Mahendra Raje Urs. 2012. Achieving usable and privacy-assured similarity search over outsourced cloud data. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'12)*. IEEE, 451–459.

Huaqun Wang. 2013. Proxy provable data possession in public clouds. *IEEE Transactions on Services Computing (TSC)* 6, 4 (2013), 551–559.

Huaqun Wang. 2015. Identity-based distributed provable data possession in multi-cloud storage. *IEEE Transactions on Services Computing (TSC)* 8, 2 (2015), 328–340.

Hui Wang and Laks V. S. Lakshmanan. 2006. Efficient secure query evaluation over encrypted XML databases. In *Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB'06)*. VLDB Endowment, 127–138.

Qian Wang, Cong Wang, Kui Ren, Wenjing Lou, and Jin Li. 2011. Enabling public auditability and data dynamics for storage security in cloud computing. *IEEE Transactions on Parallel and Distributed Systems* 22, 5 (2011), 847–859.

Brent Waters. 2011. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *Public Key Cryptography (PKC'11)*. Springer, 53–70.

Peter Williams and Radu Sion. 2012. Single round access privacy on outsourced storage. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS'12)*. ACM, 293–304.

Peter Williams, Radu Sion, and Bogdan Carbunar. 2008. Building castles out of mud: Practical access pattern privacy and correctness on untrusted storage. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*. ACM, 139–148.

Peter Williams, Radu Sion, and Alin Tomescu. 2012. Privatefs: A parallel oblivious file system. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security (CCS'12)*. ACM, 977–988.

Qiuxin Wu. 2014. A generic construction of ciphertext-policy attribute-based encryption supporting attribute revocation. *China Communications* 11, 13 (2014), 93–100.

Da Xiao, Yan Yang, Wenbin Yao, Chunhua Wu, Jianyi Liu, and Yixian Yang. 2012. Multiple-file remote data checking for cloud storage. *Computers & Security* 31, 2 (2012), 192–205.

Zhifeng Xiao and Yang Xiao. 2013. Security and privacy in cloud computing. *IEEE Communications Surveys & Tutorials* 15, 2 (2013), 843–859.

Kan Yang and Xiaohua Jia. 2012. Data storage auditing service in cloud computing: Challenges, methods and opportunities. *World Wide Web* 15, 4 (2012), 409–428.

Kan Yang, Xiaohua Jia, and Kui Ren. 2013a. Attribute-based fine-grained access control with efficient revocation in cloud storage systems. In *Proceedings of the 8th ACM Symposium on Information, Computer and Communications Security (ASIACCS'13)*. ACM, 523–528.

Kan Yang, Xiaohua Jia, Kui Ren, Bo Zhang, and Ruitao Xie. 2013b. Dac-macs: Effective data access control for multiauthority cloud storage systems. *IEEE Transactions on Information Forensics and Security (TIFS)* 8, 11 (2013), 1790–1801.

Ka Yang, Jinsheng Zhang, Wensheng Zhang, and Daji Qiao. 2011. A light-weight solution to preservation of access pattern privacy in un-trusted clouds. In *Computer Security (ESORICS'11)*. Springer, 528–547.

Peng Yanguo, Cui Jiangtao, Peng Changgen, and Ying Zuobin. 2014. Certificateless public key encryption with keyword search. *China Communications* 11, 11 (2014), 100–113.

Sergey Yekhanin. 2010. Private information retrieval. *Communications of the ACM* 53, 4 (2010), 68–73.

Shucheng Yu, Cong Wang, Kui Ren, and Wenjing Lou. 2010. Achieving secure, scalable, and fine-grained data access control in cloud computing. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'10)*. IEEE, 1–9.

Jiawei Yuan and Shucheng Yu. 2013a. Efficient Public Integrity Checking for Cloud Data Sharing with Multi-User Modification. Cryptology ePrint Archive, Report 2013/484. (2013). http://eprint.iacr.org/.

Jiawei Yuan and Shucheng Yu. 2013b. Proofs of retrievability with public verifiability and constant communication cost in cloud. In *Proceedings of the 2013 International Workshop on Security in Cloud Computing (SCC'13)*. ACM, 19–26.

Qingji Zheng and Shouhuai Xu. 2011. Fair and dynamic proofs of retrievability. In *Proceedings of the 1st ACM Conference on Data and Application Security and Privacy (CODASPY'11)*. ACM, 237–248.

Qingji Zheng, Shouhuai Xu, and Giuseppe Ateniese. 2014. Vabks: Verifiable attribute-based keyword search over outsourced encrypted data. In *Proceedings of IEEE International Conference on Computer Communications (INFOCOM'14)*. IEEE, 522–530.

Yan Zhu, Hongxin Hu, Gail-Joon Ahn, and Mengyang Yu. 2012. Cooperative provable data possession for integrity verification in multicloud storage. *IEEE Transactions on Parallel and Distributed Systems (TPDS)* 23, 12 (2012), 2231–2244.

Received July 2015; revised January 2016; accepted March 2016