Future Generation Computer Systems 79 (2018) 337–349

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Privacy-preserving personal data operation on mobile cloud—Chances and challenges over advanced persistent threat

Man Ho Au [a], Kaitai Liang [b], Joseph K. Liu [c,*], Rongxing Lu [d], Jianting Ning [e]

[a] Department of Computing, Hong Kong Polytechnic University, Hong Kong
[b] School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, UK
[c] Faculty of Information Technology, Monash University, Australia
[d] Faculty of Computer Science, University of New Brunswick, Fredericton, NB, Canada
[e] Department of Computer Science, National University of Singapore, Singapore

ARTICLE INFO

Article history:
Received 14 September 2016
Received in revised form 12 June 2017
Accepted 16 June 2017
Available online 28 June 2017

Keywords:
Mobile could
Security
Privacy
Applied cryptography

ABSTRACT

Bring your own devices have become a new symbol of industrial and education institutional culture to date. A single individual can gain access to personal data anytime at anywhere of his/her workplace due to the advanced WiFi/5G network and cloud technology. The most convenient way for us to access to cloud data is to use personal smartphone. However, smartphone is somewhat vulnerable (because of its innate disadvantage, e.g., low security protection and limited computation resource) while encountering with malicious attacks in open network. Mobile users may be the victims of a recent new type of attack - *advanced persistent threat* (APT), since attackers may penetrate into different levels of cloud and mobile infrastructures to eavesdrop, steal and temper data. This survey paper introduces some security/privacy risks on mobile cloud in the view point of applied cryptography. Meanwhile, it provides some insights as possible solutions for the risks.

1. Background

The comScore report [1] shows that the number of increasing usage of mobile devices (up to 1.9 billions) has already exceeded that of desktop (with nearly 1.7 billions) since 2015. Besides, the average time people spend on mobile apps. has increased by 21% over the last few years conducted by a Go-Globe survey [2]. Both reports interpret a strong sign that an increasing number of people tend to spend more time in using their mobile rather than other unportable electronic devices. The massive usage of mobile devices stimulates the booming ear of all kinds of network apps., which can be available and downloaded from either Apple's iTunes or Google Play Store.

Although mobile devices connected to Internet allows Internet users to enjoy many network services and applications much like desktop, they, to a large extent, cannot fully provide excellent user experiences for their users because of their "natural-born" constraints, including limited memory, processing power and battery life. To help mobile devices to move beyond the restrictions, mobile research and industrial communities invent a new framework, *mobile cloud*, which is the convergence of mobile devices and cloud, such that device users are allowed to offload heavy storage and computational cost to cloud (Fig. 1) to reduce the local resource and energy consumption. This is especially important in the era of big data [3–5].

There are long-list advantages for us to leverage cloud to deal with storage and computation barrier in mobile devices [6–8]. For instance, when trying to find a path to a sightseeing destination, say from London Bridge to British Museum, a tourist with a mobile device does not need to spend many mobile data in downloading a full local London map embedding with all hotel, restaurant, sightseeing information, but simply reporting his location to cloud-based Global Positioning System (GPS) navigation. Take social media networking apps. as another example. While using Tinder (https://www.gotinder.com) to find friends nearby, it is unnecessary for us to download all system users' information to mobile, but just upload our current locations. Besides, by using considerable computational power of cloud, mobile devices with limited computation resource can enable users to play 3D games, to run mobile commercial systems, and even to participate into mobile-learning platforms (e.g. Litmos (https://www.litmos.com)).

Lifting weight from mobile devices to mobile cloud, however, may yield security and privacy concerns. There are various challenges incurred by the usage of mobile cloud, e.g., identity management and standardization. As we mention previously, a mobile user may upload his/her personal information to a cloud, which is

* Corresponding author.
E-mail addresses: joseph.liu@monash.edu (J.K. Liu), ningjt@comp.nus.edu.sg (J. Ning).

http://dx.doi.org/10.1016/j.future.2017.06.021
0167-739X/

trusted by the user. Nevertheless, this may endanger the privacy of the user while the cloud server is intruded by malicious Internet hackers. For example, the leaking iCould celebrity picture [9], Barclays bank client records leak incidence [10], and the most recent WannaCry ransomware attacks on NHS [11] are wake-up calls for considering user privacy in cloud.

A new type of attack called *Advanced Persistent Threat* (APT) has also caught the attention of security and privacy researchers in recent years. A typical APT often targets to a group of entities to steal valuable personal data/information via continuously computer hacking with considerable resource and infiltration strategy. Commercial organizations, business companies and even governmental institutions are usually the main targets of the threat. We have reason to believe that mobile devices are in the top of victim list as the devices nowadays are used to perform various types of Internet cloud operations, such as personal on-line banking (e.g., HSBC Mobile Banking[1]) and chat apps. (e.g., WhatsApp[2]). It will definitely incur a great amount of personal privacy and property loss if our banking account is hacked or our private chat history is disclosed on line. In this survey, we focus on some practical behaviors of mobile users to discuss the security and privacy risks in mobile cloud. Specifically, we mainly focus on the following clients' behaviors: identity authentication before building up connection with cloud, data encryption before uploading to cloud, data integrity check after data uploading, remote data search, share and computation.

Paper organization. This paper is organized as follows. We introduce the advantages of mobile cloud not only for mobile users but also for community in Section 2. In Section 3, we explore the security risks of mobile cloud. We propose some countermeasures in Section 4. We conclude the paper in Section 5.

2. Opportunities of mobile cloud

Mobile cloud is able to provide "real-time" personal and public data access for all Internet users at anytime, anywhere, on any mobile device. The great potential market of mobile cloud has been attracted great investment of Internet service providers and mobile manufactures (e.g., Samsung, Apple and Nokia).

2.1. Convenient data access for users

In addition to traditional phone services (e.g., phone call), mobile service providers can promote new and more convenient offers to their clients by leveraging the considerable computing and storage ability of mobile cloud. Mobile learning, e.g., eLearning INDUSTRY,[3] is a novel merging service in which clients are allowed to take on-line and real-time courses, upload homework, assignment and participate into real-time seminar via mobile apps. On-line learners can search what they are interested in from mobile cloud, and download unlimited but easy accessible resources from the course database, on-line universities', and even public libraries.

Clinics, hospitals and health care centers can definitely get benefit from mobile cloud service, like mobile-health (mHealth) care service (e.g. TotalMobile[4]). Getting rid of tedious paper works and wasting time in long queue, patients now can use mHealth service for doctor appointment via their portable mobile devices. Moreover, new health sensor techniques can be employed into mobile devices, such that the health condition of patients can be immediately updated to doctors for better medical treatment

Fig. 1. Mobile cloud framework.

track. The new mHealth service providers are able to not only reduce medical care fraud but also improve patient safety by making use of advanced cloud computing ability.

It is a trend that Internet users prefer to launch finance-related activities on their smartphones. A blooming period for mobile finance is approaching. Due to being equipped with powerful computational resources, mobile cloud is strong enough to support various financial behaviors, such as money transfer, and bank payment. Much like bankmobile,[5] many world-wide banks have implemented mobile on-line banking to date. Take HSBC mobile app[6] as an example. The app allows on-line banking clients to check the balance of their accounts, generate security one-time login code, and do some payment/bank transfer with text notice.

Mobile cloud game service is also another potential market. There are many new and popular game apps (e.g. ROVIO Angry Birds[7]) for iOS, Android, Windows platforms emerging every year. Nevertheless, the visual/sound effect and complex game design of those apps seriously consume smartphone's battery and memory. With help of mobile cloud, the game engine and effect/upgrade packages can be completely offloaded to cloud and meanwhile, the cloud can be used to run large computational cost algorithms (e.g., graphic rendering). The mobile cloud can also make mobile users consume less storage space for the continue version update in the sense that the old version can be stored and backed up in cloud. More importantly, cloud-based game supports on-line multi-player interface, so that multiple players can compete with each other in the same game though being in different physical locations. The competition results can be shown and shared in real-time in on-line friends group.

Last but not least, mobile cloud provides large-scale stream media store (e.g., Mobile Media[8]), large volume of social network data

1 https://itunes.apple.com/gb/app/hsbc-mobile-banking/id565993818?mt=8
2 https://www.whatsapp.com/
3 https://elearningindustry.com/
4 http://www.totalmobile.co.uk/healthcare
5 https://www.bankmobile.com/app/
6 http://www.hsbc.co.uk/1/2/contact-and-support/ways-to-bank/mobile
7 http://www.rovio.com/
8 http://www.mobilemedia.co.uk/

share, and location-based service (e.g., Magellan RoadMate[9]) for smartphone users. Considerable storage space, unlimited computational power, and convenient interface, these extremely appealing advantages of mobile cloud, that light up a bright prospective for diverse mobile services.

2.2. Enlightenment for communities, industries and authorities

It is clear that mobile cloud does create visible and invisible opportunities for other entities including academic researchers, industries and authorities. The academic communities may be inspired to design more lightweight, secure and up-to-date protocols/systems that relieve the workload and worry of mobile users in defending open network malicious attacks. With the assistance of mobile cloud, industries and companies are able to provide more powerful data computing, faster data processing, and more considerable storage services for their clients, for example, Portable Genomics (http://www.portablegenomics.com/#!home) offers convenient genome data analysis services to smartphone users. The authorities, such as local transportation center, may make use of mobile cloud to fulfill real-time public surveillance, such as traffic real-time report and forecast in mobile platform.

Furthermore, the deployment of mobile cloud yields an opportunity of collaboration among mobile device users, mobile service providers, and local authorities. The collaboration of the three parties, definitely, contributes more correct, accurate and trustworthy outcomes compared to the only-one-side-work mode. Moreover, mobile device users need to worry about battery, memory and computation limitation no more with help of service provider/cloud server. For example, mobile data encryption and decryption would be partially offloaded to a cloud server, so that the users only are required to perform a light piece of computation task. The collaboration, however, should ensure that even the service provider colludes with malicious attackers, they still cannot gain access to the users' data. Working together may be an effective way to tackle efficiency, privacy and security problems.

The surveillance and authorization of local authorities for mobile cloud service providers are another effective approaches to guarantee that the services are secure and trustworthy. Malicious service providers should be tracked down and further punished under the local security and privacy law. Besides, new data protection and user privacy law enforcements are always desirable to be refined under the up-to-date situation. There have been some standards for mobile devices/networks (e.g., 3GPP[10]) and cloud computing (e.g., ETSI[11]). However, there is no standard clearly targeting to mobile cloud. It is now a great opportunity to put standardization of mobile cloud on schedule.

3. Challenges in mobile cloud

Taking advantage of advanced mobile and network technology, mobile users may enjoy various on-line activities, for example, accessing social network information, watching on-line video (e.g. YouTube), checking email (e.g. Gmail), managing on-line banking (e.g. HSBC on-line bank), and on-line shopping (e.g. Amazon, eBay). As illustrated in a U.S. smartphone use survey in 2015,[12] Internet browsing and email checking are two main behaviors of mobile users in addition to basic mobile operations, like text and phone call. In terms of the Internet browsing, over 50% of mobile users may use their devices to read health information and fulfill on-line banking, while estate hunting, job searching, on-line education, and government service look-up are other frequent behaviors.

Standing at the viewpoint of applied cryptography by the side of mobile cloud users, this paper investigates some security and privacy risks based on the following frequent user operations: (1) (login) authentication between client and mobile cloud; (2) outsource data from local mobile device to remote cloud, and data integrity check; (3) search and share client's remote data, and remote data computation. Meanwhile, the paper will show that existing tools do not fully satisfy the practical security needs of mobile cloud users.

3.1. Mobile clients authentication

While talking about user authentication, we usually consider the single way "client to cloud authentication mode", where server will only allows valid clients to access to cloud system if the clients pass the corresponding "identity check". This type of "proof of identity" is extremely helpful in protecting cloud clients' data privacy. For instance, a user of a cloud-based storage system (e.g. Box[13]) can only gain access to his/her own on-line data by using a unique and personal pair of user name and password. To date, there are various mobile-to-cloud authentication methods that have been proposed in the literature. They can mainly be categorized into three branches, namely knowledge-based, possession-based and biometric-based authentications. Leveraging one of the approaches individually that may yield some potential security concern.

Using user name and password for (knowledge-based) authentication [12–14] that is one of most user-convenience authentication mechanisms. Some of the existing systems are already built in the mobile setting. For instance, Acar et al. [13] introduce a single password authentication in which a mobile device/hardware must be fully trusted. Specifically, the hash value $Hash(pw)$ of a user's password pw is regarded as a key to encrypt a randomly string K generated by the mobile user (i.e. $CT = Encrypt(Hash(pw), K)$), and the encryption is further stored in the mobile device; meanwhile, the user's identity ID and the string K are delivered to a cloud server. When trying to login the server, the user needs to send his/her ID to the server who returns a challenge $chal$. The user further taps password pw into the mobile, such that the mobile can recover $K = Decrypt(Hash(pw), CT)$ and next to compute a $MAC(K, chal)$ for the server. With knowledge of K and $chal$, the server can check the validity of the MAC value. To secure password from being easily guessed, mobile clients usually are requested to use a long and complex enough combination, (e.g. using image as password [15]), or to install some password manager apps (e.g. SafeInCloud - https://www.safe-in-cloud.com/) to "securely" manage their passwords. However, passwords may suffer from some pitfalls because of some human information processing limitations. The same password may be reused in different systems that definitely increases the risk of domino system crack. Moreover, clients may pick up their passwords from some special and meaningful information, such as birthday and family member's name, so that the passwords will be easily revealed once the related information is known by attackers.

Possession-based authentication enables mobile clients to make use of something they hold to fulfill identity authentication. Accordingly, we may choose to use secure USB token, one-time password [16], or embed a public key infrastructure (e.g. [17]) into mobile devices, to strengthen the security of authentication. But this approach requires more computational cost and energy consumption, for example, key management might be a problem

9 http://www.magellangps.com/Store/SmartGPS-Products/Magellan-RoadMate-On-The-Go-App-for-iOS

10 http://www.3gpp.org/

11 http://www.etsi.org/standards

12 http://www.pewinternet.org/2015/04/01/us-smartphone-use-in-2015/

13 https://www.box.com/en-gb/home

for mobile devices upon the usage of public key infrastructure. Furthermore, the possessed device might be stolen by adversary or lost by careless token holder, such that they may be misused in the future.

The biometric-based authentication [18–20] can be used to provide a unique and portable way for client identification via making use of client's bio-characteristics, such as voice, face, iris and fingerprint [21]. How to secretly store and process personal bio-information in authentication is a major privacy concern. Since one's biometric information is unique, if adversary obtains the information by hacking into the client's mobile device, it will bring severe harm to personal privacy.

To achieve stronger authentication security, multi-factor authentication systems (e.g. [22–25]) have been proposed in the mobile cloud setting. In a multi-factor authentication mechanism, more than one factor/technique-base are implemented into identity verification. A device and a cloud server will need to share some secret information as a preparation for future authentication, such as $Hash(pw)$ or random string K. The authentication phase will take 2–3 factors' information (for example, password and secure token, fingerprint and password) into the "challenge-and-respond" interaction (see Fig. 2). The multi-factor mechanism strengthens the difficulty of cracking the verification in the sense that malicious attackers have to compromised all factors to lead to a successful attack. Because of its high security guarantee, many companies have employed multiple factors for clients authentication, e.g., SafeNet (http://www.safenet-inc.com/), Microsoft Azure (http://azure.microsoft.com/en-us/) and rackspace (http://www.rackspace.com/).

Nonetheless, the "most-secure-look" multi-factor authentication still suffers from thorny challenges incurred by factor update/revocation, delegation in authentication, and bidirectional authentication (see Table 1). We note that by "Cloud to Client" authentication mode we mean a cloud server may need to show a client that the service provider/server and the corresponding service are both authenticated by some trusted public authority and meanwhile, the client is protected by some necessary law enforcement. Consider a use case of eHealth mobile app. If the service provider is not under correct regulation which is bounded by patient privacy law, it may maliciously leak patients' health record for commercial benefit.

Factor update/revocation is necessary while a factor is compromised by attackers. How to effectively and efficiently detect the corrupted factor and further issue a "fresh" factor for both cloud and client is a challenging task. A delegation for identity verification is very common in daily life. For instance, an on-line eBay user is re-directed to a third-party payment platform. Here, the first login cloud service provider should take responsibility for the second platform authentication, so that no privacy information will be "curiously" collected by the latter, e.g., the client's transaction history. The authentication delegation may also happen in client side in the sense that a client Alice requires another client Bob to login a cloud storage system to use the data/service on behalf of Alice. Some naive solutions, such as requesting the server to modify access control list which allows Bob to "enter" the system, may work. But allowing the server to know the delegation between Alice and Bob may lead to high risk of secret leakage in some business scenarios. Therefore, a privacy-preserving client-side authentication delegation is desirable. Last but not least, a bidirectional authentication system should be considered (i.e. client $\leftrightarrow$ cloud) due to unpredictable security risks in an open network. The growing number of network phishing and fake cloud services have been taking serious influence in mobile cloud security. Mobile clients must need a way to verify a cloud service provider before authorizing its further operation (e.g. data collecting) on the device.

In addition to the previously introduced cloud-based authentication mechanisms, there are some interesting systems in the literature, such as behavior-based authentication [26], single sign on [27], mobile trusted module [28] and anonymous authentication [29]. These systems, however, cannot address the above challenges as well.

3.2. Data secrecy protection

The confidentiality and integrity of mobile cloud data should be put at the top of priority list. Encryption technology seems to be an appropriate secure tool that can be used to protect the on-device (local) data and the outsourced data from being tempered and information extracted. Effective and efficient data protection and integrity check techniques can deliver sense of trust to mobile cloud users.

We first consider the case that mobile device users prefer to install a cryptographic system in their devices. The traditional cryptographic encryption is classified into two branches — symmetric encryption and asymmetric encryption. Advanced Encryption Standard (AES) [30] and Data Encryption Standard (DES) [31] are the standard examples of the former, while public key based encryption [32], identity-based encryption [33], attribute-based encryption [34] and functional encryption [35] are considered as the latter. The asymmetric encryption sometimes is generally referred to as public key encryption.

Being different from symmetric encryption (in which decryption and encryption are constructed by one key), public key encryption needs a pair of public and secret key. The public key is used for encryption, while the secret one is for decryption. The traditional public key encryption, like ElGamal [36] and RSA [37], only offers a one-to-one encryption and decryption mode. Namely, an encryption of a message can be only revealed by an encryption receiver with a valid secret key. Following by the well-study of ElGamal and RSA encryption, more flexible public key encryption systems have been proposed in the literature. Identity-based encryption [38–40] is proposed to allow users to share message in encrypted format under "identity" (such as email address). More general encryption (supporting one-to-many mode), like broadcast encryption (e.g. [41,42]), attribute-based/functional encryption (e.g. [43]), are used to encrypt message under pre-specified access policy. Compared to symmetric encryption, public key encryption technique provides flexibility in encryption and fine-grained data share ability, for example, an encryptor can encrypt a message for a group of users with knowledge of some public information (e.g., identities, attribute set). The advantages of public key encryption, however, yields relatively huge computation, communication and storage complexity as opposed to symmetric encryption. We take a look at the most well-known RSA encryption mechanism below. A mobile user, say Alice, may choose two distinct prime numbers p and q, compute $n = pq$ and $\phi(n) = (p-1)(q-1)$, and choose an integer e so that $gcd(e, \phi(n)) = 1$. Alice further chooses a d so that $d = e^{-1} mod\ \phi(n)$, publishes n and e as her unique public key, and keeps d secret as a secret key. Any system user knowing Alice's public key (n, e) that can encrypt an integer m $(0 \leq m < n$, $gcd(m, n) = 1)$ as $C = m^e\ mod\ n$ to Alice, such that Alice can use her secret key d to recover the m as $m = C^d\ mod\ n$, where $n = pq$, $1 < e < \phi(n)$, $gcd(e, \phi(n)) = 1$ and $d = e^{-1} mod\ \phi(n)$.

Even RSA, the most efficient public key encryption scheme, still cannot outperform symmetric encryption in the efficiency assessment w.r.t. power consumption, and encryption/decryption speed (the benchmark can be referred to Crypto++) (see Table 2 for the efficiency comparison. We note that the data in Table 2 is collected from Crypto++(https://www.cryptopp.com/) whereby AES is 128 bits, and RSA is 2048 bits. For RSA 2048-bit encryption, 0.16 Milliseconds/Operation is given. We assume that one operation roughly proceeds 1024-bit data. Therefore, the encryption

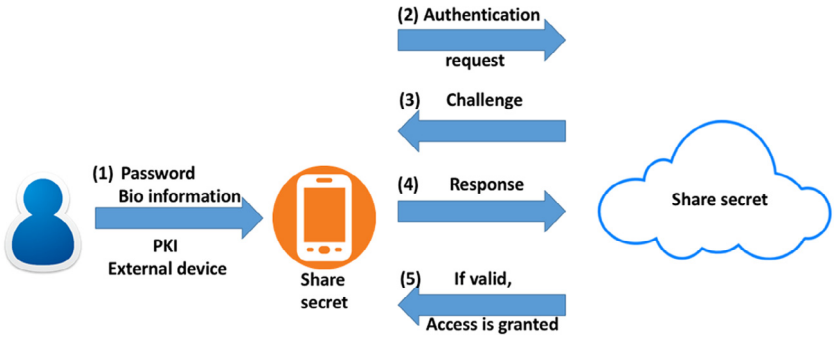


Fig. 2. Unidirectional mobile to cloud authentication structure.

Table 1
Comparison among different types of authentication.

Category	Security	Client to Cloud	Cloud to Client	Factor Update /Revoke	Authentication Delegation
Password	weak	✓	×	×	×
Possession	weak	✓	×	×	×
Biometric	weak	✓	×	×	×
Multi-factor	strong	✓	×	×	×

Table 2
Comparison among DES, AES and RSA.

	Key Size (bit)	Round	Running Time (MiB/Second)	Power Consumption	Hard/Software Implementation
DES	56	16	32	low	better in hardware
AES	128, 192, 256	10, 12, 14	139	low	fast
RSA	$\geq$ 1024	1	0.763 (Encryption) 0.020 (Decryption)	high	inefficient

complexity is around 7.63 MiB/s. Similarly, we have the decryption complexity of RSA is approximately 0.020 MiB/s.

If mobile users are only with single purpose — outsourcing their own data to mobile cloud, they may choose to employ symmetric encryption technology (e.g. AES) to encrypt the data before uploading to the cloud.

Symmetric encryption looks like a very promising solution to guarantee data security. Nevertheless, a direct and critical problem incurred by using symmetric encryption in mobile devices that is key management. Mobile users need to store encryption/decryption key locally, such that they can re-gain access to their data in the future. If the clients only upload a few files with small size (e.g. 1 MB) to cloud, key management problem may be ignored. But if they outsource a great amount of image, audio, and video data with huge size (e.g. 2 GB), the key management problem is extremely challenging as the devices may suffer from large-size key file storage issue. A naive solution to the problem is to encrypt the key file and next upload the encrypted file to mobile cloud. Nevertheless, again, the clients are still required to store some keys locally. Once the devices are intruded by attackers, the keys are compromised as well.

Symmetric and Asymmetric Method. To reduce local key storage cost, a mobile user may combine symmetric encryption with asymmetric one. Suppose *SYE* is a symmetric encryption with key generation algorithm *SYE.KeyGen*, encryption algorithm *SYE.Enc*, and decryption algorithm *SYE.Dec*; *PKE* is a public key encryption with key generation algorithm *PKE.KeyGen*, encryption algorithm *PKE.Enc*, and decryption algorithm *PKE.Dec*. The user may first generate a symmetric key *SYE.key* for a file f to be encrypted, runs $C = SYE.Enc(SYE.key, f)$ and further encrypts the key *SYE.key* as $V = PKE.Enc(PKE.pk, SYE.key)$, and finally uploads C and V to a mobile cloud, where public/secret key pair $(PKE.pk, PKE.sk) \leftarrow PKE.KeyGen$. After that, the user can reuse the same *PKE.pk* to encrypt all the symmetric keys, next upload the encryptions to the cloud. Here all ciphertexts and their corresponding encrypted keys are stored in the cloud. The user is only required to safely and locally store the *PKE.sk*. This hybrid method is more efficiency than managing a bunch of symmetric keys in local mobile devices.

Mobile Data Encryption Apps. Mobile encryption apps bring hope for lessening key management problem. Many mobile devices in various platforms (e.g. Apple iOS, Android, and Windows) enable users to encrypt personal data in a hard-cored way. Some data encryption apps (e.g. boxcryptor[14]) are also invented to allow users to encrypt mobile contents before uploading. The encryption, which is embedded in the platforms/apps, mostly depends on password/PIN mode, in which the password/PIN is used to encrypt encryption/decryption key. The encrypted key may be stored in remote cloud as well, depending on user preference. We note that even if a hard-cored security system is installed in a mobile to protect user data, a malicious attacker may be able to find a way to extract personal data from the mobile device [44].

Nonetheless, both hybrid and apps modes leave computation, communication and trust problems to us. No matter which apps or platforms we use, we have to encrypt data in local devices beforehand. This is a barrier to fully leverage the computational power of mobile cloud. Moreover, encrypting large volumes of files will occupy many local computation resource, increase battery consumption and meanwhile, large encrypted block might obstruct the mobile bandwidth. At last, a potential security risk pops up from a fact that we have to fully trust the apps/platforms we use. Once the trusted facilities are cracked/intruded by attackers, our data secrecy can be guaranteed no more.

Bypassing the usage of heavy cryptographic encryption tools, some academic research works (e.g. [45–47]) have been proposed to achieve high efficiency for mobile data encryption. For instance,

14 https://www.boxcryptor.com/en

an efficient image sharing system for mobile devices is introduced in [45], in which 90% of the image transmission cost is eliminated from the mobile user side (to a third party). However, the lightweight solutions are only the first step for mobile data outsourcing. Much like the aforementioned encryption approaches, these academic works fail to support remote data integrity check. Without integrity check, taking the image sharing system as an example, we cannot ensure that the outsourced images are "100%" identical to the original ones.

3.3. Data integrity check

The integrity check of outsourced (encrypted) data is desirable while data owner loses the physical control of the data. In traditional scenario, the check is fulfilled by simply using message digest technique (e.g. MD5 [48] and SHA [49]). Suppose there are a file f and its digest $D = H(f)$, a data owner is able to retrieve an encrypted file $Enc_{key}(f)$ from a mobile cloud, next recover f by using a decryption key key, and finally compare the value of $H(f)$ with the digest D (stored in mobile) to check if f is tempered, where H is a cryptographic hash function. Nevertheless, this technique requires data owner to possess a "copy" of the data (or its digest) which is stored locally. This brings storage hindrance for mobile device users.

Remote data auditing offers data integrity check with help of a trusted (third party) auditor even the data is outsourced to cloud. A remote data auditing system with data protection mainly works as in Fig. 3. The data owner can upload encrypted block data to the cloud server, while valid data readers are allowed to download and decrypt the data for further use. A trusted third party, called auditor, takes charge of the data integrity check. The auditor is shared with some secret information by the data owner in advance. In the checking phase, the cloud server first sends the specified data to the auditor and next gives a proof to the auditor's challenge. If the auditor accepts the proof, the data maintains its integrity.

A remote data auditing can usually be classified into one of the following models, namely provable data possession-based (PDP), proof of retrievability-based (POR) and proof of ownership-based (POW). The PDP method only focuses on preserving the integrity of outsourced data, so that data encryption technique may not be fully considered. Some existing PDP systems take no consider for data protection, e.g. [50,51], either are lack of data recovery support (i.e. the damaged data cannot be recovered) with linear complexity, e.g. [52,53] with $O(t)$ computation cost for client and the same complexity for communication, where t is the number of blocks to be changed/recovered; whilst the systems guarantee data recovery but leads to high (linearly) computation complexity on client side (e.g. [54]). The recent POR solution, [55], is a type of cryptographic proof of knowledge, protecting data secrecy and providing data recovery function. But its computation and storage overheads (with $O(tlog^2n)$ computation complexity on client side, and $O(t^2log^2n)$ for communication complexity) hinder its applicability to mobile applications, where n denotes the number of blocks of each file. Similarly, the most recent POW method designed in [56], single-instance data storage for removing data redundancy, still yields huge computation complexity - $O(t)$ for client computation and $O((m+t)n)$ for communication cost, where m is the number of symbols of a block. Besides, it fails to recover information from broken/damaged data.

On one hand, mobile device users are willing to offload computational complexity but also storage overhead to cloud. On the other hand, the users want to maintain the (periodically) data availability and integrity check for the "hand-off" data. From Table 3 and the previous paragraph description, we see that none of the existing systems cost-effectively achieves data protection, integrity check and data recovery in mobile setting.

3.4. Mobile cloud data search

Since being out of "physical control" of personal data, mobile device users may need some secure means to search and retrieve their data stored in mobile cloud. Searchable encryption mechanisms have been designed to guarantee data confidentiality and search privacy. Specifically, in a searchable encryption scheme, a data owner is allowed to upload an encrypted database and an encrypted search index structure to a cloud server (in the 1st phase), such that the server can locate the encrypted data by using so-called search token generated by the data owner (in the 2nd phase), see Fig. 4. Searchable encryption mechanism is generally based on client–server mode. A data writer is allowed to encrypt and upload the data to cloud server, while a data reader is able to generate search trapdoor for the server, so that the server can search the related encrypted file(s). There are total four searchable encryption architectures, namely single writer/single reader, multi-writer/single reader, single writer/multi-reader, multi-writer/multi-reader.

Symmetric searchable encryption (SSE) [57–60] and public key based searchable encryption [61,62] are two classic types of searchable encryption. SSE is usually leveraged in practice as its efficiency is much better than that of public key based systems. This is because SSE only makes use of lightweight cryptographic tools as building blocks, such as pseudorandom function, pseudorandom permutation, and hash function. Moreover, SSE focuses more on the optimization of encrypted search index structure compared to the design of public key base systems. On the contrary, public key based searchable encryption is built on top of public key encryption technique, e.g., identity/attribute-based encryption. The main reason of the low search efficiency of public key system is that a system usually takes pairings computation as a matching test. We here take [61] as an example. In [61], $T_W = H_1(W)^{\alpha}$ is seen as a search token/trapdoor, and the keyword W is attached to ciphertext components $A = g^r$, $B = H_2(e(H_1(W), h^r))$, where α is the secret key of data owner, $h = g^{\alpha}$, r is a random element, H_1 and H_2 are cryptographic hash functions. To successfully match a given token with a ciphertext, the server needs to perform a pairing calculation, $H_2(e(T_W, A)) = B$. We note that public key based searchable encryption has its own advantages that are not implied in SSE, although it cannot outperform SSE in terms of efficiency. For example, public key systems provide integrity check for any third party without knowledge of secret key information. The search index structure can also be effectively checked/verified even the encrypted structure is stored remotely in cloud.

Consider in practical use that a mobile device user may only choose to encrypt his/her private data before uploading to mobile cloud, and further to either search the data on his/her own or delegate the search to other parties, i.e. single writer/single-or-multi-reader SE mode. The mobile user may use an SSE system due to its high efficiency in the mode. A recent SSE system, for large scale database, is proposed in [58]. The primitive idea of the system design is that a user symmetrically encrypts each file with a keyword w as $d \leftarrow Enc(K_2, I_i)$ with the key $K_2 \leftarrow F(K, 2 \parallel w)$, and stores d into an array A ($|A| = T$), where F is a pseudorandom function, and K is its seed. The user further partitions A into b blocks ($T' \leftarrow \lceil T/b \rceil$) and computes the new indices as $l \leftarrow F(K_1, c)$ and $d' \leftarrow Enc(K_2, J_c)$, where $c \in [0, \ldots, T']$ and J_c is the cth block of A, and $K_1 \leftarrow F(K, 1 \parallel w)$. The tuples (l, d') are stored in a list γ. For data search, given K_1 and K_2, the server first locates $d \leftarrow Get(\gamma, F(K_1, c))$ from γ, recovers $(i_1, \ldots, i_b) \leftarrow Dec(K_2, d)$, and finally computes $l_i \leftarrow Dec(K_2, A[i_j])$. The system is efficient as only pseudorandom function and symmetric encryption are used in the construction. However, users have to undertake high computation complexity for encrypting "the whole" database and its search index structure (in advance), but also to spend large communication

Table 3
Summarization for Data Protection and Integrity Check.

Systems	Data Protection	Integrity Check	High Computation Complexity for Client	Data Recovery
DES	✓	×	×	×
AES	✓	×	×	×
RSA	✓	×	×	×
PDP	✓[a]	✓	✓	×[b]
POR	✓	✓	✓	×[c]
POW	✓	✓	✓	×

[a] Some PDP cannot fully provide data protection
[b] Most of PDP fail to provide data recovery
[c] Most of POR supports data recovery

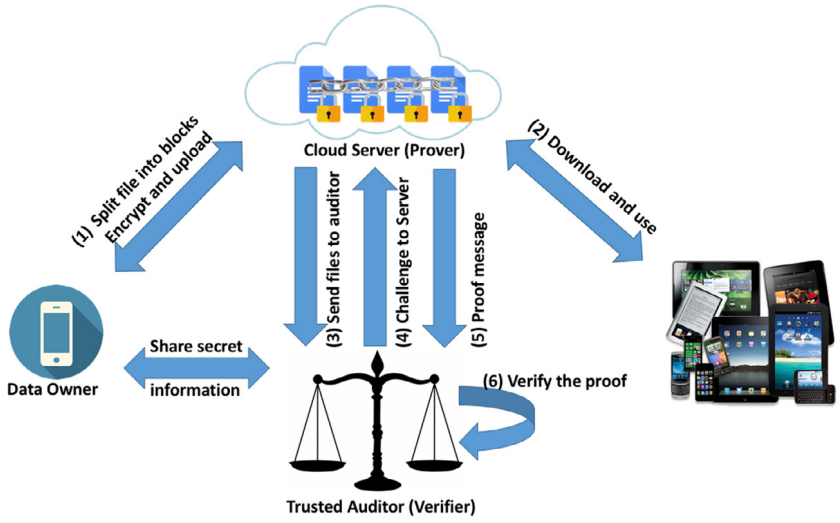


Fig. 3. Remote data auditing system with data protection.

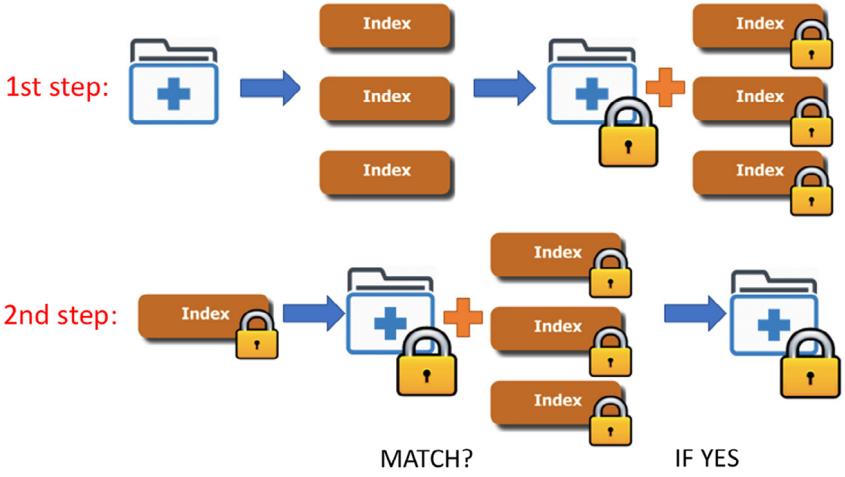


Fig. 4. Secure searchable encryption framework.

cost in transferring the encrypted database and the index structure (to cloud). This can be evidenced from the above details that a user must build up a search index structure, and next compute each related file's encryption and pseudorandom value. Furthermore, the symmetric encryption and pseudorandom computation for l as well as encrypted files are linearly in the product of number of keyword and the related files. If there is a great amount of files in the database, say 100 GB, a mobile user has to take a long time to deal with the "encryption" of index structure and the database.

To offload the above burden to a third party, we have to assume that the party is fully trusted as a secret information of data search belonging to data owner will be shared with that party. This trust assumption does not scale well in practice, since once the party is compromised by malicious attackers, the attackers can fully obtain the search ability. More recently, Li et al. [63] introduced a traffic and energy saving encrypted search system to remove the fully trust assumption and furthermore to protect data privacy. The system, unfortunately, cannot support expressive

search query, such as range, boolean and more complex formula query. We note that an expressive searchable encryption scheme based on regular language [64] has recently been proposed in the literature. Nonetheless, the scheme suffers from low efficiency that may hinder its implementation from real-world applications.

All the aforementioned systems only provide "plain" text/ symbol/ formula based search for mobile device users. In real-world applications, audio/video-based, and even bio-based search pattens are desirable. Designing privacy-preserving search with workload offloading (to cloud) without loss of search expressiveness is a challenging and unsolved problem. Besides, a data owner may would love to delegate his search rights to others, and meanwhile, the owner prefers to control the search delegation by revocation strategy.

3.5. Secure data share

To securely share a file with others, a mobile device user may use traditional encryption (e.g. attribute-based encryption [43, 65]). But the traditional encryption requires the user to be always on-line, and to consume considerable computation resource (to perform an encryption of sharing data), communication cost and battery to fulfill a simple data sharing. If the encrypted file is stored in cloud, the user has to download the file in local before proceeding to re-encryption. A naive solution would be that the data owner shares the secret key with a proxy cloud, so that the cloud may "decrypt and then re-encrypt" the corresponding ciphertext to other specified users. However, the solution depends on the assumption that the proxy is fully trusted but also being not curious on the encrypted data (as well as the corresponding secret key). It is clear that the assumption cannot firmly hold in practice.

Proxy re-encryption (PRE) has been invented to tackle the above problem in an effective and efficient way in the sense that a user only generates a special key (other than a ciphertext, as the golden coin in Fig. 5) for cloud server, such that the server can convert the ciphertexts of the user into those intended for others. In the figure, Alice is known as a delegator, while Bob is called a delegatee; the golden coin is referred to as a re-encryption key for the ciphertext conversion.

The idea of decryption rights delegation is initially introduced by Mambo and Okamoto [66]. Following the delegation concept, Blaze, Bleumer and Strauss [67] defined the seminal notion and a concrete construction of PRE. PRE can be classified into four types, namely unidirectional, bidirectional, single-hop and multi-hop PRE. PRE has been designed for various encryption scenarios in the literature[15] . For instance, there are traditional proxy re-encryption [68–70] (based on traditional public key encryption technique), identity-based proxy re-encryption [5,71–73], broadcast re-encryption [74,75], attribute-based re-encryption [76–79] and functional proxy re-encryption [80,81].

The premise of PRE relies on the design of re-encryption algorithm that guarantees a proxy server to run a "partial decryption" for an original ciphertext of a user and next create a "full encryption" for a delegatee, so that the valid delegatee can recover the message by its decryption key and meanwhile, the proxy knows nothing about the underlying message. To achieve secure re-encryption, the construction of a re-encryption key is somewhat subtle. For instance, given a ciphertext tuple $(Z_1 = g^{xr}, Z_2 = e(g,g)^r \cdot m)$, a user A may construct a re-encryption key $g^{y/x}$ for the proxy server, such that the server can compute $Z_3 = e(Z_1, g^{y/x})$ for a delegatee Y who can later recover m by computing $Z_2/Z_3^{1/y}$, where (g^x, x) and (g^y, y) are public/secret key pairs for X and Y.

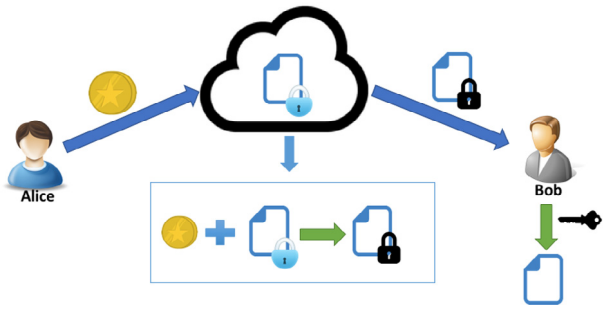


Fig. 5. Secure encrypted cloud-based data share — proxy re-encryption.

Nevertheless, the simple usage of PRE yields a potential security risk in ciphertext conversion that no one knows if the conversion is correct. A direct solution is introduced in [82] in which an encryption receiver with appropriate decryption rights can check the validity of conversion. This post-check mode, actually, does not scale well in real world, as it is too late to detect the errors – after the encrypted data being downloaded, and meanwhile, only a valid decryptor can tell the errors upon gaining access to the encryption. A practical and publicly validity check method – before downloading data, is necessary here. There is also an inevitable disadvantage of PRE system while the system is implemented in the mobile client–cloud server model. A mobile user, say Alice, cannot prevent a dishonest cloud server from re-delegating her decryption rights to other "unspecified" users, if the server colludes with any delegatee who is already granted Alice's decryption rights. For example, the server may work with a specified delegatee Bob, and further re-delegate Alice's decryption rights to a malicious user Carol. The re-delegation can be fulfilled even without any permission from Alice. This issue seems to be incurred by the subtle PRE construction. It is desirable to design a detecting mechanism to identify any misbehaves yielded by delegatee and proxy server.

PRE also easily suffers from an inside-domain-conversion limitation that is a re-encryption key can only be used to handle the conversion among the "same-type" ciphertexts[16] . For example, an identity-based encryption can be turned into another ciphertext with the same identity-based format, while an attribute-based ciphertext corresponds to an "attribute-based" conversion. Furthermore, for more fine-grained encryption, e.g., functional encryption, the cost of re-encryption key generation (as well as encryption/decryption) is extremely heavy for mobile user, as it usually is linearly in either the size of access policy or the size of attribute set. Efficient construction for re-encryption key is necessary in the context of functional encryption.

3.6. Outsourced data computation

Mobile devices were limited to very restricted computational ability and storage space around a decade ago. It is undeniable that the recent advanced mobile software and hardware technologies give birth to a new generation of mobile devices with stronger computational power, larger storage room and longer battery life. However, local data processing/computing and maintenance, in particular those related to large scale database, will definitely bring headache to mobile users. Thanks to the prevalence of cloud, mobile users are offered an option to outsourced their data to cloud, so that cloud can fulfill heavy computing tasks on behalf

[15] We note that nearly all PRE systems are designed on top of public key encryption technique.

[16] We note that [83] introduces an approach to convert traditional public key encryption to identity-based one; while [84] proposes a scheme to transform attribute-based encryption to identity-based.

of the users. Without loss of data secrecy, many cloud users may choose to leverage encryption technique to "mask" their data before outsourcing. There is a few encryption technologies that can be used to guarantee secure encrypted data computation. Below we focus on homomorphic encryption. We note that some other mechanisms, like secure two/multi-party computation, are also applicable to outsourced data computation applications. The reason we only mention homomorphic encryption is that the homomorphic technique is seen as the underpinning for "high-level" secure (computation) constructions[17] .

Performing computation on encrypted data is the initial intention of designing homomorphic encryption. Rivest et al. proposes the seminal concept of homomorphic encryption in [86]. ElGamal [36], Goldwasser–Micali [87], Paillier Encryption [88] are well-known asymmetric homomorphic encryption systems. Following by the aforementioned prominent systems, some variants of homomorphic encryption have been proposed in the literature, for example, [89,90] can be regarded as the generalization of [87], and [91] is an adaptation of [88] in elliptic curves. However, the previously introduced systems can only support either addition or multiplication computing tasks. A breakthrough was put forward in 2009. Craig Gentry [92] revisits the homomorphic encryption to introduce the first plausible fully homomorphic encryption scheme. The fully homomorphic encryption supports addition and multiplication computing at the same time. Compared to the "somewhat" homomorphic encryption (those only supporting either addition or multiplication), the "fully" feature is more practical. Since the introduction of Gentry's seminal work, homomorphic encryption schemes have been mainly developed from integer-based, lattice-based, and (ring) learning-with-errors based encryption.

Homomorphic encryption technique is an effective approach for encrypted data computation whereby an untrusted party can compute the encrypted data in a "blind" way but outputting valid and correct "encrypted" result. The party here knows nothing about the result but also underlying encrypted input. Homomorphic encryption can support ciphertext either +or $\cdot$ operation, or both of the operations. For multiplication, taking ElGamal encryption as an example, we have $Enc(m_1) \otimes Enc(m_2) = Enc(m_1 \cdot m_2)$; $Enc(m_1) \otimes Enc(m_2) = (g^{r_1}, m_1 h^{r_1})(g^{r_2}, m_2 h^{r_2}) = (g^{r_1+r_2}, (m_1 \cdot m_2) h^{r_1+r_2})$. From the design of Paillier encryption, we can see its addition homomorphic feature. We have $Enc(m_1) \otimes Enc(m_2) = Enc(m_1 + m_2)$; $Enc(m_1) \oplus Enc(m_2) = (g^{m_1} r_1^x)(g^{m_2} r_2^x) = g^{m_1+m_2}(r_1 r_2)^x$, where x is the modular, r_1, r_2 are random seeds, and m_1, m_2 are messages. Whereas the fully homomorphic encryption, the more desirable one in practice, can provide both types of calculation - $Enc(m_1) \cdot Enc(m_2) = Enc(m_1 + m_2)$ and $Enc(m_1) \cdot Enc(m_2) = Enc(m_1 \cdot m_2)$. An advantage of using homomorphic encryption in mobile cloud is that the computation cost can be offloaded from users to cloud. This is so because the homomorphic operations can be taken care by cloud, and mobile users may only need to prepare data encryption on their mobile devices. We have to note that current fully homomorphic encryption is not ready for real-time applications, for instance, the public/secret key generation phase of [93] takes hours to set up, the corresponding public key size could be up to 2 GB, and a homomorphic evaluation in AES circuit needs more than 30 h to be done [94].

Although there exist some improved versions of homomorphic encryption over efficiency (in terms of running time and memory usage), e.g., [95],[18] the homomorphic technology still does not scale well while being used in the context of mobile cloud. We note secure multi-party computation (MPC) systems can support cloud-based encrypted data computing in sense that a server intakes two respective encrypted values as input and outputs a "masked" result. However, those systems suffer from similar limitations as the homomorphic encryption does as follows. First of all, no current systems enable the encryption of arbitrary values in $\mathbb{R}$, i.e. real number. Although Chinese Remainder Theorem can be used to increase message space of system to support large integer, it seems there is a long way for homomorphic encryption to support real number calculation. In addition to huge ciphertext size/memory cost for just a simple homomorphic evaluation, there is no homomorphic system providing a native division operation. Mobile users have to download the corresponding encrypted data from cloud to decrypt-then-calculate the division on their owns. Moreover, if the homomorphic computation outputs a "long" encrypted result, such as a set of "masked" genomic-related data, the devices will suffer from huge computation and communication cost for download-then-decrypt operation. Last but not least, the existing homomorphic encryption systems fail to support search functionality, so that it is hard for the cloud server to tell which encrypted data (stored in the cloud) should be intaken for calculation.

3.7. Malicious behaviors traceability

An Internet user may encounter with various types of malicious behaviors launched by adversaries (such as network attackers), while connecting his/her devices to an open network. The behaviors are usually full of hostility but also with special purpose. For example, an on-line banking hacker may try to deceive a bank user to reveal the login password by phishing. Due to space limit, we cannot explore all ranges of malicious behaviors (targeting to mobile cloud user) in this paper but only consider the following type of "misbehaviors". We refer to this type of malicious behaviors as decryption rights leak.

The decryption rights leak commonly appears in daily Internet life. For instance, a valid monthly pay apple TV subscriber may choose to re-sell his/her subscribed channel to others, so that the person, who have not even registered to apple TV, could be able to gain access to the contents of the channel. This malicious behavior could happen to many TV broadcast services. Similarly, an eLearning registered user is able to unrestrictedly share the on-line learning contents with unlimited amount of his/her friends, by only sharing the subscribed key. Consider another example on eHealth. After a patient encrypts and uploads his/her medical/health record to an electronic health care server, an access granted doctor (e.g. a physician) might have chance to disclose the record to a third party (e.g. insurance company) for illegal extra income, for example, the doctor may obtain benefit from persuading the patient to buy medical service from the party.

In addition, key generating authority/infrastructure, sometimes, cannot be fully trusted as well. This is so because the authority may collude with network attackers to leak specified system users' decryption rights to reveal the corresponding secret information. For example, an administrator who takes fully control of the eHealth platform can allow insurance companies or medicine agents to gain access to patient's health records for business marketing.

Technically, it is difficult for us to totally eliminate the deliberate decryption rights leak, since a system insider (i.e. system user as well as system administrator) can "freely" deliver/share secret key with other entities. There is possibility to perform some post-prevention for the above case, although the behavior cannot be pre-prevented. For example, we need to identify the entity who leaks decryption rights, further revoke the dishonest user, and meanwhile, update the ciphertext(s) related to the leaked secret key (see Fig. 6, in which we denote a private key generator

[17] We refer readers to [85] for more introduction on security tools for mobile cloud computing.

[18] The paper limits the computation to small number of AND gates with shallow depth, and the multiplications are in GF(2).

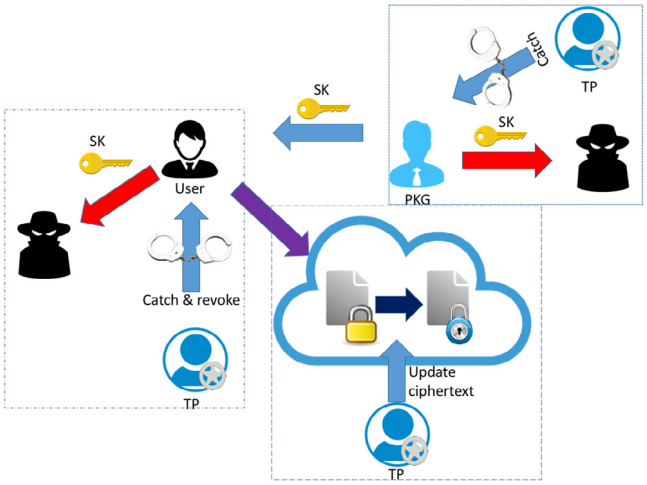


Fig. 6. Tracing misbehave user and private key generator.

authority as PKG, secret key as SK, an auditing trusted party as TP). Two decades ago, Chor, Fiat and Naor [96] proposed three pieces of seminal cryptographic schemes to track down dishonest user(s) inside a secure system. However, their schemes require huge amount of storage space for decryption and encryption, in which the most efficient scheme needs to prepare $O(klog(n/p))$ and $O(k^2log(n/p))$[19] decryption and encryption key (per message block) for each system user, respectively. After that some more efficient trace secure systems have been introduced in the literature, such as [97,98]. We note that traceability is also explored to other cryptographic systems, such as signature [99]. But here we mainly concentrate on data secrecy scenario.

Boneh and other cryptographers [100,101] explores the traceability to the public key encryption system. Following by the inspiration of the previous works, some scholars have extended traceability to more general encryption level. Libert et al. [102] designs a traceable group encryption system, while Au et al. [103] invents an identity-based encryption dealing with dishonest private key generator. There have been traceable broadcast encryption [104], attribute-based encryption [105,106] and predicate encryption [107] systems.

Although the traceability can be extended to general encryption, e.g. attribute-based encryption, the low encryption/decryption efficiency issue is still unsolved on user/client side. In other words, mobile device users are required to bear heavy encryption/decryption burden while uploading (resp. downloading) their data to (resp. from) cloud. Moreover, this traceability (as well as revocability) should be explored to more general scenarios, such as tracing misbehaved cloud server. An anonymous but malicious service provider should be caught and punished under law enforcement.

In addition to aforementioned limitations, the single ability providing in searchable encryption (searchability), homomorphic encryption/MPC (secure computation) and PRE (secure data share) cannot fully satisfy the multiple functionalities need of mobile users (see Table 4). A naive "all-in-one" solution is to trivially combine a searchable encryption, a homomorphic encryption/MPC and a PRE into one system. Nevertheless, it is unknown that if the building blocks are compatible with each other and furthermore, if the combination is effective and secure.

[19] k is the number of dishonest collaborator, n is the number of system user, and p is some positive probability.

4. Some possible countermeasures for the challenges

In the previous section, we summarize four major challenges for mobile cloud users — mobile user and mobile cloud bidirectional authentication, mobile data encryption and integrity check, and mobile data search, share and computation, and malicious behaviors traceability. In this section, we generally propose some possible solutions to the challenges. For bidirectional authentication, we may employ the aid of a trusted third party who authenticates both mobile user and mobile cloud service provider. The trusted party can reuse some existing mobile user authentication system to check the identity of user; while the same technique can be employed into the authentication for service provider.

To guarantee encryption but also data integrity, we may try to design new POR schemes which ensure data protection, integrity check and data recovery at the same time. However, the only drawback of POR is its high computation and communication complexity. Using a trusted party to handle most of heavy computational cost might be one of the effective ways to address the efficiency problem. Much like on/off-line encryption technique, (e.g., on-line/off-line attribute-based encryption), the on-line lightweight part is taken by mobile users, while the off-line heavy cost computation is assigned to the trusted party.

Designing an "all-round" system with secure data search, share and computation is extremely challenging. To the best of our knowledge, none of the existing cryptographic systems can perfectly achieve the goal. Recently, two interesting works [108,109] have been proposed to provide search and share functionalities. [108] and [109] are built on top of attribute-based and identity-based encryption, respectively. Below we show that the system [108] may be extended to offer somewhat homomorphic computation. We convert the ciphertext element A to be $m_1 \cdot e(g, \hat{g})^{\alpha s_1}$, and keep other elements unchanged, i.e. $B = g^{s_1}$, $\{C_x = h_x^{s_1}\}_{x \in S}$, $D = e(t^{H_3(KW)}z, \hat{g})^{\hat{\alpha} s_1}$, and $E_1 = f_1^{s_1}$. This is a ciphertext for message m_1. Note we only consider the chosen plaintext security here, so that we ignore the element E_2 in the original scheme in [108]. Suppose there is another ciphertext for message m_2 - $(m_2 \cdot e(g, \hat{g})^{\alpha s_2}, g^{s_2}, \{h_x^{s_2}\}_{x \in S}, e(t^{H_3(KW)}z, \hat{g})^{\hat{\alpha} s_2}, f_1^{s_2})$.

We further assume that the ciphertexts share the same attribute set S and keyword KW. We have a somewhat homomorphic computation as follows - $\bar{A} = (m_1 \cdot m_2) \cdot e(g, \hat{g})^{\alpha(s_1+s_2)}$, $\bar{B} = g^{s_1+s_2}$, $\{\bar{C}_x = h_x^{s_1+s_2}\}_{x \in S}$, $\bar{D} = e(t^{H_3(KW)}z, \hat{g})^{\hat{\alpha}(s_1+s_2)}$, and $\bar{E}_1 = f_1^{s_1+s_2}$, such that the corresponding decryption yields $m_1 \cdot m_2$. The above extension does not affect the re-encryption, search and decryption algorithms, since the above "computation" is an exact output of the encryption algorithm in [108].

However, the resulting system suffers from linearly computation and communication cost that is unbearable for mobile device users. Specifically, one encryption operation requires $O(|S|)$ and $O(1)$ exponents in $\mathbb{G}_1$ and $\mathbb{G}_2$, a search token generation needs $O(l^2)$ exponents in $\mathbb{G}_1$; while an encryption and a search token occupy $O(|S|)|\mathbb{G}|+O(1)|\lambda|$ and $O(l^2)|\mathbb{G}|$ bandwidth, respectively. One of the possible methods to reduce the complexity is to reuse the trusted party idea again in such a way that all linearly cost ($O(|S|)$ and $O(l^2)$) will be lifted to the party, and the constant cost ($O(1)$) is allocated to mobile users.

In contrast to [108], [109] is able to be extended to achieve efficient multiplicative homomorphic property. We below concentrate the extension on the algorithm *Enc* in [109]. Suppose there is a ciphertext of m_1 under ID and w, $C_1 = m_1 \cdot e(g_2, \hat{g}_1)^{t_1}$, $C_2 = g^{t_1}$, $C_3 = (h_1^{ID} g_3)^{t_1}$, $C_4 = (g_5 g_4^{-w})^{t_1}$, $C_5 = e(g_4, \hat{g}_4)^{t_1}$, $C_6 = H_1(e(h_2, \hat{g}_4)^{t_1})$, $C_7 = K^{t_1}$. We here remove the hash computation for the element

Table 4
Summarization.

Systems	Search	Share	Computation	Traceability
Searchable Encryption	✓	×	×	×
PRE	×	✓	×	×
Homomorphic Encryption/MPC	×	×	✓	×
Traceable Encryption	×	×	×	✓

C_6, so that C_6 is equal to $e(h_2, \hat{g}_4)^{t_1}$. Note since the hash computation part is removed, the algorithm *Search* is correspondingly revised to check $e(C_4, tk_2)C_5^{tk_1} \stackrel{?}{=} C_6$. We further assume there is another ciphertext of m_2, but *with a restriction* that the ciphertext is under the same *ID* and w (as well as the same keyword update status in $List_{up}$), $m_2 \cdot e(g_2, \hat{g}_1)^{t_2}, g^{t_2}, (h_1^{ID} g_3)^{t_2}, (g_5 g_4^{-w})^{t_2}, e(g_4, \hat{g}_4)^{t_2}$, $e(h_2, \hat{g}_4)^{t_2}, K^{t_2}$. Accordingly, we have a new ciphertext of $m_1 \cdot m_2$ by multiply the two ciphertexts, $m_1 \cdot m_2 \cdot e(g_2, \hat{g}_1)^{t_3}, g^{t_3}, (h_1^{ID} g_3)^{t_3}$, $(g_5 g_4^{-w})^{t_3}$, $e(g_4, \hat{g}_4)^{t_3}$, $e(h_2, \hat{g}_4)^{t_3}$, K^{t_3}, where $t_3 = t_1 + t_2$. The resulting ciphertext is the exact output of $Enc(ID, w, m_1 \cdot m_2)$ so that it does not jeopardize the further data search, share, keyword update and decryption functionalities. It can be seen from the above description that the homomorphic addition only takes constant cost in computation and communication. That may scale well in real-world mobile applications.

5. Conclusions

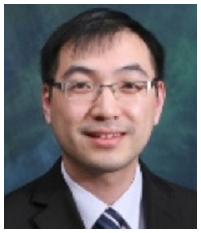
In investigating the chances and challenges of mobile cloud, our goal is to inspire academic and industrial communities to tackle all the problems involved. We also would like to light in the hopes that mobile cloud service providers, mobile device users and local authorities will be more conscious of the challenges and embrace the opportunities to work together to create a brighter future for the mobile cloud applications.

References

[1] S. Rudolph, Mobile apps usage –statistics and trend.
[2] D. Chaffey, Mobile marketing statistics compilation.
[3] R. Lu, H. Zhu, X. Liu, J.K. Liu, J. Shao, Toward efficient and privacy-preserving computing in big data era, IEEE Netw. 28 (4) (2014) 46–50.
[4] J. Baek, Q.H. Vu, J.K. Liu, X. Huang, Y. Xiang, A secure cloud computing based framework for big data information management of smart grid, IEEE Trans. Cloud Comput. 3 (2) (2015) 233–244.
[5] K. Liang, W. Susilo, J.K. Liu, Privacy-preserving ciphertext multi-sharing control for big data storage, IEEE Trans. Inf. Forensics Secur. 10 (8) (2015) 1578–1589.
[6] J.K. Liu, M.H. Au, W. Susilo, K. Liang, R. Lu, B. Srinivasan, Secure sharing and searching for real-time video data in mobile cloud, IEEE Netw. 29 (2) (2015) 46–50.
[7] X. Yang, X. Huang, J.K. Liu, Efficient handover authentication with user anonymity and untraceability for mobile cloud computing, Future Gen. Comp. Syst. 62 (2016) 190–195.
[8] S.S.M. Chow, U. Hengartner, J.K. Liu, K. Ren, Spec. Issue Secur. Priv. Mob. Cloud. 28 (2016) 100–101.
[9] B. Technology, FBI investigates 'cloud' celebrity picture leaks.
[10] W. Ashford, Barclays bank leaks thousands of customer records.
[11] NHS cyber-attack: Amber rudd says lessons must be learnt.
[12] I. Jeun, M. Kim, D. Won, Enhanced password-based user authentication using smart phone, in: GPC '12, in: LNCS, Vol. 7296, Springer, 2012, pp. 350–360..
[13] T. Acar, M. Belenkiy, A. Küpçü, Single password authentication, Comput. Netw. 57 (13) (2013) 2597–2614.
[14] X. Yi, F. Hao, L. Chen, J.K. Liu, Practical threshold password-authenticated secret sharing protocol, in: ESORICS '15, Part I, in: LNCS, Vol. 9326, Springer, 2015, pp. 347–365.
[15] W.Z. Khan, M.Y. Aalsalem, Y. Xiang, A graphical password based system for small mobile devices, CoRR abs/11103844.
[16] A. Yassin, H. Jin, A. Ibrahim, W. Qiang, D. Zou, Cloud Authentication Based on Anonymous One-Time Password, in: Ubiquitous Information Technologies and Applications, in: LNCS, Vol. 214, 2013.
[17] D. Zissis, D. Lekkas, Addressing cloud computing security issues, Future Gen. Comp. Syst. 28 (3) (2012) 583–592.
[18] M. Karnan, M. Akila, N. Krishnaraj, Biometric personal authentication using keystroke dynamics: A review, Appl. Soft Comput. 11 (2) (2011) 1565–1573.
[19] T. Bhattasali, K. Saeed, N. Chaki, R. Chaki, A survey of security and privacy issues for biometrics based remote authentication in cloud, in: CISIM '14, in: LNCS, Vol. 8838, Springer, 2014, pp. 112–121.
[20] Y. Yang, H. Lu, J.K. Liu, J. Weng, Y. Zhang, J. Zhou, Credential wrapping: From anonymous password authentication to anonymous biometric authentication, in: AsiaCCS '16, ACM, 2016, pp. 141–151.
[21] K. Xi, T. Ahmad, F. Han, J. Hu, A fingerprint based bio-cryptographic security protocol designed for client/server authentication in mobile computing environment, Secur. Commun. Netw. 4 (5) (2011) 487–499.
[22] D. Pointcheval, S. Zimmer, Multi-factor authenticated key exchange, in: ACNS '08, in: LNCS, Vol. 5037, 2008, pp. 277–295.
[23] Y. Shah, V. Choyi, L. Subramanian, Multi-factor authentication as a service, in: IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, MobileCloud 2015, pp. 144–150.
[24] J.K. Liu, K. Liang, W. Susilo, J. Liu, Y. Xiang, Two-factor data security protection mechanism for cloud storage system, IEEE Trans. Comput. 65 (6) (2016) 1992–2004.
[25] J.K. Liu, M.H. Au, X. Huang, R. Lu, J. Li, Fine-grained two-factor access control for web-based cloud computing services, IEEE Trans. Inf. Forensics Secur. 11 (3) (2016) 484–497.
[26] R. Chow, M. Jakobsson, R. Masuoka, J. Molina, Y. Niu, E. Shi, Z. Song, Authentication in the clouds: a framework and its application to mobile users, in: CCSW '10, ACM, 2010, pp. 1–6.
[27] J. Chen, G. Wu, L. Shen, Z. Ji, Differentiated security levels for personal identifiable information in identity management system, Expert Syst. Appl. 38 (11) (2011) 14156–14162.
[28] M. Kim, H. Ju, Y. Kim, J. Park, Y. Park, Design and implementation of mobile trusted module for trusted mobile computing, IEEE Trans. Consum. Electron. 56 (1) (2010) 134–140.
[29] X. Huang, J.K. Liu, S. Tang, Y. Xiang, K. Liang, L. Xu, J. Zhou, Cost-effective authentic and anonymous data sharing with forward security, IEEE Trans. Comput. 64 (4) (2015) 971–983. http://dx.doi.org/10.1109/TC.2014.2315619.
[30] Anon, Announcing the ADVANCED ENCRYPTION STANDARD (AES), in: Federal Information Processing Standards Publication 197, United States National Institute of Standards and Technology, NIST, November 26, 2001 [Retrieved October 2, 2012].
[31] C. Paar, J. Pelzl, The data encryption standard (DES) and alternatives, in: Understanding Cryptography, A Textbook for Students and Practitioners, Springer, Germany, 2000.
[32] T.V.X. Phuong, G. Yang, W. Susilo, K. Liang, Edit distance based encryption and its application, in: ACISP '16, in: LNCS, Vol. 9723, Springer, 2016, pp. 103–119.
[33] D. Boneh, X. Boyen, Efficient selective-ID secure identity based encryption without random oracles, in: EUROCRYPT '04, in: LNCS, Vol. 3027, Springer, 2004, pp. 223–238.
[34] V. Goyal, O. Pandey, A. Sahai, B. Waters, Attribute-based encryption for fine-grained access control of encrypted data, in: ACM CCS '06, pp. 89–98.
[35] B. Waters, 2012, Functional encryption for regular languages, in: CRYPTO, pp. 218–235.
[36] T.E. Gamal, A public key cryptosystem and a signature scheme based on discrete logarithms, IEEE Trans. Inf. Theory 31 (4) (1985) 469–472.
[37] R.L. Rivest, A. Shamir, L.M. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM 21 (2) (1978) 120–126.
[38] D. Boneh, M.K. Franklin, Identity-based encryption from the weil pairing, in: CRYPTO '01, in: LNCS, Vol. 2139, Springer, 2001, pp. 213–229.
[39] B. Waters, Efficient identity-based encryption without random oracles, in: EUROCRYPT '05, in: LNCS, Vol. 3494, Springer, 2005, pp. 114–127.
[40] C. Gentry, Practical identity-based encryption without random oracles, in: EUROCRYPT '06, in: LNCS, Vol. 4004, Springer, 2006, pp. 445–464.
[41] K. He, J. Weng, J. Liu, J.K. Liu, W. Liu, R.H. Deng, Anonymous identity-based broadcast encryption with chosen-ciphertext security, in: AsiaCCS '16, ACM, 2016, pp. 247–255.
[42] C. Gritti, W. Susilo, T. Plantard, K. Liang, D.S. Wong, Broadcast encryption with dealership, Int. J. Inf. Sec. 15 (3) (2016) 271–283.
[43] S. Wang, K. Liang, J.K. Liu, J. Chen, J. Yu, W. Xie, Attribute-based data sharing scheme revisited in cloud computing, IEEE Trans. Inf. Forensics Secur. 11 (8) (2016) 1661–1673.

[44] Q. Do, B. Martini, K.R. Choo, Exfiltrating data from android devices, Comput. Secur. 48 (2015) 74–91.
[45] H. Cui, X. Yuan, C. Wang, Harnessing encrypted data in cloud for secure and efficient image sharing from mobile devices, in: INFOCOM '15, IEEE, pp. 2659–2667.
[46] N. Park, Y. Song, AONT encryption based application data management in mobile RFID environment, in: ICCCI '10, in: LNCS, Vol. 6422, Springer, 2010, pp. 142–152.
[47] C. Luo, A. Fylakis, J. Partala, S. Klakegg, J. Goncalves, K. Liang, T. Seppänen, V. Kostakos, 2016, A data hiding approach for sensitive smartphone data, in: ACM UbiComp '16, pp. 557–468.
[48] T.A. Berson, Differential cryptanalysis mod 2^32 with applications to MD5, in: EUROCRYPT '92, in: LNCS, Vol. 658, Springer, 1992, pp. 71–80.
[49] B. Schneier, Cryptanalysis of md5 and sha: Time for a new standard, https://www.schneier.com/essays/archives/2004/08/cryptanalysis_of_md5html.
[50] C.C. Erway, A. Küpçü, C. Papamanthou, R. Tamassia, Dynamic provable data possession, ACM Trans. Inf. Syst. Secur. 17 (4) (2015) 15.
[51] G. Ateniese, R.C. Burns, R. Curtmola, J. Herring, L. Kissner, Z.N.J. Peterson, D.X. Song, Provable data possession at untrusted stores, in: ACM CCS '07, pp. 598–609.
[52] K. Yang, X. Jia, An efficient and secure dynamic auditing protocol for data storage in cloud computing, IEEE Trans. Parallel Distrib. Syst. 24 (9) (2013) 1717–1726.
[53] C. Wang, S.S.M. Chow, Q. Wang, K. Ren, W. Lou, Privacy-preserving public auditing for secure cloud storage, IEEE Trans. Comput. 62 (2) (2013) 362–375.
[54] G. Ateniese, R.C. Burns, R. Curtmola, J. Herring, O. Khan, L. Kissner, Z.N.J. Peterson, D. Song, Remote data checking using provable data possession, ACM Trans. Inf. Syst. Secur. 14 (1) (2011) 12.
[55] D. Cash, A. Küpçü, D. Wichs, Dynamic proofs of retrievability via oblivious RAM, in: EUROCRYPT '13, in: LNCS, Vol. 7881, Springer, 2013, pp. 279–295.
[56] Q. Zheng, S. Xu, Secure and efficient proof of storage with deduplication, in: CODASPY '12, ACM, pp. 1–12.
[57] D. Cash, S. Jarecki, C.S. Jutla, H. Krawczyk, M. Rosu, M. Steiner, Highly-scalable searchable symmetric encryption with support for boolean queries, in: CRYPTO '13, in: LNCS, Vol. 8042, Springer, 2013, pp. 353–373.
[58] D. Cash, J. Jaeger, S. Jarecki, C.S. Jutla, H. Krawczyk, M. Rosu, M. Steiner, Dynamic searchable encryption in very-large databases: Data structures and implementation, in: NDSS '14, The Internet Society, 2014.
[59] C. Zuo, J. Macindoe, S. Yang, R. Steinfeld, J.K. Liu, Trusted boolean search on cloud using searchable symmetric encryption, in: TrustCom '16, IEEE, 2016, pp. 113–120.
[60] S.-F. Sun, J.K. Liu, A. Sakzad, R. Steinfeld, T.H. Yuen, An efficient non-interactive multi-client searchable encryption with support for boolean queries, in: ESORICS '16, Part I, in: LNCS, Vol. 9878, Springer, 2016, pp. 154–172.
[61] D. Boneh, G.D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, in: EUROCRYPT '04, in: LNCS, Vol. 3027, Springer, 2004, pp. 506–522.
[62] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, H. Shi, Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions, J. Cryptol. 21 (3) (2008) 350–391.
[63] J. Li, R. Ma, H. Guan, Tees: an efficient search scheme over encrypted data on mobile cloud, IEEE Trans. on Cloud Comput. 1 (2015) 1.
[64] K. Liang, X. Huang, F. Guo, J.K. Liu, Privacy-preserving and regular language search over encrypted cloud data, IEEE Trans. Inf. Forensics Secur. 11 (10) (2016) 2365–2376.
[65] S. Wang, J. Zhou, J.K. Liu, J. Yu, J. Chen, W. Xie, An efficient file hierarchy attribute-based encryption scheme in cloud computing, IEEE Trans. Inf. Forensics Secur. 11 (6) (2016) 1265–1277.
[66] M. Mambo, E. Okamoto, Proxy cryptosystems: Delegation of the power to decrypt ciphertexts, IEICE Trans. E80-A (1997) 54–63.
[67] M. Blaze, G. Bleumer, M. Strauss, 1998, Divertible protocols and atomic proxy cryptography, in: EUROCRYPT, pp. 127–144.
[68] J. Shao, R. Lu, X. Lin, K. Liang, Secure bidirectional proxy re-encryption for cryptographic cloud storage, Pervasive Mobile Comput. 28 (2016) 113–121.
[69] K. Liang, W. Susilo, J.K. Liu, D.S. Wong, Efficient and fully CCA secure conditional proxy re-encryption from hierarchical identity-based encryption, Comput. J. 58 (10) (2015) 2778–2792.
[70] R. Lu, X. Lin, J. Shao, K. Liang, Rcca-secure multi-use bidirectional proxy re-encryption with master secret security, in: ProvSec '14, in: LNCS, Vol. 8782, Springer, 2014, pp. 194–205.
[71] K. Liang, Z. Liu, X. Tan, D.S. Wong, C. Tang, A cca-secure identity-based conditional proxy re-encryption without random oracles, in: ICISC '12, in: LNCS, Vol. 7839, Springer, 2012, pp. 231–246.
[72] K. Liang, C. Chu, X. Tan, D.S. Wong, C. Tang, J. Zhou, Chosen-ciphertext secure multi-hop identity-based conditional proxy re-encryption with constant-size ciphertexts, Theoret. Comput. Sci. 539 (2014) 87–105.
[73] K. Liang, J.K. Liu, D.S. Wong, W. Susilo, An efficient cloud-based revocable identity-based proxy re-encryption scheme for public clouds data sharing, in: ESORICS '14, in: LNCS, Vol. 8712, Springer, 2014, pp. 257–272.
[74] K. Liang, Q. Huang, R. Schlegel, D.S. Wong, C. Tang, A conditional proxy broadcast re-encryption scheme supporting timed-release, in: ISPEC '13, in: LNCS, Vol. 7863, Springer, 2013, pp. 132–146.
[75] C. Chu, J. Weng, S.S.M. Chow, J. Zhou, R.H. Deng, Conditional proxy broadcast re-encryption, in: ACISP '09, in: LNCS, Vol. 5594, Springer, 2009, pp. 327–342.
[76] K. Liang, L. Fang, D.S. Wong, W. Susilo, A ciphertext-policy attribute-based proxy re-encryption scheme for data sharing in public clouds, Concurr. Comput. Prac. Exper. 27 (8) (2015) 2004–2027.
[77] K. Liang, M.H. Au, J.K. Liu, W. Susilo, D.S. Wong, G. Yang, Y. Yu, A. Yang, A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing, Future Gen. Comp. Syst. 52 (2015) 95–108.
[78] K. Liang, M.H. Au, W. Susilo, D.S. Wong, G. Yang, Y. Yu, An adaptively cca-secure ciphertext-policy attribute-based proxy re-encryption for cloud data sharing, in: ISPEC '14, in: LNCS, Vol. 8434, Springer, 2014, pp. 448–461.
[79] K. Liang, L. Fang, W. Susilo, D.S. Wong, A ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security, in: 5th International Conference on Intelligent Networking and Collaborative Systems, IEEE, 2013, pp. 552–559.
[80] K. Liang, M.H. Au, J.K. Liu, W. Susilo, D.S. Wong, G. Yang, T.V.X. Phuong, Q. Xie, A dfa-based functional proxy re-encryption scheme for secure public cloud data sharing, IEEE Trans. Inf. Forensics Secur. 9 (10) (2014) 1667–1680.
[81] Y. Kawai, K. Takashima, Fully-anonymous functional proxy-re-encryption, IACR Cryptology ePrint Archive 2013 (2013) 318.
[82] S. Ohata, Y. Kawai, T. Matsuda, G. Hanaoka, K. Matsuura, Re-encryption verifiability: How to detect malicious activities of a proxy in proxy re-encryption, in: CT-RSA '15 p. 410–428, in: LNCS, Vol. 9048, Springer, 2015, p. p.
[83] T. Matsuo, Proxy re-encryption systems for identity-based encryption, in: Pairing '07, in: LNCS, Vol. 4575, Springer, 2007, pp. 247–267.
[84] T. Mizuno, H. Doi, Hybrid proxy re-encryption scheme for attribute-based encryption, in: Information Security and Cryptology, in: LNCS, Vol. 6151, Springer, 2011, pp. 288–302.
[85] A.N. Khan, M.L.M. Kiah, S.U. Khan, S.A. Madani, Towards secure mobile cloud computing: A survey, Future Gen. Comp. Syst. 29 (5) (2013) 1278–1299.
[86] R. Rivest, L. Adleman, M. Dertouzos, (1978) On data banks and privacy homomorphisms, in Foundations of Secure Computation 169–177.
[87] S. Goldwasser, S. Micali, 1982, Probabilistic encryption and how to play mental poker keeping secret all partial information, in: ACM Symposium on Theory of Computing, pp. 365–377.
[88] P. Paillier, Paillier encryption and signature schemes, in: H.C.A. van Tilborg (Ed.), Encyclopedia of Cryptography and Security, Springer, 2005, p. 5.
[89] D. Naccache, J. Stern, A new public key cryptosystem based on higher residues, in: CCS '98, ACM, pp. 59–66.
[90] T. Okamoto, S. Uchiyama, A new public-key cryptosystem as secure as factoring, in: EUROCRYPT '98, in: LNCS, Vol. 1403, Springer, 1998, pp. 308–318.
[91] S.D. Galbraith, Elliptic curve paillier schemes, J. Cryptol. 15 (2) (2002) 129–138.
[92] C. Gentry, Fully homomorphic encryption using ideal lattices, in: STOC '09, ACM, pp. 169–178.
[93] C. Gentry, S. Halevi, Implementing gentry's fully-homomorphic encryption scheme, in: EUROCRYPT '11, in: LNCS, Vol. 6632, Springer, 2011, pp. 129–148.
[94] C. Gentry, S. Halevi, N.P. Smart, Homomorphic evaluation of the AES circuit, in: CRYPTO '12, in: LNCS, Vol. 7417, Springer, 2012, pp. 850–867.
[95] M.R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, M. Zohner, Ciphers for MPC and FHE, in: EUROCRYPT '15, in: LNCS, Vol. 9056, Springer, 2015, pp. 430–454.
[96] B. Chor, A. Fiat, M. Naor, Tracing traitors, in: CRYPTO '94, in: LNCS, Vol. 839, Springer, 1994, pp. 257–270.
[97] D. Naor, M. Naor, Protecting cryptographic keys: The trace-and-revoke approach, IEEE Comput. 36 (7) (2003) 47–53.
[98] M. Naor, B. Pinkas, Efficient trace and revoke schemes, Int. J. Inf. Sec. 9 (6) (2010) 411–424.
[99] J. Wei, G. Yang, Y. Mu, K. Liang, Anonymous proxy signature with hierarchical traceability, Comput. J. 59 (4) (2016) 559–569.
[100] D. Boneh, M.K. Franklin, An efficient public key traitor tracing scheme, in: CRYPTO '99, in: LNCS, Vol. 1666, Springer, 1999, pp. 338–353.
[101] D. Boneh, M. Naor, Traitor tracing with constant size ciphertext, in: ACM CCS '08, pp. 501–510.
[102] B. Libert, M. Yung, M. Joye, T. Peters, Traceable group encryption, in: PKC '14, in: LNCS, Vol. 8383, Springer, 2014, pp. 592–610.
[103] M.H. Au, Q. Huang, J.K. Liu, W. Susilo, D.S. Wong, G. Yang, Traceable and retrievable identity-based encryption, in: ACNS '08, in: LNCS, Vol. 5037, 2008, pp. 94–110.

[104] M. Strefler, Broadcast Encryption with Traitor Tracing. (Diffusion Chiffrée Avec Traçage De Traîtres), École Normale Supérieure, Paris, France, 2013 Ph.D.thesis.
[105] J. Zhou, Z. Cao, X. Dong, X. Lin, TR-MABE: white-box traceable and revocable multi-authority attribute-based encryption and its applications to multi-level privacy-preserving e-healthcare cloud computing systems, in: IEEE INFOCOM '15, pp. 2398–2406.
[106] Z. Liu, D.S. Wong, Practical attribute-based encryption: traitor tracing, revocation and large universe, Comput. J. 59 (7) (2016) 983–1004.
[107] J. Katz, D. Schroder, Tracing insider attacks in the context of predicate encryption schemes, In ACITA.
[108] K. Liang, W. Susilo, Searchable attribute-based mechanism with efficient data sharing for secure cloud storage, IEEE Trans. on Inf. Forensics Secur. 10 (9) (2015) 1981–1992.
[109] K. Liang, C. Su, J. Chen, J.K. Liu, Efficient multi-function data sharing and searching mechanism for cloud-based encrypted data, in: AsiaCCS '16, ACM, 2016, pp. 83–94.



Man Ho Allen Au received the Ph.D. degree from the University of Wollongong, Australia, in 2009. Currently, he is an assistant professor at Dept. of Computing, the Hong Kong Polytechnic University. Before that, he was a lecturer at the School of Computer Science and Software Engineering, University of Wollongong, Australia. His research interests include Information Security and Privacy, Applied Cryptography, Accountable Anonymity and Cloud Computing. He has published over 100 papers in those areas in journals such as IEEE Transactions on Information Forensics and Security, IEEE Transactions on Knowledge and Data Engineering, ACM Transaction on Information and System Security and international conferences including the Network and Distributed System Security Symposium (NDSS) and the ACM Conference on Computer and Communications Security (CCS). He has served as a program committee member in over 30 international conferences/workshops. He is also a program committee co-chair of the 8th International Conference on Network and System Security and the 9th International Conference on Provable Security. He is an associate editor of the Journal of Information Security and Applications, Elsevier. He has served as a guest editor for various journals including Future Generation Computer Systems, Elsevier and Concurrency and Computation: Practice and Experience, Wiley.

Kaitai Liang received the Ph.D. degree from the Department of Computer Science, City University of Hong Kong, in 2014. He is currently an assistant professor at School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, UK. Before join Manchester Metropolitan University, he was a postdoctoral researcher with the Department of Computer Science, Aalto University, Finland. His research interests are cyber-security, privacy and security in information technology, in particular, big data security, privacy enhancing technology, genomic privacy, cloud security, privacy in Internet of Things, and lightweight secure systems.

Joseph Liu received the Ph.D. degree in Information Engineering from the Chinese University of Hong Kong in July 2004, specializing in cyber security, protocols for securing wireless networks, privacy, authentication, and provable security. He is now a senior lecturer at Faculty of Information Technology, Monash University, Australia. Prior to that, he was a Research Scientist at Infocomm Security Department, Institute for Infocomm Research (I2R) in Singapore for more than 7 years. His current technical focus is particularly cyber security in the cloud computing paradigm, big data, lightweight security, and privacy enhanced technology. He has published more than 100 referred journal and conference papers and received the Best Paper Award from ESORICS 2014 and ESORICS 2015. He is the co-founder of ProvSec (International Conference on Provable Security). He has served as the program chair of ProvSec 2007, 2014, ACISP 2016. ISPEC 2017, and as the program committee of more than 50 international conferences.

Rongxing Lu has been an assistant professor at the Faculty of Computer Science, University of New Brunswick (UNB), Canada, since August 2016. Before that, he worked as an assistant professor at the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore from May 2012 to August 2016. Rongxing Lu worked as a Postdoctoral Fellow at the University of Waterloo from May 2012 to April 2013. He was awarded the most prestigious "Governor General's Gold Medal", when he received his PhD degree from the Department of Electrical & Computer Engineering, University of Waterloo, Canada, in 2012; and won the 8th IEEE Communications Society (ComSoc) Asia Pacific (AP) Outstanding Young Researcher Award, in 2013. He is presently a senior member of IEEE Communications Society. His research interests include applied cryptography, privacy enhancing technologies, and IoT-Big Data security and privacy. He has published extensively in his areas of expertise (with more than 7500 citations from Google Scholar), and was the recipient (with his students and colleagues) of the Student Best Paper Award, ITS Summit Singapore 2015, the IEEE IES Student Best Paper Award 2014, the Best Paper Awards of TSINGHUA Science and Technology Journal 2014, IEEE ICCC 2015, IEEE WCNC 2013, BodyNets 2010, and IEEE ICCCN 2009. He was/is on the editorial boards of several international referred journals, e.g., IEEE Network, and currently serves the technical symposium co-chair of IEEE Globecom'16, and many technical program committees of IEEE and others international conferences, including IEEE INFOCOM and ICC. In addition, he is currently organizing a special issue on "security and privacy issues in fog computing" in Elsevier Journal "Future Generation Computer Systems" and a special issue on "big security challenges in big data era" in IEEE Internet of Things Journal. Dr. Lu currently serves as the Secretary of IEEE ComSoc CIS-TC (Communications and Information Security Technical Committee).

Jianting Ning received the Ph.D. degree from the Department of Computer Science and Engineering, Shanghai Jiao Tong University in 2016. He is currently a research fellow at Department of Computer Science, National University of Singapore. His research interests include applied cryptography and cloud security, in particular, Public Key Encryption, Attribute-Based Encryption, and Secure Multiparty Computation.