www.ietdl.org

Published in IET Information Security
Received on 28th January 2014
Revised on 18th July 2014
Accepted on 29th July 2014
doi: 10.1049/iet-ifs.2013.0552

ISSN 1751-8709

# RIKE+ : using revocable identities to support key escrow in public key infrastructures with flexibility

*Jingqiang Lin[1,2], Wen-Tao Zhu[1,2], Qiongxiao Wang[1,2], Nan Zhang[1,2], Jiwu Jing[1,2], Neng Gao[1,2]*

[1]*Data Assurance and Communication Security Research Center, Chinese Academy of Sciences, Beijing, People's Republic of China*
[2]*State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, People's Republic of China*
*E-mail: linjq@is.ac.cn*

**Abstract:** Public key infrastructures (PKIs) are proposed to provide various security services. Some security services such as confidentiality require key escrow in certain scenarios, whereas some others such as non-repudiation and authentication usually prohibit key escrow. Moreover, these two conflicting requirements can coexist for one PKI user. The popular solution in which each user has two different certificates and an escrow authority backs up all escrowed private keys faces the problems of efficiency and scalability. In this study, a novel key management infrastructure called RIKE+ is proposed to integrate the 'inherent key escrow' of identity-based encryption (IBE) into PKIs. In RIKE+, (the hash value of) a user's PKI certificate also serves as a 'revocable identity' to derive the user's IBE public key, and the revocation of this IBE key pair is achieved by the certificate revocation of PKIs. Therefore the certificate binds the user with two key pairs, one of which is escrowed inherently and the other is not. Furthermore, RIKE+ employs chameleon hash to flexibly control the relationship between the certificate and the IBE key pair. In the case of certificate renewal and revocation, chameleon hash enables RIKE+ to manipulate the hash value of the new certificate, so the user's IBE key pair is not unconditionally changed unless it is necessary. RIKE+ is an effective certificate-based solution compatible with traditional PKIs and can be built on existing X.509 PKIs.

## 1 Introduction

Public key infrastructures (PKIs) are proposed to securely publish public keys. In PKIs, the public key of a user is bound to its identity in a certificate [1], signed by a certification authority (CA). In this paper, the term 'user' may refer to a person, a device, an application process or any equivalent entity. After querying and validating a certificate, everybody can use the contained public key for security services such as authentication, confidentiality, data integrity and non-repudiation.

Key escrow is required or prohibited depending on different scenarios, even for one given PKI user. If a key pair is used for data encryption and decryption, key escrow is usually needed. A typical example is that, in a corporation, the backups of all employees' private keys are securely maintained by a trusted party, called the escrow authority (EA); so the corporation can decrypt the encrypted data in case some employee's private key is unavailable. On the other hand, if the key pair is used to sign and verify messages for non-repudiation, key escrow is usually prohibited. For example, key escrow is forbidden or unrecommended in the digital signature laws and the guidelines of several countries and organisations [2–5].

In the popular PKI solutions, each user holds two key pairs in two separate certificates, to distinguish the key pair for confidentiality from the one for authentication and non-repudiation. The conflicting requirements of key escrow are then satisfied by this two-certificate design. The key pair used for non-repudiation, does not support key escrow, while the other does. An instruction (called the key-usage extension [6]) is embedded in each certificate to indicate the purpose of the key pair. In this two-certificate solution, every user generates the key pair for authentication and non-repudiation by itself, and the EA generates the user's escrowed key pair for other purposes. As a result, more resources are needed to sign, publish, store and revoke certificates. Moreover, although each user holds only two valid key pairs, the EA faces the problem of scalability: as new users join the PKI system and new key pairs are generated to replace expired or revoked certificates, the EA's burden of maintaining the growing amount of escrowed key pairs becomes very heavy.

Identity-based encryption (IBE) [7] is a special type of public key algorithms, with the feature of inherent key escrow. In IBE, a trusted private key generator (PKG) initialises a secret master key and publishes the corresponding public parameters. A user's public key can be calculated from its identity (and the public parameters) by anybody. When receiving messages encrypted by its public key, the user asks the PKG to generate the private key corresponding to its identity (if it does not hold the

private key yet). As for key recovery, the PKG only needs to hold and protect the secret master key, to regenerate the private keys for all identities (or users). However, key revocation (especially when a private key is compromised) is a problem in IBE, since the public keys are bound to identities automatically and users are not willing to change their identities. On the contrary, there are various certificate revocation mechanisms in PKIs, applicable to different scenarios.

The above observation that IBE and PKIs have complementary advantages motivates us to integrate IBE and PKIs. In this paper, we propose RIKE+, using revocable identities of IBE to support key escrow in PKIs. RIKE+ does not invent any new public key algorithm; instead, it is an innovative key management infrastructure assembling the advantages of these two cryptosystems. Each RIKE+ user has only one certificate and the contained public key is only used for the security services prohibiting or not requiring key escrow. The other services requiring key escrow are achieved through IBE; however, the IBE public key is derived from (the hash value of) the user's certificate, not from its real identity. Compared with traditional PKIs, RIKE+ provides security services with the conflicting requirements of key escrow, while each user has only one certificate and two key pairs. Moreover, supporting key escrow in RIKE+ is much easier than that in traditional PKIs, because of the inherent key escrow of IBE.

RIKE+ also solves the key revocation problem of IBE, by utilising the certificate revocation mechanisms of PKIs. Unlike the public key derived from a user's unchangeable identity in IBE, the IBE public key in RIKE+ is derived from the hash value of the certificate after validating the user's certificate, including checking its revocation status. If a user's IBE private key is compromised, the CA revokes its PKI certificate and the corresponding IBE public key shall not be used any more. Once the new certificate with a different hash value is issued, another IBE key pair is available automatically while the user's identity keeps unchanged. Thus, a certificate in RIKE+ also works as a 'revocable identity'.

To avoid unconditionally changing the IBE key pair in the case of certificate renewal and revocation, RIKE+ employs chameleon hash [8] to control the relationship between the user's certificate and its IBE key pair. A chameleon hash function is a trapdoor collision-resistant hash function: without the trapdoor, it is collision-resistant; however, collisions can be found once the trapdoor is known. When a new certificate is issued to replace the revoked or expired one, the chameleon hash function enables RIKE+ to manipulate the hash value of the new certificate (i.e. the new IBE key pair). Therefore the IBE key pair is changed only if it is necessary. For example, when the certificate is revoked because the non-escrowed key pair is compromised and a new one is issued, the IBE key pair can be kept unchanged for it is not compromised.

Another advantage of RIKE+ is the high compatibility with traditional X.509 standards [6, 9]. RIKE+ can be implemented on the prevailing X.509 PKIs, with newly designed certificate extensions (see Section 4.3 for details). If somebody receives a certificate with the RIKE+ extensions that it does not support, it ignores these extensions and processes the certificate as an ordinary X.509 certificate. More importantly, existing X.509 PKIs can be smoothly transferred to RIKE+ gradually by issuing certificates with RIKE+ extensions.

This paper is organised as follows. Section 2 surveys the related work. Then, Section 3 presents the architecture and the analysis of RIKE+. Section 4 discusses how to implement RIKE+ compatibly with X.509 PKIs. Finally, Section 5 draws the conclusion.

## 2 Related work

How to perfectly support key escrow in PKIs is still an open problem. A popular approach is to store the backup of a user's private key in a centralised component [10, 11]. This solution is compatible with traditional PKIs, but lacks scalability because more and more private keys shall be stored as the number of users increases and certificates expire. Self-escrow PKIs (SE-PKIs) [12, 13] are proposed to embed the public part of trapdoor information when users generate private keys. The key recovery agent (KRA), holding the secret part of trapdoor information, can recover users' private keys. SE-PKIs offer a convenient way to escrow the user private keys, whereas RIKE+ integrates the inherent key escrow of IBE into PKIs to manage both escrowed and non-escrowed key pairs.

The concept of IBE is proposed by Shamir [7] in 1985. Boneh and Franklin [14] and Cocks [15] invented secure IBE algorithms, in which an arbitrary bit-string can be used as a public key to encrypt messages. Hierarchical IBE [16, 17] is designed to distribute the workload of the centralised PKG. All identity-based algorithms have two essential features: (a) a user's identity is used as its public key, so the certificate is eliminated and (b) IBE inherently supports key escrow because all users' private keys are generated by PKGs and can be recovered by PKGs.

The inherent key escrow is usually considered as a drawback of IBE because of the risk that the users' private keys might be maliciously disclosed or used by the PKG. Some [18, 19] proposed to generate the private keys by distributed PKGs, so that the keys are still secure unless all the PKGs are compromised. Alternatively, the privilege of PKGs can be constrained [20–22]. In certificate-based encryption [21], a user's non-escrowed secret key and a certificate from its CA are both needed to decrypt messages. In certificateless public key cryptography [20], the private key is generated by a user and the key generating centre cooperatively. Accountable authority IBE [22, 23] mitigates the problem in a passive manner: if the PKG regenerates the private key, a proof will be produced automatically. However, this feature can be leveraged in the scenarios where key escrow is necessary, and the inherence makes the key escrow design of RIKE+ concise and efficient.

The intrinsic binding of identities and public keys introduces a problem in IBE when the key pair needs to be revoked (e.g. the private key is compromised), because identities are usually expected to remain unchanged. To deal with the revocation problem, the user's identity and a period of validity can be combined together to derive its public key [14]. Once the period expires, the key pair becomes invalid; however, if the private key is compromised before the expiration, the key pair cannot be revoked in this way. Therefore the period of validity shall be very short for high security, therefore users shall apply for and the PKG shall generate private keys frequently. The security mediator (SEM) architecture originally proposed to revoke certificates in PKIs [24] is applied in IBE [25, 26]: an online SEM keeps a partition of each private key and every decryption operation requires the SEM's help. Revocation is achieved as long as the SEM stops helping a user to decrypt messages. However, an always-online SEM

service faces more risks, and more protections are needed in practical deployment.

On the contrary, a public key (or certificate) in PKIs is used only after its revocation status is checked. Lots of approaches are proposed to revoke certificates such as certificate revocation lists (CRLs) [6], over-issued CRLs [27], delta-CRLs [6], redirect CRLs [28], sliding window delta-CRLs [29], the online certificate status protocol [30], NOVOMODO [31], certificate revocation trees [32], depender graphs [33], windowed revocation [34], authenticated dictionaries [35], space partitioning [36] and counter certificates [37]. Various revocation mechanisms have advantages in different environments and scenarios, and all can be used in RIKE+ to revoke certificates. More detailed comparisons and evaluations of certificate revocation mechanisms can be found in [38, 39].

Some schemes are proposed to provide benefits similar to IBE, focusing on the compatibility issue [40–42], but key escrow is not considered. The interoperability between PKIs and IBE is also widely studied. IBE algorithms are suggested to be used at the user level [43], while the trusted authorities at the domain level are connected by certificates. A certificate-based solution [44] is designed to interoperate between a PKI domain and an IBE one. Our solution applies IBE to support key escrow in PKIs, keeping compatibility with X.509 PKIs [6, 9].

Chameleon hash [8] was originally introduced to construct chameleon signatures that possess the properties of non-repudiation and non-transferability for signed messages. Inspired by IBE, identity-based chameleon hash (IBCH) was invented [45, 46]: the identity of a recipient is used as the public hashing key by any hasher, and the recipient can find collisions using the private hashing key generated by the PKG. Pairing-based IBCH is also extended as hierarchical IBCH to support multiple levels of users [47]. Chameleon hash is designed for chameleon signatures and sanitisable signatures [48, 49, 8], but we employ it in a more straightforward way: the PKG in RIKE+ that knows the private hashing key is enabled to manipulate the chameleon hash value of a certificate.

Compared with the preliminary version [50], RIKE+ employs chameleon hash to flexibly control the relationship between the certificate and the IBE key pair, in the case of certificate renewal and revocation. The detailed guidelines on transferring existing X.509 PKIs to RIKE+, which are not included in the preliminary version, are discussed in Section 4.

## 3 RIKE+: using revocable identities to support key escrow in PKIs with flexibility

In this section, we firstly describe the background of PKIs and IBE, the basic RIKE+ and the architecture with chameleon hash. Then, we extend it as hierarchical RIKE+, to work with hierarchical PKIs and cross-certification. Some flexible implementation options are also presented. Finally, we summarise the features of RIKE+ and the comparisons with other schemes.

### 3.1 Background

PKIs are built based on traditional public key algorithms. A CA signs certificates, each of which binds a user's identity and its public key. Trusting the CA, somebody verifies the CA's signature and obtains the user's identity and public key. Then, the certified information is used for security services. The entity holding the private key is called a 'user', and the entity that uses a user's public key to verify or encrypt messages is called a 'relying-party' [6]. A user is usually also a relying-party, but a relying-party may be either a user or not.

The notations about PKIs are listed as follows. The superscript $\mathcal{P}$ indicates the key pairs in PKIs (to distinguish with those key pairs in IBE for which the superscript $\mathcal{I}$ is used, and those in chameleon hash for which the superscript $\mathcal{H}$ is used).

– $\mathrm{ID}_U$, $\mathrm{ID}_{\mathrm{CA}}$: the identities of a user $U$ and the CA, respectively.
– $\mathrm{PK}^{\mathcal{P}}$: a public key of traditional public key algorithms.
– $\mathrm{SK}^{\mathcal{P}}$: a private key of traditional public key algorithms.
– $\mathrm{Cert}(U[,e_1|e_2|\cdots])=\mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}}}(\mathrm{ID}_{\mathrm{CA}},\mathrm{ID}_U|\mathrm{PK}^{\mathcal{P}}_U[,e_1|e_2|\cdots])$: the certificate signed by the CA, binding $\mathrm{ID}_U$ and $\mathrm{PK}^{\mathcal{P}}_U$ with an 'optional' list of extensions $e_1|e_2|\ldots$. The PKI certificate is composed of the certified information and the CA's signature, but for the sake of simplicity, we emphasise the certified information in the form of $\mathrm{Cert}(\cdot)$ or the signature in the form of $\mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}}}(\cdot)$ alternatively. The CA and $U$ are called the 'issuer' and the 'subject' of the certificate, respectively.

IBE is a special type of public key algorithms, where a public key is derived from the user's identity. A PKG generates a master key and public parameters. Then, any entity can calculate a user's public key based on the user's identity and the public parameters. Only the PKG holding the secret master key can calculate the corresponding private key. The notations about IBE are as follows:

– $\mathrm{PM}^{\mathcal{I}}$: the IBE public parameters generated by the PKG.
– $\mathrm{MK}^{\mathcal{I}}$: the PKG's IBE master key.
– $\mathrm{PK}^{\mathcal{I}}(\mathrm{ID}_U)=\mathrm{GenPK}(\mathrm{ID}_U, \mathrm{PM}^{\mathcal{I}})$: the IBE public key derived from the identity of $U$ and $\mathrm{PM}^{\mathcal{I}}$. It can be calculated by anybody.
– $\mathrm{SK}^{\mathcal{I}}(\mathrm{ID}_U)=\mathrm{GenSK}(\mathrm{ID}_U, \mathrm{MK}^{\mathcal{I}})$: the IBE private key generated by the PKG for $U$. Only the PKG can calculate it.

### 3.2 Basic RIKE+

The underlying idea of RIKE+ is to use (the hash value of) a user's PKI certificate as its 'identity' to derive the IBE public key, used for the security services requiring key escrow. Then, each RIKE+ user has 'two' key pairs bound in 'only one' certificate: $(\mathrm{PK}^{\mathcal{P}}, \mathrm{SK}^{\mathcal{P}})$ is bound explicitly and $(\mathrm{PK}^{\mathcal{I}}, \mathrm{SK}^{\mathcal{I}})$ is done implicitly. The basic architecture of RIKE+ is composed of a CA, a PKG and an arbitrary number of users. The CA is responsible for issuing and revoking certificates as in traditional PKIs, and the PKG manages escrowed private keys for users.

We describe the basic RIKE+ as follows, where $H(\cdot)$ is a collision-free hash function such as SHA-3 [51].

*Initialisation:* The PKG generates $(\mathrm{PM}^{\mathcal{I}}, \mathrm{MK}^{\mathcal{I}})$ and sends $\mathrm{PM}^{\mathcal{I}}$ to the CA. Then, the CA generates $(\mathrm{PK}^{\mathcal{P}}_{\mathrm{CA}}, \mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}})$ and signs a 'self-signed' certificate $\mathrm{Cert}(\mathrm{CA}, \mathrm{PM}^{\mathcal{I}}) = \mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}}}(\mathrm{ID}_{\mathrm{CA}}, \mathrm{ID}_{\mathrm{CA}}|\mathrm{PK}^{\mathcal{P}}_{\mathrm{CA}}, \mathrm{PM}^{\mathcal{I}})$. $\mathrm{PM}^{\mathcal{I}}$ is embedded in the CA's certificate as an extension. The certificate is delivered to all relying-parties in out-of-band means, as it is done in




doi: 10.1049/iet-ifs.2013.0552

traditional PKIs, while $MK^{\mathcal{I}}$ is known only to the PKG and $SK_{CA}^{\mathcal{P}}$ is only to the CA.

*Certificate and key application:* A user $U$ generates $(PK_U^{\mathcal{P}}, SK_U^{\mathcal{P}})$ by itself, and applies for $Cert(U) = Sign_{SK_{CA}^{\mathcal{P}}}(ID_{CA}, ID_U|PK_U^{\mathcal{P}})$ from the CA and $SK_U^{\mathcal{I}} = SK^{\mathcal{I}}(H(Cert(U)))$ from the PKG. $Cert(U)$ is published publicly by the CA, and the user keeps $SK_U^{\mathcal{P}}$ and $SK_U^{\mathcal{I}}$ secret. $(PK_U^{\mathcal{P}}, SK_U^{\mathcal{P}})$ is called the 'non-escrowed key pair' of $U$, and $(PK_U^{\mathcal{I}}, SK_U^{\mathcal{I}})$ is the 'escrowed key pair' of $U$, where $PK_U^{\mathcal{I}} = PK^{\mathcal{I}}(H(Cert(U)))$ can be calculated by everybody.

*Signing and verification:* $U$ signs a message by $SK_U^{\mathcal{P}}$. Every relying-party in RIKE+ can query and validate $Cert(U)$ to obtain $PK_U^{\mathcal{P}}$, and verify the signed message. The validation of $Cert(U)$ includes checking the period of validity, the CA's signature, and its revocation status, as it does in the traditional PKIs.

*Encryption and decryption:* A relying-party that wants to send encrypted data to $U$, firstly queries and validates $Cert(U)$, and calculates $PK_U^{\mathcal{I}}$ based on $Cert(U)$ and $PM^{\mathcal{I}}$ extracted from the CA's self-signed certificate. Then, data are encrypted by $PK_U^{\mathcal{I}}$ and sent to $U$. On receiving encrypted data, $U$ decrypts them by $SK_U^{\mathcal{I}}$.

### 3.3 Certificate renewal and revocation in basic RIKE+

In RIKE+, before using the public key contained in a certificate or the IBE public key derived from it, a relying-party shall check the CA's signature, the period of validity and its revocation status. All revocation mechanisms in PKIs can be used in RIKE+. If a certificate expires or is revoked, either the non-escrowed or the escrowed key pairs bound in it shall not be used any more.

The two key pairs may change or not after certificate renewal and revocation. A user's certificate can be renewed when it expires. The non-escrowed key pair will be either regenerated, or kept unchanged if it is still considered secure. In the new certificate, at least the serial number and the period of validity are updated, so the escrowed key pair changes automatically. A user's certificate is revoked and a new one with valid information is issued, when (a) the identity information in the certificate becomes invalid, for example, its affiliation changes or (b) the non-escrowed or the escrowed key pair is (suspected to be) compromised.

– If the certificate is revoked because of the invalid identity information, the user's non-escrowed key pair may keep unchanged, but the escrowed key pair derived from the new certificate always becomes different.
– If it is revoked because of the compromised non-escrowed key pair, both the two key pairs change even when the escrowed one is not compromised.
– If it is revoked because of the compromised escrowed key pair, the user's non-escrowed key pair can keep unchanged, but at least one bit in the certificate shall be updated, for example, the serial number, to change the IBE key pair.

In summary, the non-escrowed key pair does not change unless it needs to change, whereas the escrowed key pair always changes once the certificate is replaced by a new one. A relying-party does not need to care about whether the key pairs have changed or not, for it just validates a certificate as in traditional PKIs and obtains the public keys. Although there is no extra inconvenience for relying-parties to deal with certificate renewal and revocation in RIKE+, this feature brings unnecessary burden of key management to users.

### 3.4 RIKE+ with chameleon hash

To manipulate the hash value of a certificate and then the IBE key pair derived from it, IBCH is employed in RIKE+ to replace the ordinary collision-free hash. That is, the 'chameleon hash' value of a certificate serves as the 'identity' to derive the user's IBE public key.

In IBCH, there is also a PKG that generates a secret master key and public parameters. The PKG generates the private hashing key for an IBCH user corresponding to the user's identity, and this identity is an input of the chameleon hash function as well as messages. Then, the IBCH user can find collisions using the private hashing key. The notations about IBCH are listed as follows:

– $PM^{\mathcal{H}}$: the IBCH public parameters generated by the PKG.
– $MK^{\mathcal{H}}$: the PKG's IBCH master key.
– $h = CH(ID_U, PM^{\mathcal{H}}, m, r)$: the chameleon hash value of a message $m$ for $U$, where $r$ is a public random string. It can be calculated by anybody.
– $SK^{\mathcal{H}}(ID_U) = GenHK(ID_U, MK^{\mathcal{H}})$: the private hashing key of $U$ generated by the PKG. Only the PKG can calculate it.
– $r' = Forge(ID_U, SK^{\mathcal{H}}(ID_U), m, r, m')$: the chameleon random string $r'$ calculated for a given message $m' \neq m$, satisfying that $CH(ID_U, PM^{\mathcal{H}}, m', r') = h = CH(ID_U, PM^{\mathcal{H}}, m, r)$.

In the RIKE+ scheme improved by IBCH, the PKG generates $(PM^{\mathcal{H}}, MK^{\mathcal{H}})$ in addition to $(PM^{\mathcal{I}}, MK^{\mathcal{I}})$. It keeps $MK^{\mathcal{H}}$ and $MK^{\mathcal{I}}$ secret, and $PM^{\mathcal{H}}$ is embedded in the CA's certificate together with $PM^{\mathcal{I}}$ as an extension. Before signing a certificate $Cert(U)$, the CA chooses a chameleon random string $r_U$ and embeds it into the certificate as another extension (called the 'RIKE-flag' extension; see Section 4.3 for details). The identity-based feature of IBCH eliminates the need to carry the public hashing keys in certificates.

A certificate in PKIs (and also RIKE+) is further represented as three separate parts $Cert(U) = CertRaw_U|CertExt_U|Sig_U$, where $CertRaw_U$ is composed of the elementary fields of the certificate, $CertExt_U$ is the optional list of extensions and $Sig_U$ is the signature. For example, the elementary fields of an X.509 certificate include a version, a serial number, the algorithm used to sign the certificate, the identities of an issuer and a subject, a period of validity and the contained public key [6].

Compared with the basic RIKE+ scheme in Section 3.2, the 'identity' to derive the IBE public key is updated from $H(Cert(U))$ to $CH(Cert(U)) = CH(ID_U, PM^{\mathcal{H}}, CertRaw_U, r_U)$. The IBCH function takes $CertRaw_U$ instead of $Cert(U)$ as its input, so $r_U$ is chosen based on $CertRaw_U$ before signing the certificate. Note that the 'identity' input of IBCH is the real identity of $U$, that is, $ID_U$, whereas that of IBE is the chameleon hash value of $Cert(U)$, that is, $CH(Cert(U))$. And $r_U$ is chosen as follows: if $Cert(U)$ is issued to replace to another certificate $Cert'(U)$ and the corresponding IBE key pair shall be kept unchanged, the PKG generates $r_U = Forge(ID_U, SK^{\mathcal{H}}(ID_U), CertRaw'_U, r'_U, CertRaw_U)$ where $r'_U$ is the chameleon random string of $Cert'(U)$, and sends it the CA; otherwise, $r_U$ is chosen arbitrarily by the CA.

Employing chameleon hash does not impose any additional burden of protection on users. A tricky difference from original IBCH is that $\mathrm{SK}^{\mathcal{H}}(\mathrm{ID}_U)$ is not distributed to $U$. It is held only by the PKG to control the IBE key pair, cooperatively with the CA. RIKE+ users do not need to hold any private (hashing) keys except $\mathrm{SK}^{\mathcal{P}}_U$ and $\mathrm{SK}^{\mathcal{I}}_U$, as they do in tradition PKIs.

### 3.5 Hierarchical RIKE+

PKIs are usually built hierarchically [6]. A CA (called the 'root CA') generates the self-signed certificate, and signs certificates for the second-level users, some of which further sign certificates for other users. A user that signs certificates for others is called a 'subordinate CA'. In this paper, we call it a user or a CA alternatively according to the context. This structure can be extended to support more levels, where each user applies for a certificate from an upper-level user or the root CA. The notations about the identities in a hierarchical PKI are defined as follows:

– $U_{j,k}$: the $k$th $j$th-level user; particularly, the root CA is denoted as $U_{1,1}$.
– $\mathrm{ID}_{U_{j,k}}$: the identity of the $k$th $j$th-level user $U_{j,k}$.
– $\mathbf{ID}_{U_{j,k}} = (\mathbf{ID}_{\bigstar}, \mathrm{ID}_{U_{j,k}})$: the 'identity chain' of $U_{j,k}$, where $\mathbf{ID}_{\bigstar}$ is (a) the identity chain of another user $U_{j-1,k'}$ if Cert$(U_{j,k})$ is signed by $U_{j-1,k'}$ or (b) null if it is signed by the root CA. That is, the root CA's identity chain is also null.

To validate Cert$(U_{j,k})$, a vector of certificates (called the 'certificate chain') is needed: $\mathbf{Cert}(U_{j,k}) = (\mathrm{Cert}(U_{2,k_2}), \ldots, \mathrm{Cert}(U_{j-1,k_{j-1}}), \mathrm{Cert}(U_{j,k}))$, where the issuer of each certificate is the subject of the preceding one and Cert$(U_{2,k_2})$ is signed by the root CA. Every relying-party uses the root CA's self-signed certificate to validate Cert$(U_{2,k_2})$, and then repeatedly uses a validated certificate to validate the next one in the vector until Cert$(U_{j,k})$ is validated.

The hierarchical structure brings benefits. The root CA's workload is distributed among lower-level subordinated CAs, and users can apply for certificates locally. The root CA serves only a few users (or subordinate CAs) and then works off-line in most time to reduce attack risks.

To implement RIKE+ on hierarchical PKIs in a compatible way, we propose to build hierarchical RIKE+ as follows by employing hierarchical IBE [16, 17] and hierarchical IBCH [47]. Hierarchical IBE works as follows. A PKG (called the 'root PKG') generates its master key and the public key parameters, and generates IBE private keys for the second-level users, some of which further generate private keys for others. The user that generates IBE private keys for others is also called a 'subordinated PKG'. This structure can be extended to support more levels, where all users' private keys are generated by the root PKG's master key directly or indirectly. Hierarchical IBCH works similarly to hierarchical IBE, except that each user's private hashing key instead of its IBE private key, is generated by a subordinated PKG or the root PKG.

In hierarchical IBE and hierarchical IBCH, a user's IBE public key is derived from $\mathbf{ID}_{U_{j,k}}$, the IBCH function takes $\mathbf{ID}_{U_{j,k}}$ as an input, and its IBE private key and private hashing key are generated by $U_{j-1,k_{j-1}}$ as follows:

– $\mathrm{PK}^{\mathcal{I}}(\mathbf{ID}_{U_{j,k}}) = \mathrm{GenPK}(\mathbf{ID}_{U_{j,k}}, \mathrm{PM}^{\mathcal{I}})$: the IBE public key of $U_{j,k}$ derived from $\mathbf{ID}_{U_{j,k}}$ and $\mathrm{PM}^{\mathcal{I}}$.
– $\mathrm{SK}^{\mathcal{I}}(\mathbf{ID}_{U_{j,k}}) = \mathrm{GenSK}(\mathrm{ID}_{U_{j,k}}, \mathrm{SK}^{\mathcal{I}}(\mathbf{ID}_{U_{j-1,k_{j-1}}}))$: the IBE private key of $U_{j,k}$ generated by $U_{j-1,k_{j-1}}$. Note that $\mathrm{SK}^{\mathcal{I}}(\mathbf{ID}_{U_{1,1}}) = \mathrm{MK}^{\mathcal{I}}$ is the root PKG's IBE master key.
– $h = \mathrm{CH}(\mathbf{ID}_{U_{j,k}}, \mathrm{PM}^{\mathcal{H}}, m, r)$: the chameleon hash value of a message $m$ for $U_{j,k}$, where $r$ is a public chameleon random string.
– $\mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{j,k}}) = \mathrm{GenHK}(\mathrm{ID}_{U_{j,k}}, \mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{j-1,k_{j-1}}}))$: the private hashing key of $U_{j,k}$ generated by $U_{j-1,k_{j-1}}$. Note that $\mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{1,1}}) = \mathrm{MK}^{\mathcal{H}}$ is the root PKG's IBCH master key.
– $r' = \mathrm{Forge}(\mathbf{ID}_{U_{j,k}}, \mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{j,k}}), m, r, m')$: the chameleon random string $r'$ calculated for a given message $m' \neq m$, satisfying that $\mathrm{CH}(\mathbf{ID}_{U_{j,k}}, \mathrm{PM}^{\mathcal{H}}, m', r') = h = \mathrm{CH}(\mathbf{ID}_{U_{j,k}}, \mathrm{PM}^{\mathcal{H}}, m, r)$.

Finally, hierarchical RIKE+ is built on hierarchical PKIs, where each CA cooperates with its corresponding PKG to generate the users' IBE key pairs, and these PKGs work as the PKGs of hierarchical IBE and IBCH (denoted as $U^{\mathcal{I}}_{j,k_j}$ in the following sections). The IBE and IBCH public parameters are published only in the root CA's self-signed certificate. Therefore with the same public parameters, the chameleon hash values of the certificates in each user's certificate chain are used to generate its IBE public key. Each user applies for its IBE private key from the subordinate PKG, corresponding to the CA that it is subordinated to.

In particular, the 'identity' chain, each 'identity' in which is the chameleon hash value of a certificate in $\mathbf{Cert}(U_{j,k})$, composes the input of hierarchical IBE, whereas the hierarchical IBCH function takes the real identity chain of $U_{j,k}$ as inputs, to calculate the chameleon hash value of Cert$(U_{j,k})$. Let us define $\mathrm{CH}(\mathbf{Cert}(U_{j,k})) = (\mathrm{CH}(\mathrm{Cert}(U_{2,k_2})), \ldots, \mathrm{CH}(\mathrm{Cert}(U_{j-1,k_{j-1}})), \mathrm{CH}(\mathrm{Cert}(U_{j,k})))$ and $\mathrm{CH}(\mathrm{Cert}(U)) = \mathrm{CH}(\mathbf{ID}_U, \mathrm{PM}^{\mathcal{H}}, \mathrm{CertRaw}_U, r_U)$. Then, hierarchical RIKE+ works as follows:

– The root PKG generates $(\mathrm{PM}^{\mathcal{I}}, \mathrm{MK}^{\mathcal{I}})$ and $(\mathrm{PM}^{\mathcal{H}}, \mathrm{MK}^{\mathcal{H}})$, and sends $\mathrm{PM}^{\mathcal{I}}$ and $\mathrm{PM}^{\mathcal{H}}$ to the root CA. The root CA generates $(\mathrm{PK}^{\mathcal{P}}_{\mathrm{CA}}, \mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}})$, and signs a self-signed certificate $\mathrm{Cert}(\mathrm{CA}, \mathrm{PM}^{\mathcal{I}}|\mathrm{PM}^{\mathcal{H}}) = \mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}}}(\mathrm{ID}_{\mathrm{CA}}, \mathrm{ID}_{\mathrm{CA}}|\mathrm{PK}^{\mathcal{P}}_{\mathrm{CA}}, \mathrm{PM}^{\mathcal{I}}|\mathrm{PM}^{\mathcal{H}})$.
– A second-level user $U_{2,k}$ generates $(\mathrm{PK}^{\mathcal{P}}_{U_{2,k}}, \mathrm{SK}^{\mathcal{P}}_{U_{2,k}})$, and applies for $\mathrm{Cert}(U_{2,k}) = \mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{\mathrm{CA}}}(\mathrm{ID}_{\mathrm{CA}}, \mathrm{ID}_{U_{2,k}}|\mathrm{PK}^{\mathcal{P}}_{U_{2,k}}, r_{U_{2,k}})$ from the root CA and $\mathrm{SK}^{\mathcal{I}}_{U_{2,k}} = \mathrm{SK}^{\mathcal{I}}(\mathrm{CH}(\mathbf{Cert}(U_{2,k}))) = \mathrm{GenSK}(\mathrm{CH}(\mathrm{Cert}(U_{2,k})), \mathrm{MK}^{\mathcal{I}})$ from the root PKG. Here, $(\mathrm{PK}^{\mathcal{P}}_{U_{2,k}}, \mathrm{SK}^{\mathcal{P}}_{U_{2,k}})$ is the non-escrowed key pair of $U_{2,k}$ and $(\mathrm{PK}^{\mathcal{I}}_{U_{2,k}}, \mathrm{SK}^{\mathcal{I}}_{U_{2,k}})$ is the escrowed key pair of $U_{2,k}$, where $\mathrm{PK}^{\mathcal{I}}_{U_{2,k}} = \mathrm{PK}^{\mathcal{I}}(\mathrm{CH}(\mathbf{Cert}(U_{2,k})))$. In addition, $\mathrm{SK}^{\mathcal{H}}_{U_{2,k}} = \mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{2,k}}) = \mathrm{GenHK}(\mathrm{ID}_{U_{2,k}}, \mathrm{MK}^{\mathcal{H}})$ is distributed to $U^{\mathcal{I}}_{2,k}$, if $U_{2,k}$ is a subordinate CA authorised to sign certificates and $U^{\mathcal{I}}_{2,k}$ is its corresponding PKG.
– A third-level user $U_{3,k}$ generates $(\mathrm{PK}^{\mathcal{P}}_{U_{3,k}}, \mathrm{SK}^{\mathcal{P}}_{U_{3,k}})$, and applies for $\mathrm{Cert}(U_{3,k}) = \mathrm{Sign}_{\mathrm{SK}^{\mathcal{P}}_{U_{2,k'}}}(\mathrm{ID}_{U_{2,k'}}, \mathrm{ID}_{U_{3,k}}|\mathrm{PK}^{\mathcal{P}}_{U_{3,k}}, r_{U_{3,k}})$ from a second-level user $U_{2,k'}$ and $\mathrm{SK}^{\mathcal{I}}_{U_{3,k}} = \mathrm{SK}^{\mathcal{I}}(\mathrm{CH}(\mathbf{Cert}(U_{3,k}))) = \mathrm{GenSK}(\mathrm{CH}(\mathrm{Cert}(U_{3,k})), \mathrm{SK}^{\mathcal{I}}_{U_{2,k'}})$ from the PKG $U^{\mathcal{I}}_{2,k'}$. And $\mathrm{SK}^{\mathcal{H}}_{U_{3,k}} = \mathrm{SK}^{\mathcal{H}}(\mathbf{ID}_{U_{3,k}}) = \mathrm{GenHK}(\mathrm{ID}_{U_{3,k}}, \mathrm{SK}^{\mathcal{H}}_{U_{2,k'}})$ is distributed to $U^{\mathcal{I}}_{3,k}$, if $U_{3,k}$ is a subordinate CA.




doi: 10.1049/iet-ifs.2013.0552

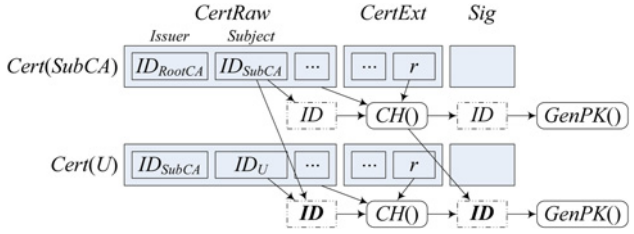


**Fig. 1** *Derivation of IBE public keys in hierarchical RIKE+*

– Any user follows the process above to apply for its certificate and escrowed key pair.

Fig. 1 shows an example about how to derive the IBE public key of $U$ from the certificate chain (Cert(SubCA), Cert($U$)). The inputs and the outputs of CH(·) and GenPK (·) are presented as arrows and the 'identity chains' of IBE and IBCH are presented as the dotted-line boxes, while $PM^{\mathcal{I}}$ and $PM^{\mathcal{M}}$ are not shown in this figure.

Hierarchical RIKE+ has both the features of distributed workload and centralised key escrow. The workload is distributed among subordinate CA and PKGs. Each subordinate CA is responsible for signing certificates as in traditional PKIs. A subordinate PKG is responsible for (a) generating the escrowed private keys of the users directly subordinated to its corresponding subordinate CA and (b) manipulating the chameleon hash value of each certificate, cooperatively with the CA. At the same time, hierarchical RIKE+ gives the root PKG the ability to recover the escrowed key pairs of all users. Given a user's certificate chain, the root PKG can regenerate the IBE private key.

### 3.6 Hierarchical RIKE+ with cross-certification

Theoretically, one hierarchical PKI (and then hierarchical RIKE+ ) can serve all users and relying-parties in the world. However, in the real world there are a number of root CAs. Users apply for certificates from different root CAs (directly or indirectly), and a relying-party is configured with a certificate trust list (CTL), a limited set of self-signed certificates. Relying-parties have different CTLs; thus, if a relying-party $R$ receives Cert($U$) signed by a root CA whose self-signed certificate is not in its CTL, $R$ cannot validate Cert($U$) and communicate with $U$ securely. Note that self-signed root certificates shall be delivered in reliable out-of-band means and be configured carefully [1], so a relying-party does not change its CTL rashly.

Cross-certification [6] helps a relying-party validate certificates signed by a root CA not in its CTL. As shown in Fig. 2, Cert($U$) is signed indirectly by $\text{RootCA}_1$, whereas the CTL of $R$ does not contain the self-signed certificate of $\text{RootCA}_1$. To help $R$ validate Cert($U$), $\text{RootCA}_2$ signs a cross-certificate $\text{CrsCert(SubCA)} = \text{Sign}_{\text{SK}^{\mathcal{P}}_{\text{RootCA}_2}}(\text{ID}_{\text{RootCA}_2}, \text{ID}_{\text{SubCA}}|\text{PK}^{\mathcal{P}}_{\text{SubCA}})$, so $R$ can validate the certificate chain (CrsCert(SubCA), Cert($U$)) by its CTL.

When cross-certification happens, a user will have multiple certificate chains. A cross-certificate is actually the same as a common certificate, except that the subject is a CA (called the 'subject CA'), which (a) already has a valid certificate and (b) has issued certificates for users. On encountering a certificate, a relying-party cannot distinguish whether it is a cross-certificate or not.

The coexisting multiple certificate chains lead to two problems, if the method in Section 3.5 is directly used to derive a user's IBE public key. Firstly, these certificate chains result in different escrowed key pairs, so all users subordinated to the subject CA of a cross-certificate have to apply for new IBE private keys after each cross-certification happens. Secondly, given a user, some of its IBE private keys are escrowed in the RIKE+ components managed by other organisations. Therefore the user's organisation cannot recover these private keys. For example, in Fig. 2, the private key that decrypts the encrypted data sent from $R$ to $U$ would be escrowed in $\text{RootPKG}_2$, which corresponds to $\text{RootCA}_2$. Note that $U$ is a user of $\text{RootCA}_1$, but $\text{RootPKG}_1$ cannot recover this private key.

The solution is to derive user's IBE public key always from the same certificate chain. We choose the one in which there is no cross-certificate and name it the 'primary certificate chain' of a user. The user's escrowed key pair is always generated based on its primary certificate chain. To achieve this aim, the cross-certificate carries (a) the IBE and the IBCH public parameters of the root PKG in the subject CA's primary certificate chain and (b) the primary certificate chain of the subject CA, more exactly, the chameleon hash values and the subject identities of certificates from the second-level CA to the subject CA, together called the 'ID-prefix' in this paper. The chameleon hash values are inputs for hierarchical IBE, and the subject identities are for hierarchical IBCH. The above information is embedded in the cross-certificate as an extension, called the 'RIKE-parameter' extension (see Section 4.3). Then, all relying-parties use the above information to derive the same IBE public key as that from the primary certificate chain.

For example, in Fig. 2 (Cert(SubCA), Cert($U$)) is the primary certificate chain of $U$, and then the cross-certificate CrsCert(SubCA) signed by $\text{RootCA}_2$ contains the IBE and the IBCH public parameters of $\text{RootPKG}_1$ and the ID-prefix of SubCA, that is, $\text{PM}^{\mathcal{I}}_{\text{RootPKG}_1}$, $\text{PM}^{\mathcal{H}}_{\text{RootPKG}_1}$, $\text{CH}_{\text{RootPKG}_1}(\textbf{Cert}(\text{SubCA}))$ and $\textbf{ID}_{\text{SubCA}}$. Therefore the IBE public key of $U$ can be always derived from $\text{PM}^{\mathcal{I}}_{\text{RootPKG}_1}$ and $\text{CH}_{\text{RootPKG}_1}(\textbf{Cert}(U)) = (\text{CH}_{\text{RootPKG}_1}(\textbf{Cert}(\text{SubCA})), \text{CH}_{\text{RootPKG}_1}(\text{Cert}(U)))$ where $\text{CH}_{\text{RootPKG}_1}(\text{Cert}(U)) = \text{CH}(\textbf{ID}_U, \text{PM}^{\mathcal{H}}_{\text{RootPKG}_1}, \text{CertRaw}_U, r_U)$, even if $R$ validates the certificate chain (CrsCert(SubCA), Cert($U$)) by Cert ($\text{RootCA}_2$).

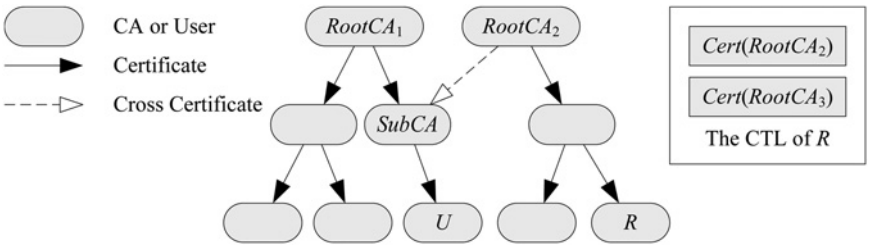


**Fig. 2** *Cross-certification and cross-certificate*

Algorithm 1 (Fig. 3) is used to reassemble the primary certificate chain of $U$ and derive its IBE public key, when a relying-party receives a non-primary certificate chain of $U$. During this process, if $\text{Cert}_i$ contains a RIKE-parameter extension (i.e. $\text{PM}^{\mathcal{I}}$, $\text{PM}^{\mathcal{H}}$ and the ID-prefix), then the IBE and IBCH public parameters are substituted by $\text{PM}^{\mathcal{I}}$ and $\text{PM}^{\mathcal{H}}$ and the ID-prefix is collected to reassemble the chameleon hash values and the subject identities of the primary certificate chain. Note that this algorithm works properly, even when $\mathbf{Cert}(U)$ is the primary certificate chain or there are multiple cross-certificates in $\mathbf{Cert}(U)$.

In Section 3.5, to guarantee the root PKG can recover all users' private keys, we required that the IBE public key parameters are only embedded in the root CA's self-signed certificate. However, following Algorithm 1 (Fig. 3) in the case of cross-certificates, the IBE public key parameters are obtained from a non-self-signed certificate. A new risk appears that a subordinate CA might maliciously embed different IBE public parameters when signing certificates for users. Note that a relying-party cannot distinguish such malicious certificates from a cross-certificate. Thus, the root PKG could not recover the IBE private keys of the users subordinated to the malicious CA, and the centralised key escrow is undermined.

Another certificate extension called the 'RIKE-parameter-lock' extension is proposed to avoid the above risk. Once this extension is set in a certificate, the subject CA and all its (directly and indirectly) subordinate CAs shall not issue (see Fig. 3) a certificate with different IBE or IBCH public parameters; otherwise, such a certificate is considered invalid. This process is also shown in Algorithm 1 (Fig. 3). These subordinate CAs are also deprived of the privilege to issue cross-certificates, because the potentially malicious CAs should not be granted this privilege.

### 3.7 Certificate renewal and revocation in hierarchical RIKE+

First of all, if the renewed or revoked certificate is held by a bottom-level user, the cases are the same as in the basic RIKE+ . Secondly, when a cross-certificate is renewed or revoked, the escrowed key pairs are not impacted because a primary certificate chain is not relative to any cross-certificate. Therefore the result is the same as that in traditional PKIs.

If the renewed or revoked certificate held by a subordinate CA is not a cross-certificate, then all certificates that it signed before become invalid automatically as in traditional PKIs. These users will apply for certificates from another subordinate CA. Therefore their non-escrowed key pairs can be kept unchanged if they are still considered secure; however, the escrowed key pairs have to be changed, unless

**Algorithm 1** The Derivation of IBE Public Key

**Input:** The certificate chain of $U$, $\boldsymbol{Cert}(U) = (Cert_2, Cert_3, \cdots, Cert_j)$;
The self-signed certificate in the CTL of $R$, $Cert_1$;
**Output:** The IBE public key of $U$, $PK_U^{\mathcal{I}}$;

$PM^{\mathcal{I},\mathcal{H}} = PMField(RikeParamExt(Cert_1))$;
// $PM^{\mathcal{I},\mathcal{H}}$ is set to the IBE and the IBCH public parameters in $Cert_1$
$\boldsymbol{ID}$ = null;
$PMLocked$ = false;
$IsrCert = Cert_1$;
**for** $(i = 2; i \leq j; i++)$ **do**
  **if** $Verify(IsrCert, Cert_i)$ **then** // $Cert_i$ is verified by $IsrCert$
    **if** $e = RikeParamExt(Cert_i)$ **then**
      // $Cert_i$ contains a RIKE-parameter extension $e$
      **if** $PMLocked$ **then**
        return null;
      **end if**
      $PM^{\mathcal{I},\mathcal{H}} = PMField(e)$;
      // $PM^{\mathcal{I},\mathcal{H}}$ is set to the IBE and the IBCH public parameters in $Cert_i$
      $\boldsymbol{ID} = IDPrefixField(e)$; // $\boldsymbol{ID}$ is set to the ID-prefix in $Cert_i$
    **else**
      // No RIKE-parameter extension in $Cert_i$
      $\boldsymbol{ID}.sid = Append(\boldsymbol{ID}.sid, Subject(Cert_i))$;
      $\boldsymbol{ID}.ch = Append(\boldsymbol{ID}.ch, CH(\boldsymbol{ID}.sid, PM^{\mathcal{H}}, Cert_i, r_i))$;
    **end if**
    **if** $f = RikeParamLocked(Cert_i)$ **then**
      // $Cert_i$ contains a RIKE-parameter-lock extension $f$
      $PMLocked = f$;
    **end if**
    $IsrCert = Cert_i$; // To verify the next certificate
  **else**
    return null;
  **end if**
**end for**
return $PK_U^{\mathcal{I}} = GenPK(\boldsymbol{ID}.ch, PM^{\mathcal{I}})$;

**Fig. 3** *Derivation of IBE public key*



*IET Inf. Secur.*, 2015, Vol. 9, Iss. 2, pp. 136–147
doi: 10.1049/iet-ifs.2013.0552

the chameleon hash values of the second subordinate CA' certificate chain are identical to the previous ones. It may happen when these two subordinate CAs correspond to a same subordinate PKG intentionally.

### 3.8 Features of RIKE+

*Effective certificate-based solution*: RIKE+ extends traditional PKIs by creatively leveraging the PKI certificates as revocable identities to support both key escrow and non-repudiation without conflict. Each user's escrowed key pair and non-escrow key pair is bound in only one certificate. These two public keys' validity is guaranteed simultaneously by validating the certificate. RIKE+ does not bring any extra communications and validations to obtain the public key for confidentiality. Then, it is an effective solution, especially in the environments where communication resources are limited.

*Inherent key escrow*: RIKE+ carries forward the inherent key escrow feature of IBE, in which all users' private keys are generated by the PKG. Therefore unlike the EA in the two-certificate solution that stores all users' private keys, the PKG in RIKE+ only stores the secret master key and achieves scalability when the user amount becomes enormous.

*Compatibility with PKIs*: RIKE+ is compatible with traditional PKIs to create, manage, distribute, use, store and revoke certificates. RIKE+ can be built on X.509 standards with newly designed certificate extensions. As a result, for the security services that do not need key escrow, RIKE+ and PKIs work with thorough interoperability and relying-parties may ignore the RIKE+ extensions if they do not support.

*Revocable identity*: Although RIKE+ borrows the key pair generation mechanism from IBE, RIKE+ does not suffer the revocation problem as IBE does. The IBE public key in RIKE+ is derived from (the chameleon hash value of) a certificate. The IBE public key is updated automatically by signing a new certificate, without changing the user's identity. The revocation of the IBE key pair is achieved by certificate revocation, for which there are abundant approaches. In this way, RIKE+ supports the 'revocable identity' of the user (not the real identity, but the 'identity' in the perspective of IBE).

*Flexible renewal and revocation*: Although the two key pairs of a user are simultaneously bound in one certificate, each of them is renewed independently. The IBE public key is derived from the chameleon hash value of the certificate, so the PKG holding the private hashing key can cooperate with the CA to control the relationship between the certificate and the IBE key pair, in the case of certificate renewal and revocation. The non-escrowed key pair explicitly bound in the certificate can also be kept unchanged or changed easily.

*Algorithm-independency*: RIKE+ combines the advantages of PKIs, IBE and IBCH, without imposing any additional requirements on these existing cryptographic algorithms. Any algorithm applicable in traditional PKIs, IBE and IBCH can be used in RIKE+. In a word, RIKE+ is an algorithm-independent framework and the algorithms can be adaptively chosen in different implementations and scenarios. Although the IBE and IBCH algorithms are not widely used, there are identity-based toolkits and products ready for use, such as pairing-based cryptography (PBC) [52], jPBC [53] and Voltage SecureMail [54].

### 3.9 Comparisons with other schemes

*RIKE+ against PKI*: The most straightforward way to satisfy the conflicting requirements of key escrow in PKIs is to generate two key pairs and sign two certificates for each user. A key-usage extension is embedded in each certificate to indicate the key pair's purpose. One of the two key pairs is generated and escrowed by the EA, and can be recovered by it when needed.

We compared RIKE+ with PKIs in three aspects as follows, showing that RIKE+ is more efficient than PKIs on both client-side and server-side. Firstly, to satisfy the conflicting requirements of key escrow, almost all the components in PKIs need to be scaled up. The CA shall have the ability to sign twice as many certificates as before. Resources for certificate distribution and revocation shall also increase 2-fold. In contrast, RIKE+ supports two key pairs bound in only one certificate, so no additional certificate distribution is needed.

Secondly, in PKIs both the two certificates of each user shall be transmitted to relying-parties. Therefore the communication cost doubles, which impedes this solution in bandwidth-limited applications. In contract, the burden for a relying-party to manage the certificates in its contact list is significantly reduced in RIKE+. The certificate for verifying messages is usually sent along with signed messages, and then it is configured automatically; but the certificate for encrypting messages is different: before encrypting messages, the relying-party manually queries the certificate and configures it on the cryptographic software or device. Thus, RIKE+ eliminates the manual query and configuration burden, because the public key to encrypt messages is derived from the certificate configured automatically.

Finally, the EA in PKIs needs to store all escrowed private keys in a well-protected repository. Besides currently valid keys, the EA also needs to store all historical keys, including out-of-date keys and revoked ones, which are necessary to decrypt the ciphertexts created when those keys were valid. As time goes on, more and more historical records will be accumulated. In RIKE+, the PKG only stores the IBE master key instead of all users' private keys. Since all the current and historical certificates are stored in plaintext, only the secret master key needs to be well protected, which is much simpler and more trustworthy than protecting a huge and accumulating set of escrowed private keys.

*RIKE+ against SE-PKI*: SE-PKIs enable PKIs to recover the users' private keys. The KRA generates its own key pair (the private key $SK_{KRA}$ and the public key $PK_{KRA}$) and publishes $PK_{KRA}$ as a public parameter for users to generate their key pairs. In this way, a trapdoor is placed in the key generation process. Thus, the user's private key can be calculated by the KRA using $SK_{KRA}$, and key escrow is achieved without storing all users' private keys.

Compared with SE-PKIs, RIKE+ is more suitable for large-scale environments and more compatible with legacy PKI systems. SE-PKIs escrow the users' private keys without differentiation, so it cannot solve the conflict between key escrow and non-repudiation. If SE-PKIs are adopted in traditional PKIs, the storage requirement of the EA is alleviated but each user still needs to hold two certificates and two key pairs, one of which is self-escrowed and the other is not. On the contrary, each RIKE+ user holds only one certificate. Moreover, in a system with a huge amount of users, the centralised KRA in SE-PKIs

**Table 1** Requirements of certificate and key escrow services

| User types | Services | | | Examples |
|---|---|---|---|---|
| | NA | CF | KE | |
| I | yes | yes | yes | users of a corporation email system |
| II | yes | yes | no | pretty good privacy individual users. The private key is used to both sign and decrypt messages |
| III | yes | no | – | timestamp authority servers. The private key is used to sign evidences that a document existed before a particular time |
| IV | no | yes | yes | Oracle transparent data encryption component [55]. The private key is used to decrypt data stored in a database |
| V | no | yes | no | users of the digital video broadcasting system [56]. The private key is used to decrypt video streams transmitted |

undergoes a heavy workload, whereas the distributed PKGs in hierarchical RIKE+ balance the workload and match the hierarchical legacy PKIs. We do not exclude the possibility of hierarchical SE-PKI algorithms in the future, but currently there is no such algorithm, to the best of our knowledge.

*RIKE+ against IBE*: As described above, the key idea of RIKE+ is the integration of PKIs and IBE by using the certificate as the 'identity' in IBE. IBE provides the favorite feature that the sender can obtain the recipient's public key from the recipient's identity, without extra online lookup. RIKE+ inherits this feature. The relying-party obtains the user's IBE public key from its certificate chain, eliminating any additional certificate to carry it.

The major obstacle for IBE to become a full-blown cbecrypto system is its lack of key revocation mechanism which is necessary in practice. In IBE, the key pair is difficult to revoke, because the public key is one-to-one bound with a user's identity and changing its identity brings unacceptable inconvenience. In contrast, RIKE+ supports key revocation without changing the user's identity.

RIKE+ borrows the IBE's spirit of using an arbitrary bit-string as a public key. Email address is usually accepted as an IBE identity, because it is already widely held and easy for human beings to remember and input. When RIKE + is deployed on existing PKIs, a relying-party has already held the PKI users' certificates and these certificates are used as IBE public keys automatically by the cryptographic software. Therefore, although the revocable identity in RIKE+ is much longer than the identity in IBE, its usability is not impaired.

## 4 Building RIKE+ on X.509 PKIs

In this section, we discuss how to build RIKE+ on the prevailing X.509 PKIs. Firstly, some special requirements of certificate and key escrow services are discussed, and RIKE+ is further improved to satisfy these special requirements as well as the conflicting key escrow requirements. Then, all newly designed certificate extensions for RIKE+ are defined in ASN.1 syntax. Finally, we provide guidelines on transferring existing X.509 PKIs to hierarchical RIKE+ .

### 4.1 Requirements of certificate and key escrow services

PKIs (and RIKE+ ) shall provide three categories of services: (NA) certificate services for non-repudiation and authentication, (CF) certificate services for confidentiality and (KE) key escrow services of the private key for confidentiality. As enumerated in Table 1, a user often needs all these services; however, there are some special users that only want part of them. Note that key escrow of the private key for non-repudiation and authentication is always prohibited.

In the two-certificate PKI solution, all types of users are served by one PKI system. For a special user of Types II, III, IV or V, it holds one certificate with a key-usage extension indicating the purpose of the corresponding key pair, and the EA escrows the private key if and only if it is needed. Moreover, a relying-party does not need to know which type a user belongs to, and simply follows the key-usage extension to determine whether the public key can be used or not.

### 4.2 Preventing the derivation of IBE public keys from certificates

When RIKE+ is built on PKIs compatibly, the following principles shall be insisted on. Firstly, different types of users are served by the same infrastructure. Secondly, on querying a user's certificates, relying-parties follow one uniform and compatible rule to obtain the public keys to encrypt and/or verify messages, without knowing which type the user belongs to. Therefore sometimes we need to prevent relying-parties from deriving IBE public keys to encrypt messages (instead, the public keys contained in the certificates shall be used) for the users of Types II and V. The case of Type III is similar, for the users do not need the certificate services for confidentiality.

We define the third extension for RIKE+ , the 'RIKE-flag' extension. Only when the RIKE-flag extension is present in a

**Table 2** Settings of the RIKE-flag and the key-usage extensions

| User types | Two-certificate PKI | | | | | | | | | | RIKE+ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Certificate #1 | | | | | Certificate #2 | | | | | Certificate #1 | | | | | |
| | *ds* | *nr* | *ke* | *de* | *ka* | *ds* | *nr* | *ke* | *de* | *ka* | *ds* | *nr* | *Ke* | *de* | *ka* | RIKE-flag |
| I | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | present |
| II | 1 | 1 | 1 | 1 | 1 | | | – | | | 1 | 1 | 1 | 1 | 1 | absent |
| III | 1 | 1 | 0 | 0 | 0 | | | – | | | 1 | 1 | 0 | 0 | 0 | absent |
| IV | | | – | | | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | present |
| V | | | – | | | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | absent |

ds: digital signature; nr: non-repudiation; ke: key encipherment; de: data encipherment and ka: key agreement




doi: 10.1049/iet-ifs.2013.0552

certificate, the escrowed IBE private key is generated for the user and relying-parties shall use the IBE public key derived from this certificate; otherwise, the IBE public key shall not be used. This RIKE-flag extension works complementarily with the key-usage extension to instruct relying-parties to obtain the proper public keys.

For different types of users, Table 2 shows the settings of the RIKE-flag and the key-usage extensions in the two-certificate PKI solution and RIKE+. Only five key-usage bits related to the users are listed; other purposes such as certificate and CRL signing are not shown. In the key-usage extension there are several bits, and each bit indicates one purpose of the public key contained in the certificate. If some bit is set ('1' in the table), then the corresponding purpose is allowed; otherwise, it is not allowed. The five purposes are divided into two categories, {*ds nr*} and {*ke de ka*}, for simplicity; and more fine-grained controls can be supported in a similar way.

Finally, a relying-party in RIKE+ follows the compatible rule as below to obtain the public key. It can be seen that, for a relying-party not supporting the RIKE+ extensions, this rule is actually the same as that in traditional PKIs.

*Public key for non-repudiation and authentication*: If the *ds*/*nr*-bits are set, the public key is contained in the certificate.
*Public key for confidentiality*: If the *ke*/*de*/*ka*-bits are set, the public key for confidentiality is contained in the certificate; or, if the *ke*/*de*/*ka*-bits are unset and the RIKE-flag extension is present, it is the IBE public key derived from the certificate.

### 4.3 Certificate extensions for RIKE+

We present the details of the newly designed certificate extensions for RIKE+. These extensions work for hierarchical RIKE+ finalised in Section 3.6; however, when RIKE+ is deployed and some of the implementation options in Section 4.5 are adopted, the extensions may be accordingly simplified and revised.

The descriptions in ASN.1 syntax are as follows (see Fig. 4).

The RIKE-parameter extension is used in two kinds of certificates: self-signed certificates and cross-certificates. The first five fields are mandatory. The fields ibeAlgorithm and ibchAlgorithm describe the IBE algorithm and the IBCH algorithm, respectively. The detailed public parameters are presented in the fields ibePublicParameters and ibchPublicParameters, and their structures depend on which algorithms are used. The field defaultChameleonRandom gives the 'default' chameleon random string of all certificates that follow this certificate in a certificate chain. That is, if the chameleon random string is not included in such a certificate as an extension, this default string is used to calculate its chameleon hash value. This field facilitates the smooth transfer of existing PKIs to RIKE+ (see Section 4.4). The optional field idPrefix, a sequence of chameleon hash values and subject identities of the certificates in the subject CA's primary certificate chain, exists only in cross-certificates.

```
-- The RIKE-parameter extension
RIKEParameter ::= SEQUENCE {
    ibeAlgorithm           OBJECT IDENTIFIER,
    ibePublicParameters    OCTET STRING,
    ibchAlgorithm          OBJECT IDENTIFIER,
    ibchPublicParameters   OCTET STRING,
    defaultChameleonRandom OCTET STRING,
    idPrefix               IDPrefix OPTIONAL }
IDPrefix ::= SEQUENCE SIZE (1..MAX) OF HashValueAndSID
HashValueAndSID ::= SEQUENCE {
    chameleonHashValue     OCTET STRING,
    subjectIdentity        OCTET STRING }

-- The RIKE-parameter-lock extension
RIKEParamLock ::= BOOLEAN

-- The RIKE-flag extension.
RIKEFlag ::= OCTET STRING
```

**Fig. 4** *Description in ASN.1 syntax*

The RIKE-parameter-lock extension RIKEParamLock is a boolean value indicating whether the RIKE+ parameters are locked or not. When this extension is set to true in a certificate, any other certificate following the certificate in a certificate chain is considered invalid if it has another extension with different RIKE+ parameters. If it is absent or set to false, different RIKE+ parameters are allowed in the following certificates, provided that there is not any other RIKE-parameter-lock extension.

The RIKE-flag extension, if it is present, indicates that an IBE public key is implicitly bound in this certificate and the chameleon random string is included simultaneously to calculate the chameleon hash value.

### 4.4 Transferring existing PKIs to RIKE+

Section 3.5 presents the steps to build RIKE+ from scratch. As a highly compatible certificate-based solution, RIKE+ can also be gradually built on existing X.509 PKIs by issuing certificates with RIKE+ extensions, without interrupting the certificate services. In this section, we discuss the issues about transferring 'existing' PKIs to RIKE+.

During the transfer, the root CA signs a new self-signed certificate to distribute the RIKE+ parameters, although the previous one has not expired yet. Other certificates, including the subordinate CAs' and the users', are replaced by new certificates with RIKE+ extensions only when they expire or are revoked. There is no extra certificate revocation or renewal because of the transfer to RIKE+, except the root CA's new certificate. In particular, the root PKG generates ($PK^{\mathcal{I}}$, $MK^{\mathcal{I}}$) and ($PM^{\mathcal{H}}$, $MK^{\mathcal{H}}$), and the root CA signs a certificate with a RIKE-parameter extension. Then, the PKGs of hierarchical IBE and IBCH are established. After receiving its IBE private key and private hashing key from the root PKG, each subordinate PKG works cooperatively with its corresponding CA. The subordinate CAs signs certificates with RIKE-flag extensions and the PKG generates IBE private keys for users, when their existing certificates expire or are revoked.

On the other hand, to derive IBE public keys, a relying-party needs to firstly accept the new self-signed certificate with the RIKE+ parameters. This can be done as key change-over [57], but not in any out-of-band means, because it is validated by the root CA's existing certificate that was delivered in reliable means. Then, the relying-parties can derive IBE public keys from the certificates with RIKE-flag extensions, based on the public parameters in the root CA's new certificate. When the IBE public key is derived by relying-parties, the subordinate CAs' certificates may be still ordinary PKI ones without RIKE+ extensions. In this case, the default chameleon random string in the root CA's certificate is used to calculate the chameleon hash values of these certificates.

### 4.5 Implementation options

When RIKE+ is deployed (especially on existing PKIs), there are implementation options that offer flexible choices for different scenarios.

– The PKG can be implemented as a component of the CA and managed by one department, especially for the key management within a corporation.
– An ordinary hash function is used, but not the chameleon hash functions, if (a) certificate renewal and revocation are rare and/or (b) the burden of IBE private key management is not an issue for the users.
– Only one non-hierarchical PKG serves for all users of a hierarchical PKI, so the derivation of IBE private keys will be simplified.
– The IBE and IBCH public parameters are embedded in a subordinate CA's certificate, and used to derive the IBE public keys of users subordinated to this CA. Another option is to directly embed these parameters in the bottom-level users' certificates, when the size of certificates does not matter. Then, the root CA's self-signed certificate is not renewed.
– Different IBE and IBCH public parameters are used to derive the IBE private keys. That is, the CAs choose different (root) PKGs with different public parameters embedded in certificates.

## 5 Conclusions

In this paper, we integrate PKIs, IBE and IBCH into a novel key management scheme called RIKE+, which leverages revocable identities to support key escrow in PKIs with flexibility. Each RIKE+ user holds two key pairs, one of which is escrowed inherently and the other is non-escrowed, bound in only one certificate. The escrowed key pair is usually for confidentiality, and the non-escrowed one is for authentication and non-repudiation. As an innovative key management infrastructure, RIKE+ satisfies the conflicting requirements of key escrow, and reduces the cost of managing key pairs and certificates. The escrowed IBE key pair is derived from the chameleon hash value of its PKI certificate, which enables the CAs and the PKGs to control the relationship between the certificate and the IBE key pair in the case of certificate renewal and revocation. This certificate-based scheme assembles the advantages of several cryptosystems and works compatibly with prevailing X.509 PKIs.

## 6 Acknowledgment

This work was partially supported by the National 973 Program of China under Grant 2013CB338001, the National Natural Science Foundation of China under Grant 61272479 and the Strategic Priority Research Program of Chinese Academy of Sciences under Grant XDA06010702. We would like to thank Dr. Peng Liu at the Pennsylvania State University, USA and Dr. Xianhui Lu at State Key Laboratory of Information Security, China for their helpful comments. A preliminary version of this paper appeared under the title 'RIKE: Using Revocable Identities to Support Key Escrow in PKIs' in 'Proceedings of the Tenth International Conference on Applied Cryptography and Network Security', Singapore, 2012.

## 7 References

1 Kohnfelder, L.: 'Towards a practical public-key cryptosystem'. Bachelor's thesis, Massachusetts Institute of Technology, 1978
2 China, Electronic signature law, 2004
3 Asia-Pacific Economic Cooperation: 'Guidelines for schemes to issue certificates capable of being used in cross jurisdiction e-commerce', 2004
4 European Telecommunications Standards Institute: 'Policy requirements for certification authorities issuing qualified certificates', 2000
5 European Union: 'Directive on a community framework for electronic signatures', 1999
6 Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., Polk, W.: IETF RFC 5280: Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile, 2008
7 Shamir, A.: 'Identity-based cryptosystems and signature schemes'. Crypto'84, Santa Barbara, CA, USA, 1985, pp. 47–53
8 Krawczyk, H., Rabin, T.: 'Chameleon signatures'. Seventh ISOC Network and Distributed System Security Symp., San Diego, CA, USA, 2000, pp. 42–53
9 International Telecommunication Union: 'X.509, information technology – open systems interconnection – the directory', Public-key and Attribute Certificate Frameworks, 2008
10 Entrust: Entrust authority digital certificate solution, 2012
11 RSA: 'the security division of EMC', RSA digital certificate solution, 2012
12 Brown, J., Gonzalez-Nieto, J., Boyd, C.: 'Efficient and secure self-escrowed public-key infrastructures'. Second ACM Symp. on Information, Computer and Communications Security, Singapore, 2007, pp. 284–294
13 Paillier, P., Yung, M.: 'Self-escrowed public-key infrastructures'. Second Int. Conf. on Information Security and Cryptology, Seoul, Korea, 1999, pp. 257–268
14 Boneh, D., Franklin, M.: 'Identity-based encryption from the Weil pairing'. Crypto'2001, Santa Barbara, CA, USA, 2001, pp. 213–229
15 Cocks, C.: 'An identity-based encryption scheme based on quadratic residues'. Eighth IMA Conf. on Cryptography and Coding, Cirencester, UK, 2001, pp. 360–363
16 Gentry, C., Silverberg, A.: 'Hierarchical ID-based cryptography'. Asia Crypt'2002, Queenstown, New Zealand, 2002, pp. 548–566
17 Horwitz, J., Lynn, B.: 'Toward hierarchical identity-based encryption'. Euro Crypt'2002, Amsterdam, the Netherlands, 2002, pp. 466–481
18 Geisler, M., Smart, N.: 'Distributing the key distribution centre in Sakai-Kasahara-based systems'. 12th IMA Conf. on Cryptography and Coding, Cirencester, UK, 2009, pp. 252–262
19 Kate, A., Goldberg, I.: 'Distributed private-key generators for identity-based cryptography'. Seventh Int. Conf. on Security and Cryptography for Networks, Amalfi, Italy, 2010, pp. 436–453
20 Al-Riyami, S., Paterson, K.: 'Certificateless public key cryptography'. AsiaCrypt'2003, Taipei, Taiwan, 2003, pp. 452–473
21 Gentry, C.: 'Certificate-based encryption and the certificate revocation problem'. EuroCrypt'2003, Warsaw, Poland, 2003, pp. 272–293
22 Goyal, V.: 'Reducing trust in the PKG in identity-based cryptosystems'. Crypto'2007, Santa Barbara, CA, USA, 2007, pp. 430–447
23 Goyal, V., Lu, S., Sahai, A., Waters, B.: 'Black-box accountable authority identity-based encryption'. 15th ACM Conf. on Computer and Communications Security, Alexandria, VA, USA, 2008, pp. 427–436
24 Boneh, D., Ding, X., Tsudik, G., Wong, M.: 'A method for fast revocation of public key certificates and security capabilities'. Tenth USENIX Security Symp., Washington, D.C., USA, 2001, pp. 297–308
25 Baek, J., Zheng, Y.: 'Identity-based threshold decryption'. Seventh Int. Workshop on Theory and Practice in Public Key Cryptography, Singapore, 2004, pp. 262–276
26 Libert, B., Quisquater, J.J.: 'Efficient revocation and threshold pairing based cryptosystems'. 22nd Annual ACM Symp. on Principles of Distributed Computing, Boston, MA, USA, 2003, pp. 163–171
27 Cooper, D.: 'A model of certificate revocation'. 15th Annual Computer Security Applications Conf., Phoenix, AZ, USA, 1999, pp. 256–264
28 Adams, C., Zuccherato, R.: 'A general, flexible approach to certificate revocation'. Entrust, 1998
29 Cooper, D.: 'A more efficient use of delta-CRLs'. 21st IEEE Symp. on Security and Privacy, Oakland, CA, USA, 2000, pp. 190–202
30 Myers, M., Ankney, R., Malpani, A., Galperin, S., Adams, C.: 'IETF RFC 2560: X.509 Internet public key infrastructure online certificate status protocol – OCSP, 1999
31 Micali, S.: 'NOVOMODO: scalable certificate validation and simplified PKI management'. First Annual PKI Workshop, Gaithersburg, MD, USA, 2002, pp. 15–25




doi: 10.1049/iet-ifs.2013.0552

32 Kocher, P.: 'On certificate revocation and validation'. Second Int. Conf. on Financial Cryptography, Anguilla, British West Indies, 1998, pp. 172–177
33 Wright, R., Lincoln, P., Millen, J.: 'Efficient fault-tolerant certificate revocation'. Seventh ACM Conf. on Computer and Communications Security, Athens, Greece, 2000, pp. 19–24
34 McDaniel, P., Jamin, S.: 'Windowed certificate revocation'. 19th IEEE INFOCOM, Tel-Aviv, Israel, 2000, pp. 1406–1414
35 Naor, M., Nissim, K.: 'Certificate revocation and certificate update'. Seventh USENIX Security Symp., San Antonio, TX, USA, 1998, pp. 217–228
36 Goyal, V.: 'Certificate revocation using fine grained certificate space partitioning'. 11th Int. Conf. on Financial Cryptography and Data Security, Scarborough, Trinidad and Tobago, 2007, pp. 247–259
37 Kong, J., Zerfos, P., Luo, H., Lu, S., Zhang, L.: 'Providing robust and ubiquitous security support for mobile ad-hoc networks'. Ninth IEEE Int. Conf. on Network Protocols, Riverside, CA, USA, 2001, pp. 251–260
38 Iliadis, J., Spinellis, D., Katsikas, S., Gritzalis, D., Preneel, B.: 'Evaluating certificate status information mechanisms'. Seventh ACM Conf. on Computer and Communications Security, Athens, Greece, 2000, pp. 1–8
39 Myers, M.: 'Revocation: options and challenges'. Third Int. Conf. on Financial Cryptography, Anguilla, British West Indies, 1999, pp. 165–171
40 Ding, X., Tsudik, G.: 'Simple identity-based cryptography with mediated RSA'. CT-RSA'2003, San Francisco, CA, USA, 2003, pp. 193–210
41 Khurana, H., Basney, J.: 'On the risks of IBE'. Int. Workshop on Applied PKC, Dalian, China, 2006, pp. 1–10
42 Callas, J.: 'Identity-based encryption with conventional public-key infrastructure'. Fourth Annual PKI Workshop, Gaithersburg, MD, USA, 2005, pp. 98–111
43 Chen, L., Harrison, K., Moss, A., Soldera, D., Smart, N.: 'Certification of public keys within an identity based system'. Fifth Int. Conf. on Information Security, Sao Paulo, Brazil, 2002, pp. 322–333
44 Price, G., Mitchell, C.: 'Interoperation between a conventional PKI and an ID-based infrastructure'. Second European PKI Workshop, Canterbury, UK, 2005, pp. 73–85
45 Ateniese, G., de Medeiros, B.: 'Identity-based chameleon hash and applications'. Eighth Int. Conf. on Financial Cryptography, Key West, FL, USA, 2004, pp. 164–180
46 Chen, X., Zhang, F., Susilo, W., Tian, H., Li, J., Kim, K.: 'Identity-based chameleon hash scheme without key exposure'. 15th Australasian Conf. on Information Security and Privacy, Syndey, Australia, 2010, pp. 200–215
47 Bao, F., Deng, R., Ding, X., Lai, J., Zhao, Y.: 'Hierarchical identity-based chameleon hash and its applications'. Ninth Int. Conf. on Applied Cryptography and Network Security, Nerja, Spain, 2011, pp. 201–219
48 Ateniese, G., Chou, D., de Medeiros, B., Tsudik, G.: 'Sanitizable signatures'. Tenth European Symp. on Research in Computer Security, Milan, Italy, 2005, pp. 159–177
49 Canard, S., Laguillaumie, F., Milhau, M.: 'Trapdoor sanitizable signatures and their application to content protection'. Sixth Int. Conf. on Applied Cryptography and Network Security, New York, NY, USA, 2008, pp. 258–276
50 Zhang, N., Lin, J., Jing, J., Gao, N.: 'RIKE: using revocable identities to support key escrow in PKIs'. Tenth Int. Conf. on Applied Cryptography and Network Security, Singapore, 2012, pp. 48–65
51 National Institute of Standards and Technology: 'FIPS PUB 180-4', Secure hash standard, 2012
52 Lynn, B.: 'The pairing-based cryptography library', 2014, Available at http://crypto.stanford.edu/pbc/
53 Lynn, B.: 'The java pairing-based cryptography library', 2014, Available at http://libeccio.dia.unisa.it/projects/jpbc/index.html
54 Voltage Security, Inc: SecureMail, 2014
55 Oracle: 'Database advanced security administrator's guide (v11.2)', 2013
56 Cutts, D.: 'DVB conditional access', *Electron. Commun. Eng. J.*, 1997, **9**, (1), pp. 21–27
57 Chokhani, S., Ford, W., Sabett, R., Merrill, C., Wu, S.: 'IETF RFC 3647'. Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework', 2003