An Efficient and Secure Delegated Multi-authentication Protocol for Mobile Data Owners in Cloud

Lifei Wei[1,2], Lei Zhang[1(✉)], Kai Zhang[3], and Mianxiong Dong[4]

[1] College of Information Technology,
Shanghai Ocean University, Shanghai 201306, China
Lzhang@shou.edu.cn
[2] State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications, Beijing 100081, China
[3] Department of Computer Science and Technology, East China Normal University,
Shanghai 200241, China
[4] Department of Information and Electronic Engineering,
Muroran Institute of Technology, Muroran, Japan

Abstract. Due to plenty of cloud-based applications emerging and booming recently, data owners always store their data in cloud and share them to data consumers through cloud servers. For security equirements, data owners are often asked to provide authentication tags to the corresponding data. Data consumers obtain the authenticated data from the cloud and expect the computation on the authenticated data. However, it is impractical for the mobile data owners to be online all the time and provide the authenticated computing results according to various data consumers' request. To tackle this issue, we propose an efficient and secure delegated multi-authentication protocol for mobile data owners in cloud, which enables the mobile data owners to conditionally delegate signing right to specified cloud servers without exposing the secret signing keys. The cloud servers provide the authentication services when data owners are not available. The security is built on an identity-based multi-proxy signature (IBMPS) scheme, which depends on the cubic residue assumption, equaling to the factorization assumption. Furthermore, our protocol is efficient compared to the pairing based schemes and the overhead is almost independent of the number of cloud servers.

Keywords: Authenticated computing · Mobile data owner · Delegated multi-authentication · Multi-proxy signature · Cloud computing

This work was supported by the National Natural Science Foundation of China (No. 61402282), Shanghai Sailing Program (No. 14YF1410400), Youth Scholars of Shanghai Higher Education Institutions (No. ZZHY14025), Open Foundation of State key Laboratory of Networking and Switching Technology of BUPT (No. SKLNST-2013-1-12), ECNU Fund for Graduate Student's Scientific Research, Innovation, and Practice (No. YJSKC2015-30) and JSPS KAKENHI (No. 26730056, 15K15976), JSPS A3 Foresight Program.

K. Xu and J. Zhu (Eds.): WASA 2015, LNCS 9204, pp. 612–622, 2015.
DOI: 10.1007/978-3-319-21837-3_60

1 Introduction

With the development of cloud computing techniques, a number of cloud-based applications have been emerging and booming recently [1,2]. In particular, applications focusing on the outsourced computation in cloud have been paid much attention due to the powerful computation and storage capability of cloud. Different from the conventional computation in which the users have fully controlled their data, cloud computing enables servers to manage the physical machines and the data on these workstations while the cloud users only retain the control over the virtual machines [3]. As a result, the outsourced computation performed on those untrustworthy cloud has been restricting the further progress of the cloud computing.

In the scenarios, the data owners store their own collected data into cloud servers and share with the data consumers. For the security requirement, the data owners need to provide an authentication tag with the corresponding data to show the valid of the data to the cloud. Related work tried to add authentication tags [4,5] along with the data in order to keep the integrity and authentication in the cloud servers. Signature often provides authenticity and non-repudiation in the communication networks. Data consumers not only want to get the authenticated data from the cloud, but also need to compute on the authenticated data [6] with the help of the powerful computation capability of the cloud servers. In some scenarios, the data owners can not be online all the time such as 24 hours a day due to their mobile characteristics [7]. Moreover, they can not afford the data computation overhead for a large number of the computation request. As a result, they need to delegate the authentication rights (signing rights) to the cloud [8].

There are some straightforward solutions to overcome this obstacle. One solution is that if the data consumers just request the original data, the data owner could pre-store all the authenticated tags in the cloud servers. However, the data owner can not pre-store all kinds of the authenticated data in the cloud before since he/she can not foresee the different data computation requests. In most cases, the demand is that the stored data need to be calculated or taken transforms from the original data according to the data user's requests. Another solution is that all the request must be sent to data owner for further authentication, which leads to lots of computational overhead for the data owners and considerable delay by the offline of the data owners.

To cut down the communication cost and delay, another solution is that the data owners hand out their secret signing keys to the cloud and share them by cloud servers. In this case, the authentication service could be accompanied without the help of the data owners since the cloud servers could finish the data authentication dependently, which achieves the outsourced authentication. However, some of the remote cloud servers may be compromised by the attackers and the secret keys may be revealed, which results in a worse consequence since the data authentication service might be out of control and be misused by malicious cloud servers.

When it comes to the data computation, ordinary signature may not be well available since it violates the existential unforgeable property of signature.

Some essential variations of ordinary digital signature schemes have been proposed. Homomorphic signature [9,10] seems a feasible solution to this problem since homomorphic signature achieves either additional homomorphic or multiply homomorphic. However, for some complicated operation such as data comparison results and SQL based query [11], the homomorphic signature can not achieve the target. In addition, homomorphic signature always bring in considerable extra overhead to the authenticated data.

In this work, we aim to address the above challenges and propose an efficient and secure data delegated multi-authentication protocol for mobile data owners in cloud. Our proposed protocol is feathered by delegated authentication to a number of cloud servers and high efficiency on both the data owner side and the data consumer side. Thanks to the novel identity based multi-proxy signature scheme, the data owners can delegate the signing rights to a number of cloud servers which need to collaborate to carry out the authentication tags generation and aggregation instead of the data owners. Moreover, our protocol allows the data owners to conditionally delegate to the cloud servers and the data consumers could easily verify the authentication tags and alert to the system authority when they find the misusing of the cloud servers. Last but not least, our protocol is quite efficient compared to pairing based protocols.

To the best of our knowledge, this work is the first effort towards considering delegated authentication among mobile data owners and cloud servers in order to provide authentication service for data consumers. The contributions can be briefly summarized:

- Firstly, we propose an identity-based multi-proxy signature scheme which is proved secure under the hardness of integer factorization assumption in the random oracle model.
- Secondly, we propose a secure delegated multi-authentication protocol, based on our efficient multi-proxy signature schemes, to provide authenticated computing results for mobile data owner in cloud environment.
- Thirdly, in contrast to previous solutions, our protocol achieves a conditional delegation with traceability that is malicious cloud servers can be caught and accused of misusing by data consumers.

The rest of the paper is organized as follows. Section 2 describes our system architecture, security models, designed goals, and formal definition of framework for identity-based multi-proxy signature scheme. Section 3 introduces some necessary mathematic preliminaries. In Sect. 4, we propose a detail construction, which is proven secure in the random oracle model in Sect. 5. Section 6 sees the performance comparison of the related work. Finally, Sect. 7 makes a conclusion.

2 System Architecture and Design Goals

2.1 System Architecture

As shown in Fig. 1, we consider a general cloud system, which is composed of four major entities: *system authority*, *cloud servers*, *data owners* and *data consumers*.

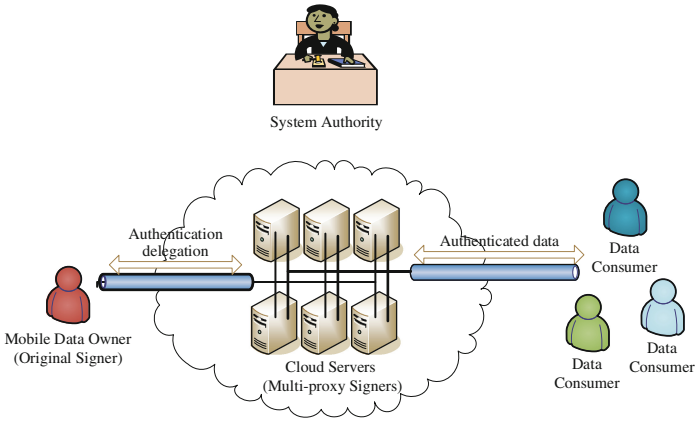

Fig. 1. A delegated multi-server authentication architecture in Cloud.

- **System Authority (SA).** In our scenario, the system authority is the administrator of the system and independent with other roles, which is in charge of system initialization and secret key generation of each parties in the system. We also assume that the system authority can not be compromised by the adversary.
- **Cloud Servers (CSs).** The cloud system is constituted of a number of cloud servers, named as $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_n$, with plenty of computation resources and storage resources and controlled by cloud service provider, like the general cloud model. The cloud servers provide the data storage and data computation service and response to the data consumers' requests.
- **Data Owners (DOs).** The mobile data owners obtain the data through the wireless collecting devices and store them into the cloud when the data owners encounter to connect to the networks. We assume that the data owners are not always online (ex. 24-hours) due to the mobile characteristic. For the security requirement, the data owners add the tags to the data. The tags should be unforgeable achieved by digital signature. Thus, the data owners upload the data as well as the tags.
- **Data Consumers (DCs).** Each data consumers queries the data and asks for the data computation from cloud servers. For the security requirement, the data consumers need get the data and authenticated tags for keeping the authenticity of the source.

2.2 Design Goals

The protocol is expected to achieve the following goals:

- **Authentication Delegation.** The data owners can conditionally delegate the signing rights to cloud servers without leaking its own secret keys.
- **Data Authentication.** The data consumers can get the authenticated data from the cloud servers which can not forge such authentication tags.

- **Security.** The delegation should be secure that the cloud servers can not forge the invalid authentication tags without being caught.
- **Efficiency.** The computational overhead of the scheme should be at a low level and be better to meet the minimum.

2.3 Security Models

We assume the following factors that may impact data authentication. Firstly, the adversary knows the identities of both the data owner and cloud servers as well as the other public information. Secondly, some of the cloud servers might be compromised by the adversary. To maximize harm to the system, we model an extreme case in which the adversary compromises totally $n-1$ cloud servers and works against one single honest cloud server to forge invalid authentication tags and find out the secret key of the data owners. In addition, we assume that there is no secure channels between cloud servers and data owners.

2.4 Framework of IBMPS

In an IBMPS scheme [12,13], there exists an original signer $\mathcal{O}$ and a group of proxy signers $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n$ who provide the signing service delegated by original signer. We denote the identity of the original signer as $ID_{\mathcal{O}}$ and the proxy signer $\mathcal{P}_i$ as $ID_{\mathcal{P}_i}$, respectively.

Definition 1 (Framework). *Our delegated authentication protocol is a collection of the following algorithm based on an identity-based multi-proxy signature scheme:* ***Setup, Extract, Sign, Verify, MPGen, MPSign, and MPVerify****.*

3 Preliminaries

To further explain our schemes, we introduce the necessary concepts in our scheme construction and security proof.

3.1 Mathematical Preliminaries

We introduce the definition of *cubic residue* and the following lemma [14,15].

Definition 2. *For an integer* $N \equiv 1 \pmod 3$, $a \in \mathbb{Z}_N^*$ *is a cubic residue modulo* N *if* $X^3 \equiv a \pmod N$ *for some* $X \in \mathbb{Z}_N^*$.

Lemma 1. *The integer factoring problem becomes easy if we find two different cubic roots* s_1, s_2 *of a cubic residue* a *modulo* N, *where* $s_1^3 \equiv s_2^3 \equiv a \pmod N$ *and* $s_1 \neq \pm s_2 \pmod N$.

Note that the 3-th root pairs satisfying $s_1 \equiv \pm s_2 \pmod N$ do not help to factor N.

4 Protocol Construction

We construct our delegated authentication protocol based on our designed identity-based proxy multi-signature scheme. Our protocol consists of five phases: *System initialization*, *Tag generation and verification*, *Authentication delegation*, *Multi-server authentication*, *Verification*.

4.1 System Initialization

The system initialization phase is to set up globe parameters and generate each entity's secret key.

Setup. Taking the security parameters λ, this algorithm can be done as follows.

(1). **SA** generates two random secure prime numbers p, q such that $p \equiv 2 \pmod 3$ and $q \equiv 4 \pmod 9$ or $q \equiv 7 \pmod 9$ and computes their product $N = p \cdot q$.
(2). **SA** computes $\eta = [q-1 \pmod 9]/3$, $\lambda = \eta \pmod 2 + 1$, and $\beta = (q-1)/3$.
(3). **SA** chooses a non-cubic residue a such that $\left(\frac{a}{q}\right) = -1$ and sets $\xi = a^{\eta\beta} \pmod q$.
(4). **SA** picks up five hash functions H_1, H_2, H_3, H_4, H_5, where $H_1, H_4 : \{0,1\}^* \to \mathbb{Z}_N^*$, $H_2, H_3, H_5 : \{0,1\}^* \to \{0,1\}^\ell$.

Finally, **SA** finishes **Setup** algorithm by outputting master secret keys $msk = (p, q, \beta)$ and public parameter $pp = (N, H_1, H_2, H_3, H_4, H_5, a, \eta, \lambda)$. **SA** keeps msk secretly.

Key Generation. For each entity in the system , on input the identity ID, **SA** computes the corresponding private key d as follows.

(1). **SA** computes $\omega = h_1(ID)^{\lambda\beta} \pmod q$.
(2). **SA** computes

$$c = \begin{cases} 0, \text{if} \omega = 1 \\ 1, \text{if} \omega = \xi \\ 2, \text{if} \omega = \xi^2 \end{cases}$$

and sets $H(ID) = a^c \cdot H_1(ID) \pmod N$. It is easy to show that $H(ID) \in \mathbb{CR}_N$.
(3). **SA** computes

$$d_{ID} = H(ID)^{\frac{2^{\eta-1}(p-1)(q-1)-3}{9}} \pmod N \tag{1}$$

and sends the secret key d_{ID} to entity with a tag c through a secure channel. Note that $d_{ID}^3 \cdot H(ID) \equiv 1 \pmod N$. After that, every entity can compute $H(ID)$ since the ID and c are in public.

4.2 Tag Generation and Verification

Tag Generation. The data owner could generate the authentication tags for the data m as follows.

1. The data owner $\mathcal{O}$ chooses a random number $r \in \mathbb{Z}_N^*$ and computes $R = r^3 \pmod N$.
2. $\mathcal{O}$ obtains the hash value $h_2 = H_2(m||R)$ and computes $V = r \cdot d_o^{h_2}$ using its secret key d_o.

Finally, the authentication tag $tag = (R, V)$ combined with the data m are sent to the cloud servers $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n$.

Tag Verification. When the cloud servers receives the data m and the corresponding authentication tag, they should verify the tag by checking the Eq. (2).

$$V^3 \cdot H(ID_o)^{h_2} \stackrel{?}{=} R, \tag{2}$$

where $h_2 = H_2(m||R)$. The cloud server accepts the data if it has a valid authentication tag. Otherwise, it requests a new valid authentication tag from data owners, or terminates the protocol.

4.3 Authentication Delegation

To delegate the signing capability to the cloud servers as proxy signers $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n$, the data owners, as a original signer $\mathcal{O}$, does the following *three* steps to make the valid warrant ω, which specifies the necessary delegation details, such as the identity of the data owners and the cloud servers (the original signers and the proxy signer), the expiry time of delegation, etc.

Delegation Generation. The data owner confirms the warrant ω by signing it.

1. The data owner $\mathcal{O}$ chooses a random number $r_o \in \mathbb{Z}_N^*$ and computes $R_o = r_o^3$.
2. $\mathcal{O}$ obtains the hash value $h_2 = H_2(\omega||R_o)$ and computes $V_o = r_o \cdot d_o^{h_2}$ using its secret key d_o.
3. $\mathcal{O}$ broadcasts the delegation information $\sigma = (\omega, R_o, V_o)$ to the cloud servers $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n$.

Delegation Verification. After receiving the delegation requests, every cloud server $\mathcal{P}_i$ first confirms $\sigma = (\omega, R_o, V_o)$ by checking the Eq. (3).

$$V_o^3 \cdot H(ID_o)^{h_2} \stackrel{?}{=} R_o, \tag{3}$$

where $h_2 = H_2(\omega||R_o)$. Every cloud server accepts the delegation if it is a valid delegation; otherwise, it requests a new valid delegation from $\mathcal{O}$, or terminates the protocol.

Proxy Signing Key Generation. After accepting the delegations, every cloud server computes its new proxy signer key by Eq. (4).

$$sk_{p_i} = V_o \cdot d_{p_i}^{h_3}, \tag{4}$$

where $h_3 = H_3(\omega||R_o)$.

4.4 Multi-server Authentication Generation

Each cloud server $\mathcal{P}_i$ signs the message m under ω on behalf of the data owner $\mathcal{O}$ as follows:

(1). $\mathcal{P}_i$ chooses a random number $r_i \in \mathbb{Z}_N$ and computes $R_i = r_i^3$. $\mathcal{P}_i$ computes $t_i = H_4(R_i)$ and broadcasts t_i to other cloud servers.
(2). After receiving t_i, $\mathcal{P}_i$ broadcasts R_i to other cloud servers.
(3). After receiving R_i, $\mathcal{P}_i$ check each $t_i \stackrel{?}{=} H_4(R_i)$. If any of above equation does not satisfy, the algorithm aborts. $\mathcal{P}_i$ computes $R = \prod_{i=1}^{n} R_i$ and gets the hash $h_5 = H_5(\omega||m||R)$ and computes $v_i = r_i \cdot sk_{p_i}^{h_5}$ and broadcasts v_i to other proxy signers.
(4). After receiving v_i from other cloud servers, $\mathcal{P}_i$ first checks each Eq. (5).

$$v_i^3 \cdot H(ID_{p_i})^{h_3 h_5} \stackrel{?}{=} R_i \cdot V_o^{h_5}, \tag{5}$$

where $h_3 = H_3(\omega||R_o)$ and $h_5 = H_5(\omega||m||R)$. If any of above equation does not satisfy, the algorithm abort. After that, $\mathcal{P}_i$ computes $V = \prod_{i=1}^{n} v_i$.

Once all partial signatures are correct, the multi-server authentication of message m can be generated as $p\sigma = (\omega, V_o, V, R)$

4.5 Authentication Verification

After receiving the message m and the multi-authentication tag $p\sigma = (\omega, V_o, V, R)$, the data consumer operates as follows:

(1). checks whether or not the message m conforms to the warrant ω.
(2). checks whether or not the n cloud servers $\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_n$ are authorized by the data owner $\mathcal{O}$ in the warrant ω.
(3). accepts the multi-server authentication tag if and only of the following Eq. (6) holds:

$$V^3 \cdot H(ID_o)^{h_2 h_5} \cdot \prod_{i=1}^{n} H(ID_{p_i})^{h_3 h_5} \stackrel{?}{=} R \cdot V_o^{h_5} \tag{6}$$

where $h_3 = H_3(\omega||R_o)$ and $h_5 = H_5(\omega||m||R)$.

5 Security Analysis

5.1 Correctness

The delegated multi-authentication protocol is correct since

$$V \equiv \prod_{i=1}^{n} v_i \equiv \prod_{i=1}^{n} r_i \cdot sk_{p_i}^{h_5} \equiv \prod_{i=1}^{n} r_i \cdot (V_o \cdot d_{p_i}^{h_3})^{h_5}$$
$$\equiv \prod_{i=1}^{n} r_i \cdot (r_o \cdot d_o^{h_2})^{h_5} \cdot d_{p_i}^{h_3 h_5} \equiv \prod_{i=1}^{n} r_i \cdot r_o^{h_5} \cdot d_o^{h_2 h_5} \cdot d_{p_i}^{h_3 h_5} \pmod{N}. \quad (7)$$

Thus, we have

$$V^3 \equiv \prod_{i=1}^{n} R_i \cdot V_o^{h_5} \cdot (d_o^3)^{h_2 h_5} \cdot (d_{p_i}^3)^{h_3 h_5} \equiv R \cdot V_o^{h_5} \cdot (d_o^3)^{h_2 h_5} \cdot \prod_{i=1}^{n} (d_{p_i}^3)^{h_3 h_5}$$
$$\equiv R \cdot V_o^{h_5} \cdot (H(ID_o)^{-1})^{h_2 h_5} \cdot \prod_{i=1}^{n} (H(ID_{p_i})^{-1})^{h_3 h_5} \pmod{N}. \quad (8)$$

From Eq. (8), we can find out the correctness proof since

$$V^3 \cdot H(ID_o)^{h_2 h_5} \cdot \prod_{i=1}^{n} H(ID_{p_i})^{h_3 h_5} \equiv R \cdot V_o^{h_5} \pmod{N}. \quad (9)$$

5.2 Security Proof

Theorem 1 (Main Theorem). *Our scheme is* $(t, q_E, q_S, q_H, n, \epsilon)$*-secure against existential forgery on the adaptively chosen message attack and chosen identity attack in the random oracle model if the factoring problem is hard.*

Due to space, we refer the reader to the full version for the proof of this theorem.

6 Performance Comparison

In this section, we compare our protocol with related work, which are provable security based on different hardness assumptions in the oracle model. From [16], we can find the major operation times for one bilinear pairing operation ($P, 20.01\ ms$), map-to-point hash operation ($H, 3.04\ ms$), modular-exponentiation ($E, 11.20\ ms$), normal scale multiplication ($M, 0.83\ ms$), pairing based scalar multiplication ($Psm, 6.38\ ms$). Though every large integer multiplication takes little time, considering the number of cloud servers n, it is necessary to add a reasonable time in **MPSign** and **MPVerify** algorithm.

From the Table 1, it is obvious that our protocol is quite efficient since we do not use the bilinear pairing technique compared to [12,13,17].

Table 1. Comparison of computation cost and running time.

Schemec	MPGen	Time	MPSign	Time	MPVerify	Time	Assumption
LC05 [17]	$3P+6Psm$	103.71	$3P+1E+3Psm$	84.57	$4P+1E+3Psm$	78.19	CDH
CC09 [12]	$3P+2H+3Psm$	85.34	$P+3H+1E+2Psm$	127.39	$4P+2H+3Psm$	99.0	CDH
SP15 [13]	$2P+5Psm$	71.98	$P+5Psm$	71.98	$2P+4Psm$	65.6	CD
Ours	$3E+7M$	39.41	$3E+(n+6)M$	38.58*	$3E+(n+4)M$	36.92*	Factorization

7 Conclusion

In this paper, we have built an efficient and secure delegated multi-authentication protocol in cloud, which enables the mobile data owners to conditionally delegate the signing right to multiple cloud servers without exposing the signing keys. The security of our protocol is based on an identity-based multi-proxy signature scheme, which depends on the factorization assumption.

In our future work, we consider three aspects to improve efficiency and security. Firstly, our protocol needs three round interactions in the multi-proxy signature generation, which leads to considerable communication overhead. We suggest to reduce the interactive rounds by constructing commitment schemes under cubic residue assumptions like that in [18]. Secondly, it is possible to propose general constructions of identity based signature schemes under cubic residues and even higher residues such as 2^k-th power residues [19]. Thirdly, we suggest to consider reliability of the cloud system and design a threshold multi-proxy signature scheme which allows authentication service not stopping even though a part of the cloud servers have been compromised.

References

1. Dong, M., Li, H., Ota, K., Zhu, H.: Hvsto: efficient privacy preserving hybrid storage in cloud data center. In: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS 2014), pp. 529–534 (2014)
2. Dong, M., Li, H., Ota, K., Yang, L.T., Zhu, H.: Multicloud-based evacuation services for emergency management. IEEE Cloud Comput. **1**(4), 50–59 (2014)
3. Wei, L., Zhu, H., Cao, Z., Dong, X., Jia, W., Chen, Y., Vasilakos, A.V.: Security and privacy for storage and computation in cloud computing. Inf. Sci. **258**, 371–386 (2014)
4. Wang, C., Chow, S.S., Wang, Q., Ren, K., Lou, W.: Privacy-preserving public auditing for secure cloud storage. IEEE Trans. Comput. **62**(2), 362–375 (2013)
5. Yuan, J., Yu, S.: Efficient public integrity checking for cloud data sharing with multi-user modification. In: INFOCOM 2014, pp. 2121–2129 (2014)
6. Ahn, J.H., Boneh, D., Camenisch, J., Hohenberger, S., shelat, A., Waters, B.: Computing on authenticated data. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 1–20. Springer, Heidelberg (2012)
7. Jia, W., Zhu, H., Cao, Z., Wei, L., Lin, X.: SDSM: a secure data service mechanism in mobile cloud computing. In: IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS 2011), pp. 1060–1065 (2011)

8. Boldyreva, A., Palacio, A., Warinschi, B.: Secure proxy signature schemes for delegation of signing rights. J. Cryptology **25**(1), 57–115 (2012)
9. Johnson, R., Molnar, D., Song, D., Wagner, D.: Homomorphic signature schemes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 244–262. Springer, Heidelberg (2002)
10. Wang, Z., Sun, G., Chen, D.: A new definition of homomorphic signature for identity management in mobile cloud computing. J. Comput. Syst. Sci. **80**(3), 546–553 (2014)
11. Yuan, J., Yu, S.: Flexible and publicly verifiable aggregation query for outsourced databases in cloud. In: IEEE CNS 2013, pp. 520–524 (2013)
12. Cao, F., Cao, Z.: A secure identity-based multi-proxy signature scheme. Comput. Electr. Eng. **35**(1), 86–95 (2009)
13. Sahu, R.A., Padhye, S.: Provable secure identity-based multi-proxy signature scheme. Int. J. Commun. Syst. **28**(3), 497–512 (2015)
14. Shoup, V.: A Computational Introduction to Number Theory and Algebra. Cambridge University Press, Cambridge (2009)
15. Wang, Z., Wang, L., Zheng, S., Yang, Y., Hu, Z.: Provably secure and efficient identity-based signature scheme based on cubic residues. Int. J. Netw. Secur. **14**(1), 33–38 (2012)
16. He, D., Chen, J., Zhang, R.: An efficient and provably-secure certificateless signature scheme without bilinear pairings. Int. J. Commun. Syst. **25**(11), 1432–1442 (2012)
17. Li, X., Chen, K.: Id-based multi-proxy signature, proxy multi-signature and multi-proxy multi-signature schemes from bilinear pairings. Appl. Math. Comput. **169**(1), 437–450 (2005)
18. Bagherzandi, A., Jarecki, S.: Identity-based aggregate and multi-signature schemes based on RSA. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 480–498. Springer, Heidelberg (2010)
19. Joye, M., Libert, B.: Efficient cryptosystems from 2^k-th power residue symbols. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 76–92. Springer, Heidelberg (2013)