Non-conventional digital signatures and their implementations – A review

David Arroyo[1], Jesus Diaz[2], and Francisco B. Rodriguez[3]

Grupo de Neurocomputacion Biologica, Departamento de Ingenieria Informatica,
Escuela Politecnica Superior, Universidad Autonoma de Madrid
[1,2,3]{david.arroyo,j.diaz,f.rodriguez}@uam.es

Abstract. The current technological scenario determines a profileration of trust domains, which are usually defined by validating the digital identity linked to each user. This validation entails critical assumptions about the way users' privacy is handled, and this calls for new methods to construct and treat digital identities. Considering cryptography, identity management has been constructed and managed through conventional digital signatures. Nowadays, new types of digital signatures are required, and this transition should be guided by rigorous evaluation of the theoretical basis, but also by the selection of properly verified software means. This latter point is the core of this paper. We analyse the main non-conventional digital signatures that could endorse an adequate tradeoff betweeen security and privacy. This discussion is focused on practical software solutions that are already implemented and available online. The goal is to help security system designers to discern identity management functionalities through standard cryptographic software libraries.

1 Introduction

Identity management is one of the most challenging matters in communication networks. Although it is possible to reach a conclusion about physical identity, it is not so easy to establish a relationship between a physical identity and a digital identity. Cryptography provides a means to associate a digital identity to an user on the grounds of asymmetric cryptography. In this guise, a pair of keys (the public and the private keys) are generated such that each component of this pair is connected to the other, but it is not computationally possible to use one of them to obtain the other. If one uses her private key, then the encrypted information can be only decrypted by means of the related public key. Correspondingly, if an user sends a (*hashed*) message and the message encrypted with her private key, the previous procedure leads to a verification of both the integrity of the message and the correctness of the private key. Loosely speaking, this procedure depicts the way the basic digital signatures are generated. Digital signatures are *publicly verifiable* and *transferable* cryptographic primitives, and they also have the property of non-repudiation [33, Chapter 1]. These are the main properties of what we can call *conventional digital signatures*.

The combination of non-repudiation and transferability are not always required and they are even replaced by a *deniability* commitment. For example, this is the case of scenarios as e-voting and e-coin where a user is interested on proving the authenticity of a piece of information to a certain receiver, but she wants to prevent the sender from proving this fact to other parties. Besides, in other situations it is demanded to design a signature scheme where a message or a document can be signed by multiple users (for example, if we are dealing with a committee that has to endorse as a whole a document). As it is underlined in [12], there are more than 60 digital signatures models. The classification of those models is not an easy task, since the discern between the underlying properties is far from a straightforward operation. Regarding the implementation of the different digital signatures models, this fact incorporates an additional risk. Certainly, one of the major problems in the design of security software is drawn by non-complete description of basic assumptions and their implications in concrete application contexts. This task should be based on the identification of standard products offering the functionalities that our design demands [19]. Indeed, software standards are products that have been carefully evaluated, which implies that we can assume a reasonable certainty about their reliability.

In this paper we review the most relevant families of *non-conventional digital signatures*, being the core of our effort to identify software libraries sustaining each one of the considered schemes. We discuss the main properties and applications of the considered digital signatures. In some cases we show that there does not exist well-founded and properly evaluated software libraries. However, it is possible to establish a set of basic cryptographic primitives and software libraries as a means to finally implement the referred digital signatures.

2 Group signatures

Group signatures allow members of a group of signers to issue signatures on behalf of the group that can be verified without telling which specific member issued it [15]. These schemes typically include a group manager (GM) responsible for setting up the group and, sometimes, managing it. The main functionality is issuing signatures (*sign*) and *verify*ing them. Additionally, GM can *open* a group signature to retrieve the identity of its issuer. These schemes endow users with anonymous authentication and unlinkability. However, there are also schemes to enable signatures *linkability* [47] or *traceability* [34]. It is also possible to *revoke* a member's private key, preventing her to issue new signatures. This may be done by publishing the trapdoor used for *open*, publishing the trapdoor used as *trace*, or just enabling an authority for answering this kind of status requests. Some schemes modify this basic scenario, by dividing GM in multiple authorities for the tasks related to managing the group [6]; or add new authorities for performing other delicate tasks, like tracing revoked group members [34].

Group signatures provide an anonymity degree proportional to the group size. Thus, they are typically used for privacy-respectful authentication in anonymous certificates [5], e-voting [47], e-cash [38], and anonymous attestation [10].

Standards and implementations. Group signatures have been standardised by the ISO/IEC [29] (general setting and main operations) and [30] (which defines a total of 7 schemes with opening and linking capabilities). In [5,20] extended X.509 certificates are used to manage digital identities based on group signatures. Several implementations of group signature schemes are currently available online. [8] is implemented in C within the `PBC_sig` library[1] and using Python within the Charm framework[2] [3]; [6] is implemented in C in the `FTMGS` library[3]; [11] and [4] are implemented in the `libgs` library using Java, as part of the PP2db project[4]; and the extensible `libgroupsig` C library[5] implements [8], [34] and [16], and allows the addition of new group signature schemes.

3 Ring signatures

As an alternative to group signatures, ring signatures can be considered to have a more flexible solution where anonymity revocation is not possible. Ring signatures were first introduced in [42], and further contributions incorporate additional controls on the original proposal. Thus, in some specific contexts it is necessary to determine whether two signatures have been created by the same group member, which is addressed by the so-called linkable ring signatures [37]. In other situations it is convenient to adopt traceable ring signatures to trace the origin of two signatures with respect to the same metainformation or tag [22]. The main difference between group and ring signatures is given by the initial setup and the possibility of conforming ad hoc groups. Ring signatures are not ruled by a central authority and there is no need for an initial setup. Moreover, if one adheres to ring signatures, then groups can be generated without an extra cost derived from re-organization (i.e., a new setup to include the new members of the group). This characteristic is of major importance to tackle non-closed groups in e-cash, e-voting, and e-token systems [47]. In fact, here we have to acknowledge the efforts from the bitcoin-related community. Ring signatures are key components of P2P electronic cash infrastructures where the existence of a trusted central authority cannot be taken for granted. Privacy preserving social networks also demand procedures to validate a piece of information as originated by a certain group and to avoid identity forgery attacks (as sockpuppetry or sybil attacks), and correspondingly there are some recent proposals applying linkable ring signatures for such a goal [39].

Standards and implementations. Along with group signatures, ring signatures are one of the cryptographic means to manage users' anonymity. This being the case, the most relevant standard for ring signatures is given by [29] and [30]. Regarding software implementation of ring signatures, we have to underline the

[1] `http://crypto.stanford.edu/pbc/sig/`.
[2] `https://code.google.com/p/charm-crypto/`.
[3] `http://www.lcc.uma.es/~vicente/swprj/index.html#libftmgs`.
[4] `http://www.ing.unibs.it/ntw/tools/pp2db/`.
[5] `https://bitbucket.org/jdiazvico/libgroupsig`.

variant of [23] provided in cryptonote's library[6], the inclusion of a Python version of [17] in Charm, the C implementation given in the PBC library for [49], and the software counterpart of [36] in the Crypto-book prototype [7].

4 Blind signatures

A blind signature scheme allows a user U to obtain a signature from a signer S over any arbitrary message m, but without S learning anything about m [14]. There are variants of this basic behavior, like partially blind signatures [1], allowing to include a message known by both signer and user; restrictive blind signatures [9], which only allow the issuance of a blind signature for messages that comply certain rules; finally, in fair blind signatures [45] an authority has privileged information allowing the signer to link message and signature pairs. Usually, a blind signing protocol is a three step process. First, during the *blinding* step, the user blinds the message to be signed with the help of a random value. This blinded value is then sent to the signer, who performs some verifications upon it, *signs* the blinded message and sends the result to the user. Finally, the user generates the final signature applying the random value used to blind the message to *unblinds* the received token. Blind signatures offer a distinction between authentication and token assignment, which is of major importance for creating privacy respectful protocols, like e-cash [32] and e-voting [35].

Standards and implementations For blind signatures, there is currently an undergoing effort by the ISO/IEC to standardise the general setting, entities and processes [27], along with the discrete logarithm based mechanisms [28]. As for available implementations, the Bouncy Castle Java library[8] includes the class `RSABlindingEngine` for [14] blind signatures. Many current systems use the basic variant of blind signatures (like OpenCoin[9]) and, since it is quite simple to program given a working RSA implementation, there seems to be a lack of independent open source libraries. There also does not seem to exist open source implementations of any of the derivatives of blind signatures.

5 Multi and aggregate signatures

In a multisignature n signers create a signature over a message m, such that it is possible to verify that all of them have participated in the signing process [31]. While one naive way for achieving this will be to have each signer attach her conventional signature (e.g., using RSA), this has the drawback of producing multisignatures that are of linear size in the number of participants and with linear verification time (also depending on the participants). In a multisignature scheme there are n signing steps (one for each signer), and a verification process, which is run independently of the number of signers.

[6] `https://github.com/AlbertWerner/cryptonotecoin`.
[7] `https://github.com/jyale/crypto-book/`.
[8] `https://www.bouncycastle.org/`.
[9] `http://opencoin.org/`.

Standards and implementations. [24] highlights that the ISO/IEC 14888-2 standard [26] can be used to build identity-based multisignatures. Besides, naive multisignatures can be implemented using conventional digital signatures.

6 Threshold signatures

In these schemes, at least t signers out of n need to collaborate in order to produce a valid signature, composing what is called a (t, n)-threshold signature scheme [18]. Group signatures and multisignatures can be seen as $(1, n)$ and (n, n) threshold signature schemes, respectively. Besides signers and verifiers, it is also frequent to find a special entity, the *combiner*, who gathers all the shares and joins them to produce the final signature. Therefore, the processes in a threshold signature scheme are: a *signing* algorithm through which each of the signers produces a signature share; a *combination* process (which may be performed by the signers, verifier, or by the combiner), which merges all the available shares (that must be at least t in a (t, n) scheme); and the *verify* algorithm, determining whether the signature produced after combining the shares is valid.

Standards and implementations. *threshsig*[10] implements [44] in Java, and *Threshold_ECDSA*[11] implemented in Java an ECDSA based threshold scheme, although it does not seem to be available at present. Finally, Bitcoin uses a simple approach for threshold signatures for contract signing[12].

7 Proxy signatures

Proxy signatures allow a user U_1 (the delegatee) to sign a message on behalf of another user U_2 (the delegator), if a trusted proxy cooperates [40]. Proxy re-signatures [7] allow a proxy to convert a valid signature by U_1 over a given message into a valid signature by U_2 over the same message. In proxy signature and proxy re-signature schemes, there are delegators, delegatees and a proxy required to convert signatures. The proxy and the delegatee must run two separate processes *psign* and *dsign*, respectively, in order to produce the final signature on behalf of the delegator. In proxy re-signatures, the equivalent to these two processes is named *resign*, and it is executed by the proxy. Also, the function *rekey* creates the necessary keys for the proxy to be able to perform the transformation. The main application of proxy signatures is on delegating signing capabilities. In [25] proxy re-signatures are also proposed for authenticated routing.

[10] `https://code.google.com/p/threshsig`. As a work of an undergraduate student, however, it is no longer maintained. See `http://www.metzdowd.com/pipermail/cryptography/2013-November/018674.html`

[11] `http://nssl.eew.technion.ac.il/files/Projects/Threshold_ECDSA/html/doc/ECDssSignature.html`.

[12] `https://en.bitcoin.it/wiki/Contracts`. In Bitcoin *multisig* transactions are specified through the `CHECKMULTISIGVERIFY` opcode and demand n valid signatures out of m for a transaction to be aproved. This is actually an approach for threshold signatures, although it requires n separate signatures (instead of just one).

Standards and implementations. While we have found publications reporting analysis on implementation of specific schemes (like [46], but which does not make the code available), we have not been able to locate either standards or implementations worth mentioning for this primitive.

8 Signatures of knowledge

A conventional digital signature proves a statement of the form "*The issuer of this signature, with public key PK, knows the corresponding private key SK*". Signatures of knowledge extend this, allowing to issue digital signatures proving knowledge of witnesses for any NP statement [13]. Specifically, for any NP language L, given a statement $x \in L$ and a witness w proving it, a signature of knowledge provides a signing algorithm $\sigma = sign(m, w, x, L)$ which creates a signature σ of m over $x \in L$, that can be read as "*Someone knowing a witness to $x \in L$ is sending the message m*". This can be verified with the verification counterpart algorithm $verify(\sigma, m, x, L)$. Signatures of knowledge allow creating privacy respectful signatures, since there is no need to *leak* any additional information besides the knowledge of some fact. In turn, this enables important functionalities demanded, for example, to implement delegate credentials.

Standards and implementations. There are no standards for this primitive. Additionally, we have not been able to find direct implementations either. However, it is worth noting that signatures of knowledge can be easily constructed from Zero-Knowledge proof systems using the technique explained in [21] (this is usually called Non-Interactive Zero-Knowledge proofs, or NIZK proofs).

9 Identity-based signatures

Identity-based signatures eliminate the need of distributing public keys, allowing the verification of digital signatures just from the identity that the signer claims to own [43]. For this, initial schemes relied on a trusted Private Key Generators (PKG), which are basically trusted entities generating the private keys used for signing that produce public key independence. However, recent proposals have reduced the trust placed in this entity, by allowing the detection of dishonest actions on its behalf (thus, addressing the key escrow problem) [48]. Identity-based signature schemes include *signing* and *verifying* processes, the latter requiring the identity of the signer instead of her public key. Additionally, the mentioned schemes reducing the trust in the PKG add another process for checking if the signature has been generated by a dishonest PKG.

Standards an implementations. in [24] it is highlighted that the standard ISO/IEC 14888-2 [26] can be applied to derive identity-based multisignatures. Concerning implementations, there is an implementation of the [41] scheme in the JPBC library[13] and implementations of several schemes in Cayrel's website[14].

[13] `http://gas.dia.unisa.it/projects/jpbc/schemes/ibs_ps06.html`.
[14] `http://cayrel.net/?Implementation-of-code-based-zero`.

10 Conclusion

The laws of identity have a plethora of implications in its general scope, but even more in the specific context of the digital realm. Along this paper we have distinguished some of the most relevant properties of digital identities in today communication networks. We have conducted a survey to expose some important contributions both from the theoretical and practical point of view. In Table 1 we show a summary of the standards and implementations cited along this paper. This list in some cases leads to highlight the lack of software proposals and/or formal standards. We hypothesise that the lack of implementations may be due to the fact that system designers (usually computer science engineers, not cryptographers) are not typically aware of these new digital signature schemes. Consequently, system designers resort to conventional methods to implement the required functionality, creating a circular dependency that could be problematic unless those conventional primitives are efficiently implemented and rigorously tested.

However, since the transition from formal definition of cryptographic primitives to cryptography engineering should be done through a rigorous evaluation process, standardisation is not an option but a commitment. Consequently, we should claim the absence of software libraries for certain digital signatures schemes as a call and an urging. This need should be guided by a rigorous evaluation process on the grounds of formal and computational models, but also taking as bottom line basic cryptographic libraries validated by the cryptographic community and broadly accepted. This is the case of GMP[15], MIRACL[16], Cryptopp[17], and Ben Lynn's library for Pairing Based Cryptography[18]. These libraries contain the most fundamentals symmetric and asymmetric cryptosystems, but there also exist libraries providing implementations of Zero-Knowledge proofs[19], crytographic commitments[20] and Oblivious Transfers[21]. Finally, it is necessary to comment on current efforts to adequate digital signatures to low computational-power environments. Certainly, before incorporating any cryptographic library we should realised a performance study using benchmarks in the vein of [2]. Here, it is relevant the NaCL library[22], since it contains some important lightweight digital signatures implementations for ARM architectures.

[15] https://gmplib.org/.
[16] `http://docs.certivox.com/docs/miracl`.
[17] `http://www.cryptopp.com/`.
[18] `http://crypto.stanford.edu/pbc/`.
[19] `https://www.peloba.de/index.php/zk-library/?lang=en`
[20] `git://git-crysp.uwaterloo.ca/polycommit`, `http://scapi.readthedocs.org/`.
[21] `https://github.com/JHUISI/charm/releases`.
[22] `http://nacl.cr.yp.to/`.

Signature type	Standardization efforts	Implementations
Group signatures	[29,30,5,20]	Extensible libraries
Ring signatures	[29,30,5]	Specific schemes
Blind signatures	[27,28]	Specific schemes
Multi signatures	[26] (related)	Unknown
Threshold signatures	Unknown	Specific schemes
Proxy signatures	Unknown	Unknown
Signatures of knowledge	Unknown	Unknown
Identity-based signatures	[26] (related)	Specific schemes

Table 1: Reviewed signature types and related standards and implementations. If there are standards or implementations that we have not located/referenced, please contact us.

Acknowledgements

This work was supported by Comunidad de Madrid (Spain) under the project S2013/ICE-3095-CM (CIBERDINE) and the Spanish Government project TIN2010-19607.

References

1. Abe, M., Fujisaki, E.: How to date blind signatures. In: ASIACRYPT (1996)
2. Abusharekh, A.: Comparative analysis of software libraries for public key cryptography
3. Akinyele, J.A., Garman, C., Miers, I., Pagano, M.W., Rushanan, M., Green, M., Rubin, A.D.: Charm: a framework for rapidly prototyping cryptosystems. J. Cryptographic Engineering 3(2), 111–128 (2013)
4. Ateniese, G., Camenisch, J., Joye, M., Tsudik, G.: A practical and provably secure coalition-resistant group signature scheme. In: Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings. pp. 255–270 (2000)
5. Benjumea, V., Choi, S.G., Lopez, J., Yung, M.: Anonymity 2.0 - X.509 extensions supporting privacy-friendly authentication. In: Cryptology and Network Security, 6th International Conference, CANS 2007, Singapore (2007)
6. Benjumea, V., Choi, S.G., Lopez, J., Yung, M.: Fair traceable multi-group signatures. In: Financial Cryptography. pp. 231–246 (2008)
7. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Advances in Cryptology - EUROCRYPT '98, International Conference on the Theory and Application of Cryptographic Techniques, Espoo, Finland, May 31 - June 4, 1998, Proceeding. pp. 127–144 (1998)
8. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: CRYPTO (2004)
9. Brands, S.: Untraceable off-line cash in wallets with observers (extended abstract). In: CRYPTO. pp. 302–318 (1993)
10. Brickell, E.F., Camenisch, J., Chen, L.: Direct anonymous attestation. In: Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004, Washington, DC, USA, October 25-29, 2004. pp. 132–145 (2004)

11. Camenisch, J., Groth, J.: Group signatures: Better efficiency and new theoretical aspects. In: Security in Communication Networks, 4th International Conference, 2004, Italy, September 8-10, 2004, Revised Selected Papers. pp. 120–133 (2004)
12. Cao, Z., Liu, M.: Classification of signature-only signature models. Science in China Series F: Information Sciences 51(8), 1083–1095 (2008)
13. Chase, M., Lysyanskaya, A.: On signatures of knowledge. In: Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings. pp. 78–96 (2006)
14. Chaum, D.: Blind signatures for untraceable payments. In: CRYPTO (1982)
15. Chaum, D., van Heyst, E.: Group signatures. In: Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings. pp. 257–265 (1991)
16. Choi, S.G., Park, K., Yung, M.: Short traceable signatures based on bilinear pairings. In: IWSEC. pp. 88–103 (2006)
17. Chow, S.S., Yiu, S.M., Hui, L.C.: Efficient identity based ring signature. In: Applied Cryptography and Network Security. pp. 499–512. Springer (2005)
18. Desmedt, Y., Frankel, Y.: Shared generation of authenticators and signatures (extended abstract). In: Advances in Cryptology - CRYPTO '91, 11th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings. pp. 457–469 (1991)
19. Diaz, J., Arroyo, D., Rodriguez, F.B.: A formal methodology for integral security design and verification of network protocols. Journal of Systems and Software 89(0), 87 – 98 (2014)
20. Diaz, J., Arroyo, D., Rodriguez, F.B.: New x.509-based mechanisms for fair anonymity management. Computers & Security 46, 111–125 (2014)
21. Fiat, A., Shamir, A.: How to prove yourself: Practical solutions to identification and signature problems. In: Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings. pp. 186–194 (1986)
22. Fujisaki, E.: Sub-linear size traceable ring signatures without random oracles. IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences 95(1), 151–166 (2012)
23. Fujisaki, E., Suzuki, K.: Traceable ring signature. In: Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, China, April 16-20, 2007, Proceedings. pp. 181–200 (2007)
24. Harn, L., Ren, J., Lin, C.: Efficient identity-based GQ multisignatures. Int. J. Inf. Sec. 8(3), 205–210 (2009)
25. Hohenberger, S.: Advances in Signatures, Encryption, and E-Cash from Bilinear Groups. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science (2006)
26. ISO 148888-2: Information technology – Security techniques – Digital signatures with appendix – Part 2: Integer factorization based mechanisms (2014)
27. ISO/IEC 18370-1: Information technology – Security techniques – Blind digital signatures – Part 1: General (2015)
28. ISO/IEC 18370-2: Information technology – Security techniques – Blind digital signatures – Part 2: Discrete logarithm based mechanisms (2014)
29. ISO/IEC 20008-1: Information technology – Security techniques – Anonymous digital signatures – Part 1: General (2013)
30. ISO/IEC 20008-2: Information technology – Security techniques – Anonymous digital signatures – Part 2: Mechanisms using a group public key (2013)
31. Itakura, K., Nakamura, K.: A public-key cryptosystem suitable for digital multisignatures. NEC J. Res. Dev. (1983)

32. Juels, A., Luby, M., Ostrovsky, R.: Security of blind digital signatures. In: Advances in Cryptology-CRYPTO'97, pp. 150–164. Springer (1997)
33. Katz, J.: Digital Signatures. Advances in Information Security, Springer US (2010)
34. Kiayias, A., Tsiounis, Y., Yung, M.: Traceable signatures. In: Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Switzerland, May 2-6, 2004 (2004)
35. Kucharczyk, M.: Blind signatures in electronic voting systems. In: Kwiecien, A., Gaj, P., Stera, P. (eds.) Computer Networks, Communications in Computer and Information Science, vol. 79, pp. 349–358. Springer Berlin Heidelberg (2010)
36. Liu, J.K., Wei, V.K., Wong, D.S.: Linkable spontaneous anonymous group signature for ad hoc groups. In: Information Security and Privacy. Springer (2004)
37. Liu, J.K., Wong, D.S.: Linkable ring signatures: Security models and new schemes. In: Computational Science and Its Applications–ICCSA. Springer (2005)
38. Lysyanskaya, A., Ramzan, Z.: Group blind digital signatures: A scalable solution to electronic cash. In: Financial Cryptography, Second International Conference, FC'98, British West Indies, February 23-25, 1998, Proceedings. pp. 184–197 (1998)
39. Maheswaran, J., Wolinsky, D.I., Ford, B.: Crypto-book: an architecture for privacy preserving online identities. In: Twelfth ACM Workshop on Hot Topics in Networks, HotNets-XII, College Park, MD, USA, November 21-22, 2013. p. 14 (2013)
40. Mambo, M., Usuda, K., Okamoto, E.: Proxy signatures for delegating signing operation. In: CCS '96, Proceedings of the 3rd ACM Conference on Computer and Communications Security, India, March 14-16, 1996. pp. 48–57 (1996)
41. Paterson, K.G., Schuldt, J.C.N.: Efficient identity-based signatures secure in the standard model. In: Information Security and Privacy, 11th Australasian Conference, ACISP 2006, Melbourne, Australia, July 3-5, 2006, Proceedings. pp. 207–222 (2006), http://dx.doi.org/10.1007/11780656_18
42. Rivest, R.L., Shamir, A., Tauman, Y.: How to leak a secret. In: Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings. pp. 552–565 (2001)
43. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings. pp. 47–53 (1984)
44. Shoup, V.: Practical threshold signatures. In: Proceedings of the 19th International Conference on Theory and Application of Cryptographic Techniques. pp. 207–220. EUROCRYPT'00, Springer-Verlag, Berlin, Heidelberg (2000)
45. Stadler, M., Piveteau, J.M., Camenisch, J.: Fair blind signatures. In: EUROCRYPT. pp. 209–219 (1995)
46. Tang, S., Xu, L.: Proxy signature scheme based on isomorphisms of polynomials. In: Network and System Security - 6th International Conference, NSS 2012, Wuyishan, Fujian, China, November 21-23, 2012. Proceedings. pp. 113–125 (2012)
47. Tsang, P.P., Wei, V.K.: Short linkable ring signatures for e-voting, e-cash and attestation. In: Information Security Practice and Experience, First International Conference, ISPEC 2005, April 11-14, 2005, Proceedings. pp. 48–60 (2005)
48. Yuen, T.H., Susilo, W., Mu, Y.: How to construct identity-based signatures without the key escrow problem. Int. J. Inf. Sec. 9(4), 297–311 (2010)
49. Zhang, F., Kim, K.: Id-based blind signature and ring signature from pairings. In: Advances in cryptology-ASIACRYPT 2002, pp. 533–547. Springer (2002)