Unbalanced Identity-Based Key Agreement With Identity Privacy

Zhaohui Cheng and Richard Comley

School of Computing Science, Middlesex University
The Burroughs Hendon, London NW4 4BT, UK
{m.z.cheng,r.comley}@mdx.ac.uk

Abstract. On AisaCrypt 2005 Barreto *et al.* proposed an elegant identity-based signcryption scheme built from pairings. By using this scheme as the main ingredient, we present a suit of identity-based key agreements, which not only achieve strong security, but also have unbalanced computation complexity and identity privacy. Furthermore, the protocols are also equipped to mitigate the threat of Denial-of-Service attacks. These make the protocols particularly suitable for certain applications, such as authentication in mobile networks with low power devices.

1 Introduction

Key agreement allows parties to exchange messages and establish a common key known only to the participants. As an important cryptographic primitive, key agreements are widely used in the computer world, including *wired* and *wireless* networks, to authenticate entities and to build secure channels.

To design a key agreement for wireless networks, apart from basic security requirements, some special issues should be considered. Normally a wireless network consists of mobile devices and access points which provides network access to mobile devices (we refer to this scenario as the client/server model). First, as the mobile devices have only restricted computation power while the access points can perform some heavy computations, the protocol should have unbalanced computation complexity, i.e., the mobile devices should only be involved with light-weight computations while heavier operations should be on the access points. Second, as the air interface is open to everyone, user's identity becomes sensitive information. The disclosure of identity enables adversaries to track mobile users' movement and to associate specific communication with a user. Hence a protocol for the this type of network should consider hiding user's identity. And third, since the Denial-of-Service (DoS) attack has become a common threat, when we design an unbalanced protocol, the DoS attack on a heavy-computing server should be taken into account.
OUR CONTRIBUTION. In this work, building on recently published signcryption scheme [4] we present two efficient unbalanced key agreements which have very strong security, at the same time can hide the client's identity. The protocols are also equipped to mitigate the threat of DoS attacks. The protocols

are identity-based schemes following Shamir's formulation [18]. In such system a certificate is no longer required and an entity's identifier is used as its public key. This greatly simplifies the management of public keys and removes the reliance on a PKI.

RELATED WORK. Using Schnorr's signature scheme [15], Jakobsson and Pointcheval [11] and Wang and Chan [21] designed two unbalanced key agreement suitable for low power devices. While these schemes are certificated-based and rely on a PKI system which proves hard to deploy. Recently many two-party identity-based key agreements using bilinear pairings [2] have been proposed. Key agreements in [19, 9, 13, 20, 7] are Diffie-Hellman style protocols constructed from pairings and each party in the protocols has equal computation complexity, though the scheme in [7] indeed provides identity privacy for one party. Using the identity-based signcryption scheme [4] as we shall do, Choi *et al.* proposed a scheme with unbalanced computation complexity intended for low power mobile devices [8]. However, the protocol has a few problems which restrict the application of the scheme in practice. (1) The scheme does not achieve the perfect forward secrecy (PFS) and master-key forward secrecy, i.e., the disclosure of a server's private key (or the master key of the used identity-based system) enables the adversary to recover session keys established before. (2) The scheme does not hide the client's identity which might be very sensitive information in the mobile network. (3) In the scheme, the server is very much vulnerable to the DoS attack. An adversary can trivially launch the DoS attack to force the server to perform all the heavy computations including pairings in vain. (4) The scheme is a two-flow protocol which only provides explicit server authentication. Such type of protocol could cause problems in certain environments. For example, an adversary can replay a client (mobile device) message which will deceit the server (access point) into creating a new channel and this could cause legitimate sessions to terminate prematurely. This also wastes certain resources.

The paper is constructed as follows. In Section 2, we first recall some facts of bilinear pairings and then overview the pairing computations. We describe the details of the proposed protocols and analyse the security in Section 3. In Section 4, we discuss the protocols' ability of mitigating DoS attacks. Then we compare our proposals with existing schemes in Section 5. Finally, we conclude the paper.

2 Preliminary

2.1 Bilinear Pairing

Here we briefly review the basic facts about bilinear maps and the associated groups.

- $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are cyclic groups of prime order q.
- P_1 is a generator of $\mathbb{G}_1$ and P_2 is a generator of $\mathbb{G}_2$.
- ψ is an isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$ with $P_1 = \psi(P_2)$.
- $\hat{e}$ is a map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

The map $\hat{e}$ must have the following properties.

Bilinear: $\forall(P,Q) \in \mathbb{G}_1 \times \mathbb{G}_2$ and $\forall(a,b) \in \mathbb{Z}_q \times \mathbb{Z}_q$, we have $\hat{e}(aP,bQ) = \hat{e}(P,Q)^{ab}$.
Non-degenerate: $\hat{e}(P_1,P_2) \neq 1$.
Computable: $\forall(P,Q) \in \mathbb{G}_1 \times \mathbb{G}_2$, $\hat{e}(P,Q)$ is efficiently computable.

In recent years, using the pairing instance on elliptic (or hyperelliptic) curves or (the Weil or Tate pairing) [2], many exciting cryptosystems were constructed. In the sequel, we use pairing only to refer to the Tate (or Weil) pairings.

2.2 Computing Pairing

When designing a protocol, apart from the security concerns, the computation performance is also an important considieration, particularly in the restricted device environments. Here we overview the computation complexity of basic operations in a pairing-based scheme which helps to read the rationale of choosing operations in the proposed protocols.

A pairing-based scheme normally involves three heavy operations: the point Scalar in $\mathbb{G}_1$ or $\mathbb{G}_2$, the Pairing, and the Exponentiation in $\mathbb{G}_T$. The computation of pairing is complicated and the performance of the algorithm (Miller's algorithm [3]) heavily depends on the security level and the parameters chosen for the elliptic curves, including the characteristic and size of the underlying field, embedding degree and coefficients of the curve and even the representation of $\mathbb{G}_1$ and $\mathbb{G}_2$ [3]. Here we only recall some basic results in this field and for details, please refer to [3, 16, 10].

In [16], Scott summarised the techniques of efficient pairing computation on a non-supersingular curve. Similar results on supersingular curves can be obtained as well. Here we simply recall the computation complexity result in [16] as in Table 1 with some improvements (for other cases, please refer to [10]). The numbers are counted when the curve and parameters are chosen as in [16] which has about the same security as 1024-bit RSA.

Pairing	Pairing with precomp	Scalar	Exponentiation
3277M^{*1}	1655M	1222M^{*2}	318M^{*3}

^{*1}: This number is smaller than the one counted in [16] (3992.7), mainly because we note that step 5 of the BLKS algorithm in [16] can be computed by $9M + 8S$ instead of 21 muls (M or S) counted in [16].
^{*2}: This is according to Table IV.3. in [6] by assuming the windowed NAF algorithm is used and the scalar number is a 160-bit random integer.
^{*3}: As noticed in [17], the exponentiation in $\mathbb{G}_2$ can be computed by the Lucas ladder. The number is counted when the exponent is a 160-bit random integer.

Table 1. Operation Complexity

As in [16], we assume that the cost of one modular inversion (I) is roughly equal to 10 modular multiplications (M) and a modular squaring (S) is equivalent to 0.89 of a multiplication. And we can see that pairing can be sped up dramatically if some intermediate results are pre-computed.

3 Unbalanced IB Key Agreements With Identity Privacy

In this section we present two closely related schemes attempting to achieve following goals: (1) to achieve strong security as possible, (2) to hide the client's identity, (3) to make the operations on clients light-weight as possible, and (4) to reduce the threat of DoS attacks on servers. To meet goal one, we combine a Diffie-Hellman exchange with a variant of the Barreto *et al.*'s signcryption. The client identity privacy (IP) can be guaranteed by the straight application of the ciphertext anonymity of the used signcryption. For goal three, the client should perform fewer heavy operations, particularly the costly pairing computation. Moreover, the client should be able to compute more heavy operations off-line as possible. This will make the protocol be executed more efficiently. Finally, to reduce threat of DoS attacks, the server should be able to check invalid client messages as early as possible before performing some heavy operations. We present two protocols as below to meet these goals.

3.1 Two Protocols

The protocols involving clients, servers and a Key Generation Center (KGC) work as follow.
Setup. Given a security parameter k, the KGC proceeds the following steps.

1. Generate three cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order q, and a bilinear pairing map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. Pick a random generator $P_2 \in \mathbb{G}_2^*$, $P_1 \in \mathbb{G}_1^*$ and set $g = \hat{e}(P_1, P_2)$.
2. Pick a random $s \in \mathbb{Z}_q^*$ and compute $P_{pub} = sP_2$.
3. Pick five cryptographic hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q^*$, $H_2 : \{0,1\}^l \times \mathbb{G}_T \to \mathbb{Z}_q^*$, $H_3 : \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G}_T \times \{0,1\}^l \to \{0,1\}^{2l}$, $H_4 : \mathbb{G}_T \times \mathbb{G}_T \to \{0,1\}^n$ and $H_5 : \{0,1\}^l \times \{0,1\}^l \times \mathbb{G}_T \times \mathbb{G}_T \times \mathbb{G}_T \to \{0,1\}^w$ for some integers $l, n, w > 0$.
4. Pick a message authentication code (MAC) scheme with existential unforgeability [1] such that $MAC_K(m)$ is the tag on message m under key K.

The KGC keeps s secretly as the master key. ($\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, P_1, P_2, P_{pub}, $\hat{e}$, H_1, H_2, H_3, H_4, H_5, MAC) are published as the system parameters.

Extract. Given an identifer string $ID_A \in \{0,1\}^l$ of entity A, the master key and the system parameters, the KGC computes $d_A = \frac{1}{s+H_1(ID_A)}P_1$ and passes it securely to A after successfully authenticating the entity.

$$
\begin{array}{rcl}
(Client)A & & B(Server) \\
\hline
 & \xleftarrow{ID_B, g^{r_B}} & r_B \leftarrow \mathbb{Z}_q^* \\
r_A \leftarrow \mathbb{Z}_q^*, R = g^{r_A} & & \\
h = H_2(ID_A, R) & & \\
S = (r_A + h)d_A & & \\
T = r_A H_1(ID_B)P_2 + r_A P_{pub} & & \\
w = H_3(R, g^{r_B}, g^{r_A r_B}, ID_B) & & \\
C = w \oplus ID_A \| ID_B & \xrightarrow{C,S,T} & R = \hat{e}(d_B, T) \\
 & & w = H_3(R, g^{r_B}, R^{r_B}, ID_B) \\
 & & ID_A \| ID_B = C \oplus w \\
 & & \text{check } ID_B \\
 & & h = H_2(ID_A, R) \\
 & & \text{check } R \cdot g^h \stackrel{?}{=} \hat{e}(S, H_1(ID_A)P_2 + P_{pub}) \\
 & & MK = H_4(R, R^{r_B}) \\
MK = H_4(R, g^{r_A r_B}) & \xleftarrow{t_B} & t_B = MAC_{\texttt{MK}}(ID_B, ID_A, g^{r_B}) \\
\text{check MAC } t_B & & \\
\multicolumn{3}{c}{SK = H_5(ID_A, ID_B, g^{r_A}, g^{r_B}, g^{r_A r_B})} \\
\hline
\end{array}
$$

Fig. 1. Protocol 1

The first key agreement proceeds as in Fig. 1 where $\leftarrow$ means uniformly sampling and $a\|b$ is the concatenation of a and b. The agreed session key is computed as SK.

In some cases, it is more convenient for the client to initiate a session if it knows the server's identity in advance by some means. So, we present a simple variant of Protocol 1 as in Fig. 2. Protocol 2 shares a similar **Setup** and the same **Extract** algorithm with Protocol 1. The only difference in **Setup** is that Protocol 2 uses a variant of H_3 as $H_3' : \mathbb{G}_T \times \{0,1\}^l \rightarrow \times \{0,1\}^{2l}$.

The protocols are unbalanced and clients only need to compute three scalars and two exponentiations and so the costly pairing is avoided. Furthermore, if the server identity is known to the client a prior, the three scalars and one exponentiation can be performed off-line (two scalars and one exponentiation can be off-line computed even if the server's identifer is obtained during the execution of the protocols). This enables the client to complete the protocol efficiently (note from Table 1 that the exponentiation is only a mild operation). We also note that the first pairing on the server can make use of the pre-computation technique as d_B is always fixed.

3.2 Security Analysis

We use C in Protocol 1 (resp. the message tag t_A in Protocol 2) instead of a tweaked signature such as $S = (r_A + H_2'(ID_A, R, g^{r_B}, g^{r_A r_B}))d_A$ to bind the

$$\begin{array}{rcl}
(Client)A & & B(Server) \\
\hline
& \xleftarrow{ID_B} & \\
r_A \leftarrow \mathbb{Z}_q^*, R = g^{r_A} & & \\
h = H_2(ID_A, R) & & \\
S = (r_A + h)d_A & & \\
T = r_A H_1(ID_B)P_2 + r_A P_{pub} & & \\
C = H_3'(R, ID_B) \oplus ID_A \| ID_B & \xrightarrow{C,S,T} & R = \hat{e}(d_B, T) \\
& & ID_A \| ID_B = C \oplus H_3'(R, ID_B) \\
& & \text{check } ID_B \\
& & r_B \leftarrow \mathbb{Z}_q^* \\
& & MK = H_4(R, R^{r_B}) \\
MK = H_4(R, g^{r_A r_B}) & \xleftarrow{g^{r_B}, t_B} & t_B = MAC_{\mathtt{MK}}(ID_B, ID_A, g^{r_B}) \\
\text{check MAC } t_B & & \\
t_A = MAC_{\mathtt{MK}}(ID_A, ID_B, g^{r_B}, S, T) & \xrightarrow{t_A} & \text{check MAC } t_A \\
& & h = H_2(ID_A, R) \\
& & \text{check } R \cdot g^h \overset{?}{=} \hat{e}(S, H_1(ID_A)P_2 + P_{pub}) \\
\multicolumn{3}{c}{SK = H_5(ID_A, ID_B, g^{r_A}, g^{r_B}, g^{r_A r_B})} \\
\hline
\end{array}$$

Fig. 2. Protocol 2

message from the server and provide the client key confirmation, because we want to enable the client to compute S off-line. However, this certainly complicates the formal security analysis of the protocols. Here as did in many literatures such as [12], we only heuristically reason that the protocols have following security attributes. We analyse Protocol 1 only and for similar reasons Protocol 2 is secure.

– Key-Compromise Impersonation (KCI) resilience. We analyse two cases. Case 1, the adversary $\mathcal{A}$ knows the server's private key and wants to impersonate a client whose private key it doesn't know. In the impersonation session, it is clear that $\mathcal{A}$ has to query $H_3(R, g^{r_B}, R^{r_B}, ID_B)$ to generate C to pass the check on ID_B (we call it check 1) with probability greater than $1/2^l$ if we assume that H_3 is a random oracle [5]. In fact, the server would only accept the message with even lower probability because it is very unlikely that S is the signature of the recovered ID_A if H_3 has not been queried on the input. However it appears it is difficult to compute R^{r_B} from (g, g^{r_B}, R) without knowing r such that $R = g^r$. On the other hand, if $\mathcal{A}$ indeed knows r and generates S passing another check on equation $R \cdot g^h \overset{?}{=} \hat{e}(S, H_1(ID_A)P_2 + P_{pub})$, it obviously can compute $d_{ID_A} = \frac{1}{r+h}S$. This means we can use $\mathcal{A}$ to totally break Barreto *et al.*'s signature scheme in [4]. Case 2, $\mathcal{A}$ knows the client's private key and wants to impersonate a server whose private key it doesn't know. As the client will only accept the session if the message tag t_B is generated using the authentication key $H_4(R, R^{r_B})$ which implies that $\mathcal{A}$ has computed R. It is easy to show that if $\mathcal{A}$ can compute R, then we can use $\mathcal{A}$ to completely break the semantic security of the Barreto *et al.*'s signcryption scheme [4]. Note here that H_3 is a random oracle, and the extra inputs including $g^{r_B}, g^{r_A r_B}$ and ID_B would not affect the security of the signcryption.

- Known-Session Key (KnSK) security. For the way of generating SK in the protocol, if H_5 is modelled as a random oracle, the adversary has to query $H_5(ID_A, ID_B, g^{r_A}, g^{r_B}, g^{r_A r_B})$ even only to find a single bit of SK with non-negligible advantage over random guess. This means the adversary can compute $R = g^{r_A}$ and $R^{r_B} = g^{r_A r_B}$. While from the analysis of the KCI resilience, we know such $\mathcal{A}$ doesn't exist.
- Unknown Key-Share Resilience (UknKSh). Because the session key generation function H_5 includes the two party identifiers. There is no way for $\mathcal{A}$ to force a party A to share a key with $\mathcal{A}$ or a party B but at the same time to believe that the key is shared with another party C.
- Forward Secrecy (FS). In both protocols, even the master key s is compromised, the session keys established before are still secure (i.e., the protocols have master-FS which implies Perfect-FS). To compute SK, it is clear that the adversary $\mathcal{A}$ has to compute $g^{r_A r_B}$. Given (g, g^{r_B}, S, T), it appears that $\mathcal{A}$ has to compute r_A to compute $g^{r_A r_B}$. While from both elements (S, T), to find r_A, $\mathcal{A}$ has to solve the discrete logarithm in the chosen group $\mathbb{G}_1$, $\mathbb{G}_2$ or $\mathbb{G}_T$ which supposes to be hard.
- Client Identity Privacy (IP). The client identifier is encrypted in message C and can only be recovered by the intended server. The ciphertext anonymity of the used signcryption scheme guarantees that the third party cannot differentiate one client's message from others'.

4 DoS Attack and Resilience

Now we evaluate the DoS attacks on the protocols and elaborate how to equip the protocols to mitigate the threat. In both protocols, it is very much unlikely that the server B would verify the signature S if the attacker did not know r_A used in the client message. If r_A is not known to the adversary, in Protocol 1, the server stops with probability $1 - 1/2^l$ after check 1 and in Protocol 2, the server almost certainly stops after checking the message tag t_A (we call it check 2). This means the adversary cannot simply replay any client message to launch the full DoS attack (in which the server computes everything except the session key). Now we consider that the attacker indeed samples r_A and computes T. The attacker can compute T and $g^{r_A r_B}$ online to force the server to verify the signature. However, the cost to launch this kind of attack is no longer trivial which needs one scalar and one exponentiation (see Table 1 for the cost). The attacker may reuse T and only compute $g^{r_A r_B}$ online. Though the computation is low, the attack can be easily thwarted by a message replay check mechanism of T on the server. However, an attacker may generate a large pool of (r_A, R, T)

off-line and randomly replay T and compute $g^{r_A r_B}$ online. This would surely overrun the replay check.

Therefore, we have to resort to other means to defeat the sophisticated DoS attack. One solution is to introduce extra online computation on the client side, which can be done by a classic "puzzle" mechanism. The "puzzle" works as follow. In the **Setup** procedure, another hash function f and an integer t is chosen as part of the system parameters. In Protocol 1, if the server finds that the failure times of verifying S reach a chosen threshold, it indicates by a puzzle flag in the first message to the client that the client should solve a puzzle. Then the client tries to find an input N such that $f(g^{r_B}\|N) = 0^t$. Once the client has found such N, it sends message (C, S, T, N) to the server. The server now first verifies that N is indeed a solution to the puzzle before computing R. Depending on the level of the DoS attack, the server may tune the hardness of the puzzle by giving in the first message a new value of t, which can only be little than the one specified in the system parameters. Similarly, in Protocol 2, the client tries to find a solution N after verifying the message tag t_B which is computed on a message including the extra puzzle flag from the server. If we consider the possibility of DoS attacks on the client by making use of the new puzzle challenge, Protocol 2 might be a better choice if the client is aware of the server's identity before hand, though the server is not protected against the DoS attack on the first half of the protocol (recall that the first pairing can be computed with the pre-computation technique).

5 Evaluation of Security and Complexity

Here we summarise the security properties and computation complexity of the proposed protocols and some other schemes in the literature in Table 2.

Protocol	KnSK	FwS	UknKSh	KCI	IP	Comp Complexity	
						Client	Server
Protocol 1	✓	Master-FS	✓	✓	✓	0P+3S+2E	2P+1S+3E
Protocol 2	✓	Master-FS	✓	✓	✓	0P+3S+2E	2P+1S+3E
CHLS[8]	✓	Client-FS	✓	✓	χ	0P+3S+1E	2P+1S+1E
CCCT[7]	✓	Perfect-FS	✓	✓	✓	1P+2S+1E	
Wang[20]	✓	Perfect-FS	✓	✓	χ	1P+2S+1E	
CK[9]	✓	FS	✓	✓	χ	1P+2S+0E	
Smart[19]	✓	FS	✓	✓	χ	2P+2S+0E	

Table 2. Computation Complexity and Security Property

We can see that the proposals achieve very strong security, at the same time are very efficient, particularly on the client side with off-line pre-computation.

6 Conclusion

We have presented two efficient mutual-authenticated two-party key agreements with unbalanced computation. The schemes have very strong security including identity privacy and high efficiency on the client side which makes the protocols particularly suitable for the applications in the mobile networks with low power devices. We have also discussed how to equip the protocol to mitigate the DoS attacks on the server.

References

1. M. Bellare, R. Canetti and H. Krawczyk. Keying Hash Functions for Message Authentication. In *Proceedings of Advances in Cryptology - Crypto 96*, LNCS 1109, pp. 1-15.
2. D. Boneh and M. Franklin. Identity Based Encryption from The Weil Pairing. In *Proceedings of Advances in Cryptology - Crypto 2001*, LNCS 2139, pp. 213-229.
3. P.S.L.M. Barreto, H.Y. Kim, B. Lynn and M. Scott. Efficient Algorithms for Pairing-Based Cryptosystems. In *Proceedings of Advances in Cryptology - Crypto 2002*, LNCS 2442, pp. 354-368.
4. P.S.L.M. Barreto, B. Libert, N. McCullagh and J. Quisquater. Efficient and Provably-Secure Identity-Based Signatures and Signcryption from Bilinear Maps. In *Proceedings of AsiaCrypt 2005*, LNCS 3788, pp. 515-532.
5. M. Bellare and P. Rogaway. Random Oracles are Practical: A Paradigm for Desiging Efficient Protocols. In *Proceedings of First ACM Conference on Computer and Communication Security 1993*, pp. 62-73.
6. I.F. Blake, G. Seroussi and N.P. Smart. Elliptic Curve in Cryptography. Cambridage Press, 1999.
7. Z. Cheng, L. Chen, R. Comley and Q. Tang. Identity-based Key Agreement with Unilateral Identity Privacy Using Pairings. To appear in ISPEC 2006. Full version avaialbe on Cryptology ePrint Archive, Report 2005/339.
8. K. Choi, J. Hwang, D. Lee and I. Seo. ID-based Authenticated Key Agreement for Low-Power Mobile Devices. In *Proceedings of ACISP 2005*, LNCS 3574, pp. 494-506.
9. L. Chen and C. Kudla. Identity Based Authenticated Key Agreement from Pairings. In *Proceedings of the 16th IEEE Computer Security Foundations Workshop*, pp. 219-233, 2003.
10. R. Granger and D. Page and N.P. Smart. High Security Pairing-Based Cryptography Revisited. Cryptology ePrint Archive, Report 2006/059.
11. M. Jakobsson and D. Pointcheval. Mutal Authentication for Low Power Mobile Devices. In *Proceedings of Financial Cryptography*, LNCS 2339, pp. 178-195.
12. A. Menezes, M. Qu and S. Vanstone. Some New Key Agreement Protocols Providing Mutual Implicit Authentication. In Proceedings of Second workshop on Selected Areas in Cryptography (SAC 95), LNCS
13. N. McCullagh and P.S.L.M. Barreto. A New Two-Party Identity-Based Authenticated Key Agreement. In *Proceedings of CT-RSA 2005*, LNCS 3376, pp. 262-274.
14. A. Menezes, P. van Oorschot and S. Vanstone. Handbook of Applied Cryptography. CRC Press, 2001.
15. C.P. Schnorr. Efficient Identification and Signatures for Smart Cards. In *Proceedings of Crypto '89*, LNCS 435, pp. 235-251.

16. M. Scott. Computing the Tate Pairing. In *Proceedings of CT-RSA 2005*, LNCS 3376, pp. 293-304.
17. M. Scott and P.S.L.M. Barreto. Compressed Pairings. In *Proceedings of Advances in Cryptology - Crypto 2004*, LNCS 3152, pp. 140-156.
18. A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In In *Proceedings of Advances in Cryptology - Crypto '84*, LNCS 196, pp. 47-53.
19. N.P. Smart. An Identity Based Authenticated Key Agreement Protocol Based on the Weil Pairing. *Electronics Letters* 38, pp. 630-632, 2002.
20. Y. Wang. Efficient Identity-Based and Authenticated Key Agreement Protocol. Cryptology ePrint Archive, Report 2005/108.
21. D. Wong and A. Chan. Efficient and Mutually Authenticated Key Exchange for Low Power Computing Devices. In *Proceedings of Advances in Cryptology - Asiacrypt 2001*, LNCS 2248, pp. 272-289.