A New Two-Party Identity-Based Authenticated Key Agreement

Noel McCullagh[1]* and Paulo S. L. M. Barreto[2]

[1] School of Computing
Dublin City University
Glasnevin, Dublin 9, Ireland.
noel.mccullagh@computing.dcu.ie
[2] Escola Politécnica, Universidade de São Paulo
Av. Prof. Luciano Gualberto, tr. 3, 158
BR 05508-900 São Paulo(SP), Brazil.
pbarreto@larc.usp.br

Abstract. We present a new two-party identity-based key agreement that is more efficient than previously proposed schemes. It is inspired on a new identity-based key pair derivation algorithm first proposed by Sakai and Kasahara. We show how this key agreement can be used in either escrowed or escrowless mode. We also describe conditions under which users of different Key Generation Centres can agree on a shared secret key. We give an overview of existing two-party key agreement protocols, and compare our new scheme with existing ones in terms of computational cost and storage requirements.

Keywords: authenticated key agreement, identity-based cryptography, bilinear maps, Tate pairing.

1 Introduction

In this paper we propose a new two-party authenticated identity-based key agreement from bilinear maps. The basic idea behind an identity-based cryptosystem is that end users can choose an arbitrary string, for example email addresses or other online identifiers, as their public key. This eliminates much of the overhead associated with key management. In traditional PKI settings, key agreement protocols relies on the parties obtaining each other's certificates, extracting each other's public keys, checking certificate chains (which may involve many signature verifications) and finally generating a shared secret. The technique of identity-based encryption (IBE) greatly simplifies this process. This idea was first proposed by Shamir [20] in 1984, made viable by Cocks [9] and Boneh and Franklin [4] in 2001, further streamlined by Sakai and Kasahara [17] in 2003, and is currently an area of very active research (see e.g. [10] for a survey).

* This author wishes to thank Enterprise Ireland for their support with this research under grant IF/2002/0312/N.

There are many key agreement protocols based on bilinear maps, and most have subsequently been broken. One of the first applications of pairing based cryptography was a tripartite key agreement protocol by Joux [13]. Although this key agreement does not authenticate the users, and thus is susceptible to the man-in-the-middle attack, it was a significant step in the development of pairing based cryptography. This original scheme was not identity-based. Many key agreements from bilinear maps have been since proposed. Scott [18], Smart [25], and Chen and Kudla [6] have proposed two-party key agreement protocols, none of which have been broken. All of these schemes require that all parties involved in the key agreement are clients of the same Key Generation Centre (KGC). Nalla proposes a tripartite identity-based key agreement in [14], and Nalla and Reddy propose a scheme in [15], but both have been broken [7, 22]. Shim presents two key agreements [24, 23], but both these schemes have been broken by Sun and Hsieh [26]. Another authenticated tripartite key agreement proposed by Al-Riyami and Patterson [1] was broken by Shim [21].

Most identity-based key agreement protocols have the property of *key escrow*: the trusted authority that issues private keys can recover the agreed session key. This feature is either acceptable, unacceptable, or desired depending on the circumstances. For example, escrow is essential in situations where confidentiality as well as an audit trail is a legal requirement, as in confidential communication in the health care profession. There are other examples, such as personal communications, where it would be advantageous to turn escrow off.

The two-party key agreements proposed by Smart and by Chen and Kudla are escrowed schemes by default. A modification suggested by Chen and Kudla [6] to remove escrow can also be applied to Smart's scheme. However, this modification creates additional computational overhead. Scott's scheme does not allow escrow, and there seems no obvious way to introduce this feature — bar one party to the protocol sending a third party a copy of the agreed key.

Chen and Kudla also suggest a modification that allows two parties that have their public keys generated by two different Key Generation Centre's to communicate. We say that these parties are members of *different domains*. Most key agreements require both parties to be from the same domain. This, for example, might mean that two workers from the same company would be able to generate a shared secret, however employees from two different companies would not be able to generate such a shared secret. We suggest a protocol that, without pairing precomputation, is twice as efficient as the scheme suggested in [6].

We suggest key agreement between domains is an important property of this scheme as, from a commercial viewpoint, identity-based cryptography (IBC) seems particularly well suited to encrypted telephony and encrypted VoIP. For encrypted VoIP to work on a global scale there simply must be compatibility between networks, and therefore key agreement between different networks is important.

Our contributions in this paper are:

– An efficient identity-based authenticated key agreement protocol that can be instantiated in either escrowed or escrowless mode without imposing extra computational steps.
– An efficient key agreement that allows users who have their private keys generated by distinct Key Generation Centres to establish a shared secret without additional overhead, provided standardised curve parameters are used.

This paper is organised as follows. Section 2 introduces basic mathematical concepts. Section 3 describes our proposed authenticated key agreement with escrow, and section 4 introduces our proposed escrowless scheme. In section 5 we present a key agreement protocol for members of distinct key generation domains. We discuss efficiency issues in section 6 and security issues in section 7. Finally, we draw our conclusions in section 8.

2 Mathematical Preliminaries

An elliptic curve $\mathbb{E}(\mathbb{F}_{q^k})$ is the set of solutions (x, y) over the field $\mathbb{F}_{q^k}$ to an equation of the form $y^2 = x^3 + Ax + B$, together with an additional point at infinity, denoted O. There exists an Abelian group law on $\mathbb{E}$, with explicit formulas for computing the coordinates of a point $P_3 = P_1 + P_2$ from the coordinates of P_1 and P_2. Scalar multiplication of a point is defined as the repeated addition of a point to itself n, e.g. $3P_1 = P_1 + P_1 + P_1$. O is the identity element.

The number of points of an elliptic curve $\mathbb{E}(\mathbb{F}_{q^k})$ is called the order of the curve over the field $\mathbb{F}_{q^k}$. A point P has order r if $rP = O$ for the smallest possible $r > 0$. The set of r-torsion points on $\mathbb{E}$ is the set $\mathbb{E}[r] = \{P \in \mathbb{E} \mid rP = O\}$. The order of a point always divides the curve order. There is an operation on a point in the extension field that will reduce that point to a point in the base field; this is called the trace map, and is denoted as $\mathrm{Tr}(P)$. One of these cyclic subgroups is called the *trace zero* subgroup, $\mathcal{T} = \{P \in \mathbb{E} \mid \mathrm{Tr}(P) = O\}$. A subgroup $\mathbb{G}$ of an elliptic curve is said to have embedding degree k if its order r divides $q^k - 1$ for the smallest possible k. We assume $k > 1$. We let $\mathbb{G}_0$ be the group of order r defined over $\mathbb{F}_q$ and $\mathbb{G}_1$ be the trace zero group, again of order r. The results of Weil and Tate pairing operations equate to one of the r-th roots of unity. Again this is a group of order r, we call this group $\mathbb{G}_2$ [12].

The modified Tate pairing over supersingular curves [5] denoted $\hat{t}(P, Q)$ is $t(P, \psi(P))$ where $t : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_2$ is the Tate pairing and $\psi : \mathbb{G}_0 \to \mathbb{G}_1$ is an efficiently computable distortion map [12]. It is an example of a bilinear map of the form $\hat{t} : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_2$ where $\mathbb{G}_0$ and $\mathbb{G}_2$ are groups of order r.

The possibility of exploiting differences between the pairings $\hat{t}(P, P)$ and $t(P, Q)$ to implement protocols with different properties has occurred to other authors [11, 18]. We use the modified Tate pairing in the escrowed system, and the Tate pairing in the escrowless system.

3 An Authenticated Key Agreement With Escrow

As with all other identity-based cryptosystems we assume the existence of a trusted Key Generation Centre (KGC) that is responsible for the creation and secure distribution of users private keys. This agreement algorithm can be implemented using the modified Tate pairing.

Setup: The KGC inputs a security parameter κ into a BDH parameter generator $\mathcal{B}_{mt}$ which returns two groups $\mathbb{G}_0$ and $\mathbb{G}_2$, both of prime order r, a suitable bilinear map $\hat{t} : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_2$ (which can be implemented as the modified Tate pairing), a generator element P such that $\langle P \rangle = \mathbb{G}_0$, and a random oracle $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_r^*$. The KGC randomly generates a master secret $s \in_R \mathbb{Z}_r^*$, and calculates a master public key sP. The parameters and master public key are distributed to the users of the system through a secure authenticated channel. We assume that the number of users is polynomial in κ.

Extract: The KGC checks that a user has a claim to a particular online identifier. If they do, the KGC generates their private key and communicates it privately to them. Let Alice's online identifier map to $a \in \mathbb{Z}_r^*$ by means of the random oracle $\mathcal{H}$. Alice's public key is $(a+s)P$, which can be computed as $aP + sP$. The KGC computes Alice's private key as $A_{pri} = (a+s)^{-1}P$. While it may be argued that this key pair derivation is not as elegant as that in the Boneh-Franklin IBE [4], since the public key no longer relies on the user's identity alone, most key agreements, except Scott's and Ryu *et al.*'s [16], also use the KGC's master secret in the key agreement stage.

Key Agreement: Assume that Alice and Bob have private keys issued by the same KGC, respectively A_{pri} and B_{pri}. Alice and Bob each generate one unique random nonce $x_a, x_b \in_R \mathbb{Z}_r^*$, respectively.

Alice		**Bob**
$A_{KA} = x_a(bP + sP)$	$\rightleftarrows$	$B_{KA} = x_b(aP + sP)$
$key_a = \hat{t}(B_{KA}, A_{pri})^{x_a}$		$key_b = \hat{t}(A_{KA}, B_{pri})^{x_b}$

This scheme is consistent because:

$$\begin{aligned} key_a &= \hat{t}(B_{KA}, A_{pri})^{x_a} \\ &= \hat{t}(P,P)^{x_a x_b} \\ &= \hat{t}(A_{KA}, B_{pri})^{x_b} \\ &= key_b. \end{aligned}$$

The escrow property derives from the KGC's ability to recover the shared session key by computing:

$$\begin{aligned} x_a P &= (s+b)^{-1} A_{KA}, \\ x_b P &= (s+a)^{-1} B_{KA}, \\ key &= \hat{t}(x_a P, x_b P). \end{aligned}$$

Our scheme is *role symmetric*, with each party performing the same operations and thus incurring the same computational cost.

4 An Authenticated Key Agreement Without Escrow

The key agreement without escrow differs only slightly from the algorithm given in section 3. Again there are three algorithms, **Setup**, **Extract** and **Key Agreement**. This key agreement protocol can be implemented using the conventional Tate pairing, not the modified Tate pairing as in the escrowed scheme.

Setup: The KGC inputs a security parameter κ into a BDH parameter generator $\mathcal{B}_t$ which returns three groups $\mathbb{G}_0, \mathbb{G}_1$ and $\mathbb{G}_2$, $\mathbb{G}_0$ and $\mathbb{G}_2$ being groups of prime order r, a suitable bilinear map $t : \mathbb{G}_0 \times \mathbb{G}_1 \to \mathbb{G}_2$ (which can be implemented as the Tate pairing), two generator elements P and Q such that $\langle P \rangle = \mathbb{G}_0$ and $\langle Q \rangle = \mathbb{G}_1$, and a random oracle $\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_r^*$. It is important that the discrete logarithm between $\psi(P)$ and Q is unknown[3]. This can be achieved by obtaining P and Q as the output of random oracles $\mathcal{H}_0 : \{0,1\}^* \to \mathbb{G}_0$ and $\mathcal{H}_1 : \{0,1\}^* \to \mathbb{G}_1$ evaluated on publicly known constant strings cs_0 and cs_1 (cs_0 and cs_1 may be the same string). The KGC randomly generates a master secret $s \in_R \mathbb{Z}_r^*$, and calculates a master public key sP. The parameters, master public key and the constant strings used in the derivation of P and Q are distributed to the users of the system through a secure authenticated channel. We assume that the number of users is polynomial in κ.

Extract: The KGC checks that a user has a claim to a particular online identifier. If they do, the KGC generates their private key and communicates it privately to them. Let Alice's online identifier map to $a \in \mathbb{Z}_r^*$ by means of the random oracle $\mathcal{H}$. Alice's public key is $A_{pub} = (a+s)P$, which can be computed as $aP + sP$. Alice's private key is generated as $A_{pri} = (a+s)^{-1}Q$. End user Alice is encouraged to check that the KGC has used the correct Q in the construction of her private key by checking the following:

$$\begin{aligned} P &\leftarrow \mathcal{H}_0(cs_0) \\ Q &\leftarrow \mathcal{H}_1(cs_1) \\ t(A_{pub}, A_{pri}) &\stackrel{?}{=} t(P, Q) \end{aligned}$$

Key Agreement: Assume that Alice and Bob have private keys issued by the same KGC, respectively A_{pri} and B_{pri}. Alice and Bob each generate one unique random nonce $x_a, x_b \in \mathbb{Z}_r^*$, respectively.

Alice		**Bob**
$A_{KA} = x_a(bP + sP)$	$\rightleftarrows$	$B_{KA} = x_b(aP + sP)$
$key_a = t(B_{KA}, A_{pri})^{x_a}$		$key_b = t(A_{KA}, B_{pri})^{x_b}$

[3] If the KGC knows λ such that $\psi(P) = \lambda Q$, it can use the distortion map to get a representation in $\langle Q \rangle$ of A_{KA} or B_{KA} and then recover the session key using the technique outlined in the previous section. On non-supersingular curves no efficiently computable distortion map exists [27] and this attack does not apply.

This scheme is consistent because

$$\begin{aligned} key_a &= t(B_{KA}, A_{pri})^{x_a} \\ &= t(P,Q)^{x_a x_b} \\ &= t(A_{KA}, B_{pri})^{x_b} \\ &= key_b. \end{aligned}$$

We also note that, although the KGC has the ability to generate the private keys of both users in the protocol, it is not able to obtain the shared session key for any particular run of the protocol. The KGC can, in this instance, easily compute $t(P,Q)^{x_a}$ and $t(P,Q)^{x_b}$, but calculating the key from these values involves solving the Computational Diffie-Hellman Problem (CDHP) over the group $\mathbb{G}_2$ [29].

5 Key Agreement Between Members of Distinct Domains

We now look at key agreements between members of separate domains. This idea was first suggested in [6]. We suggest a scheme that is twice as efficient as their scheme without precomputation, whilst being similar with precomputation. Again this protocol can be instantiated in escrowed or escrowless mode.

For key agreement to be possible between members of different groups all that is needed is for the points P, Q in the case of the escrowless system, or just P in the case of the escrowed system, and the curve description to be the same (standardised). Elliptic curves, suitable group generator points and other cryptographic tools have been standardised for non-IBE applications, for example in the NIST FIPS standards. It is reasonable, therefore, to assume the availability of standard pairing-friendly curves as well.

Once these group generator points and curves have been agreed upon, each KGC can generate their own random master secret.

Alice's private key is generated by KGC_1 with a master secret s_1. Bob's private key is generated by KGC_2 with a master secret s_2. Alice's public key is $(a+s_1)P$ and her private key is $A_{pri} = (a+s_1)^{-1}P$. Likewise, Bob's public key is $(b+s_2)P$ and his private key is $B_{pri} = (b+s_2)^{-1}P$. Notice that now Alice must obtain s_2P (the master public key of Bob's KGC) and vice-versa; it is critical that the master public keys are obtained in an authenticated manner, as with any IBC scheme.

Alice and Bob now perform the authenticated key agreement:

Alice		**Bob**
$A_{KA} = x_a(bP + s_2P)$	$\rightleftarrows$	$B_{KA} = x_b(aP + s_1P)$
$key_a = \hat{t}(B_{KA}, A_{pri})^{x_a}$		$key_b = \hat{t}(A_{KA}, B_{pri})^{x_b}$

This scheme is consistent because

$$\begin{aligned} key_a &= \hat{t}(B_{KA}, A_{pri})^{x_a} \\ &= \hat{t}(P, P)^{x_a x_b} \\ &= \hat{t}(A_{KA}, B_{pri})^{x_b} \\ &= key_b. \end{aligned}$$

6 Efficiency

Smart's protocol [25] requires each party to perform 2 point scalar multiplications and 2 pairing evaluations. One of these pairings can be partially precomputed, reducing the cost to 1 point scalar multiplication, 1 pairing evaluation and 1 pairing exponentiation per party at an additional storage cost of one pairing per recipient. Our new scheme achieves the same efficiency without incurring the extra storage requirements.

The Chen-Kudla authenticated key agreement protocol [6] requires 2 elliptic curve point scalar multiplications, 1 point addition and 1 pairing evaluation.

Scott's key agreement [18], using the pairing as a SPEKE generator, only requires two pairing exponentiations when precomputation is used. Again it restricts all users to having private keys generated by the same KGC.

The scheme proposed here requires 1 point scalar multiplication, 1 pairing exponentiation and one 1 pairing evaluation. We note that a pairing exponentiation is quicker than a point scalar multiplication.

We also note that the method of generating public keys from identities — namely, by mapping identities to integer coefficients and performing a scalar multiplication — is faster than the technique used in Boneh-Franklin key pair generation. Their technique involves mapping the identifier to a coordinate, solving the curve equation and then multiplying by a large cofactor to generate a point of order r. Public keys in our system will always be points of order r.

In Smart's protocol the recipient's public key is used either explicitly or implicitly (if pairings are precomputed) to complete the protocol. In our scheme, public keys of form $uP + sP$ may be stored to save one scalar multiplication, with the advantage that such values require a much smaller storage space than pairing values, namely, a fraction[4] $1/k$ where k is the embedding degree of the curve $E(\mathbb{F}_q)$.

We leave public key generation out of the following complexity analysis as it is only slightly faster for our system — and can be precomputed in all IBE systems. We also leave out $E(\mathbb{F}_{q^k})$ multiplication, point addition and hashing as they are fast to compute compared to the other principle operations.

key: *p = pairing evaluation, e = $E(\mathbb{F}_{q^k})$ (pairing) exponentiation, m = scalar multiplication, n = number of recipients, s = storage space per pairing evaluation, rac = requires additional computation (two point multiplications).*

[4] If pairing compression techniques as described in [19] are used, the fraction is $2/k$ in general or $3/k$ in a special case.

	Proposed	Smart	Chen-Kudla	Scott
No Precomp	$\mathbf{1p+1e+1m}$	$2p+1m$	$1p+2m$	$1p+2e$
Precomp	$\mathbf{1p+1e+1m}$	$1p+1e+1m+ns$	$1p+1e+1m+ns$	$2e$
Escrow	***Yes / No***	*Yes / No (rac)*	*Yes / No (rac)*	*No*
Between Domains	***Yes***	*No*	*No*	*No*

7 Security of the proposed scheme

The proof of security of the above algorithm relies on the conjectured intractability of a problem which Zhang *et al.* [30] call the **Bilinear Inverse Diffie-Hellman Problem**: For $\alpha, \beta \in \mathbb{Z}_r^*$, given P, αP, βP, compute $v = \hat{t}(P,P)^{\alpha^{-1}\beta}$.

7.1 The security of the authentication mechanism

Assuming that the BIDHP is hard (with respect to the security parameter κ), we now show the security of the above protocols.

We adopt the security model proposed by Bellare and Rogaway [2], modified by Blake-Wilson *et al.* [3], and used in proving the security of the key agreement protocol introduced in [6] and others.

The model includes a set of parties, each modelled by an oracle. We use the notation $\prod_{i,j}^n$, meaning a participant/oracle i believing that it is participating in the n-th run of the protocol with j. Oracles keep transcripts of all communications in which they have been involved. Each oracle has a secret private key, issued by a KGC, which has run a BDH parameter generator $\mathcal{B}$ and published groups $\mathbb{G}_0$ and $\mathbb{G}_2$, a bilinear map of the form $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_2$, a group generator P of $\mathbb{G}_0$, and a master public key sP.

The model contains and adversary E which has access to all message flows in the system. E is not a user or KGC. All oracles only communicate with each other via E. E can replay, modify, delay, interleave or delete messages. E is benign if it acts like a wire and does not modify communication between oracles. From [2], if two oracles receive, via the adversary, property formatted messages that have been generated exclusively by the other oracle, and both oracles accept, we say that these two oracles have had a matching conversation.

The adversary at any time can make the following queries:

Create E sets up a new oracle in the system that has public key ID, of E's choosing. E has access to the identity / public key of the oracle. The private key is obtained from the KGC.

Send E sends a message of his choice to an oracle i, $\prod_{i,j}^n$, in which case i assumed that the message came from j. E can also instruct the actual oracle j to start a new run of the protocol with i by sending a λ. Using the terminology of [6] an oracle is an *initiator oracle* if the first message that it receives is a λ, otherwise it is a *responder oracle*.

Reveal E receives the session key that is currently being held by a particular oracle.

Corrupt E receives the long term asymmetric private key being held by a particular oracle.

Test E receives either the session key or a random value from a particular oracle. Specifically, to answer the query the oracle flips a fair coin $c \in \{0,1\}$; if the answer is 0 it outputs the agreed session key, and if the answer is 1 it outputs a random element of $\mathbb{G}_2$. E then must decide whether c is 0 or 1; call this prediction c'. E's advantage in distinguishing the actual session key held by an uncorrupted party from a key sampled at random from $\mathbb{G}_2$ in this game, with respect to the security parameter κ, is given by:

$$Advantage^E(\kappa) = |Pr[c' = c] - 1/2|$$

The Test query can be performed only once, against an oracle that is in the Accepted state (see below), and which has not previously been asked a Reveal or Corrupt query.

An oracle may be in one of the following states (it cannot be in more than one state).

Accepted If the oracle decides to accept a session key, after receipt of properly formated messages.
Rejected If the oracle decides to not to accept and aborts the run of the protocol.
* If the oracle has yet to decide whether to accept to reject for this run of the protocol. We assume that there is some time out on this state.
Opened If a Reveal query has been performed against this oracle for its last run of the protocol (its current session key is revealed).
Corrupted If a Corrupt query has ever been performed against this oracle.

Definition 1.
A protocol is an AK protocol if:

- *In the presence of the benign adversary on $\prod_{i,j}^{n}$ and $\prod_{j,i}^{t}$, both oracles always accept holding the same session key, and this key is distributed uniformly at random on $\mathbb{G}_2$; if for every adversary E:*
- *If uncorrupted oracles $\prod_{i,j}^{n}$ and $\prod_{j,i}^{t}$, have matching conversations then both oracles accept and hold the same session key;*
- *$Advantage^E(\kappa)$ is negligible.*

Theorem 1. *The proposed key agreement protocol is a secure AK protocol.*

Proof. Condition 1 holds as follows: Both oracles accept holding the same session key as a direct result of the commutativity of exponentiation of members of the group $\mathbb{G}_2$. The session key is distributed uniformly at random by the fact that both oracles generate truly random $x \in \mathbb{Z}$. Therefore the product of these elements will also be random. Since the exponent is random, and $e(P,P)$ is a generator of the group $\mathbb{G}_2$, the session key will be uniformly distributed over $\mathbb{G}_2$.

Condition 2 holds by the fact that if they have matching conversations then the communication was generated entirely by the two oracle's. Therefore, by the

bilinearity of the pairing and the commutativity of exponentiation they accept and hold the same session key.

Condition 3 holds as follows: Consider by contradiction that $Advantage^E(\kappa)$ is non-negligible. Then we can construct from E an algorithm $\mathcal{F}$ that solves the BIDHP with non-negligible advantage. $\mathcal{F}$ is given as input the output of the BDH generator $\mathcal{B}$. $\mathcal{F}$'s task is to solve the BIDHP, namely, given P, αP and βP, compute $v = \hat{t}(P,P)^{\alpha^{-1}\beta}$.

All queries by the adversary E now pass through $\mathcal{F}$.

Create For each oracle $\mathcal{F}$ chooses $y_i \in_R \mathbb{Z}_p^*$, creates a public key as $u_i P = (y_i P - sP)$, and computes the private key as $y_i^{-1} P$. Obviously $y_i P = u_i P + sP$. However, for the j-th oracle $\mathcal{F}$ answers αP. Since $\mathcal{F}$ does not know α, it cannot calculate $\alpha^{-1} P$, the correct private key for this oracle.

Corrupt $\mathcal{F}$ answers Corrupt queries in the usual way, revealing the private key of the oracle being queried. However, $\mathcal{F}$ does not know the private key for oracle j; if E asks a Corrupt query on oracle j, $\mathcal{F}$ gives up.

Send $\mathcal{F}$ answers all send queries in the usual way, except if E asks Send $\prod_{i,j}^n$, $\mathcal{F}$ answers $x_i P$, for a known x_i, which is, from E's perspective, indistinguishable from $x_t(\alpha P)$ for a random $x_t \in_R \mathbb{Z}_q^*$. In response it will get a value from j, this is set as the value βP — this is a genuine value from j and $\mathcal{F}$ does not influence it.

Test At some point E will ask a single Test query of some oracle, which we assume is oracle j; if it is not, $\mathcal{F}$ aborts. The chance of $\mathcal{F}$ picking j is $\xi = 1/n$ where n is the number of oracles (Create queries). Since it is picked it must have accepted and it must be holding a session key of the form $e(P,P)^{x_i x_j}$. However, $\mathcal{F}$ cannot compute this key and hence cannot simulate the query, so it simply outputs a random element of the group $\mathbb{G}_2$.

If $\mathcal{F}$ does not abort and E does not detect $\mathcal{F}$'s inconsistency in answering the Test query then its advantage in predicting the correct session key is $Advantage^E(\kappa)$ as before. For this to be non-negligible, E must have some advantage in calculating $e(P,P)^{x_i x_j}$, given $\beta P = x_j \alpha P$ as input from j.

If E does not detect any inconsistencies in $\mathcal{F}$'s responses, then $\mathcal{F}$ must have non-negligible advantage $A(\kappa)$ in calculating $e(P,P)^{x_i x_j}$, but, since $\mathcal{F}$ does not know j's private key the session key was calculated as $e(P,P)^{\alpha^{-1}\beta x_i}$. Provided that $\mathcal{F}$ is able to calculate $e(P,P)^{\alpha^{-1}\beta x_i}$, it can calculate $e(P,P)^{\alpha^{-1}\beta}$ since it knows x_i.

We assume that there is some timeout τ_s on the length of a run of the protocol, including the time spent in the $*$ state. We also assume that some time τ_c is allocated to allow the construction of oracles in the Create query, and time τ_o allocated for each Corrupt query. We assume that n oracles are needed, and that m send queries are needed, and o corrupt queries are needed. The expected time needed to solve the BIDHP is:

$$\frac{(n\tau_c)(m\tau_s)(o\tau_o)\xi}{A(\kappa)}$$

We note that our protocol is vulnerable to an attack described by Blake-Wilson *et al.* [3], namely, that an active adversary can offset the agreed session key by an exponent ϵ unbeknownst to Alice or Bob. Most key agreements without key confirmation are vulnerable to this attack, for example, those by Chen and Kudla, Smart and Scott. The attack is shown below, with E, being an active attacker.

Alice	E	Bob
$K_A = x_a(s+b)P$	$\rightarrow K'_A = \epsilon K_A \rightarrow$	
	$\leftarrow K'_B = \epsilon K_B \leftarrow$	$K_B = x_b(s+a)P$
$key = e(K'_B, A_{pri})^{x_a}$		$key = e(K'_A, B_{pri})^{x_b}$
$= e(P,P)^{x_a x_b \epsilon}$		$= e(P,P)^{x_a x_b \epsilon}$

Table 1. Key offset attack

Although this attack (which exists against many key agreements) is interesting, it should be noted that it does not allow the attacker to gain any knowledge of the agreed session key.

7.2 Further security considerations

Here we look at the new key agreement using a few security definitions that are often used to judge key agreements. We only consider the basic protocol given in section 3.

Known Key Security: If one session key is compromised this does not mean that any other session keys are compromised. This is from the fact that the agreed session keys rely on random ephemeral keys. A session key as a result is distributed uniformly in $\mathbb{G}_2$ with no connection to other session keys.

Key-Compromise Impersonation: The above protocol is affected by Key-Compromise Impersonation as pointed out first in [8] and later in [28]. This attack proceeds as in table 2, assuming that Bob has a copy of Alice's private key $A_{pri} = (s+a)^{-1}P$:

Alice		**Bob**
$A_{KA} = x_a(bP + sP)$	$\rightleftarrows$	$B_{KA} = x_b(bP + sP)$
$key_a = \hat{t}(B_{KA}, A_{pri})^{x_a}$		$key_b = \hat{t}(A_{KA}, A_{pri})^{x_b}$
$key_a = \hat{t}(P,P)^{x_a x_b (b+s)(a+s)^{-1}}$		$key_b = \hat{t}(P,P)^{x_a x_b (b+s)(a+s)^{-1}}$

Table 2. Key-Compromise Impersonation attack

However this can be easily solved, with no additional overhead, if the shared key is calculated as $\hat{t}(PP)^{x_a+x_b}$ instead. This is achieved as shown in table 3.

Alice		**Bob**
$A_{KA} = x_a(bP + sP)$	$\rightleftarrows$	$B_{KA} = x_b(aP + sP)$
$key_a = \hat{t}(P,P)^{x_a} \cdot \hat{t}(B_{KA}, A_{pri})$		$key_b = \hat{t}(A_{KA}, B_{pri}) \cdot \hat{t}(P,P)^{x_b}$
$key_a = \hat{t}(P,P)^{x_a+x_b}$		$key_b = \hat{t}(P,P)^{x_a+x_b}$

Table 3. A variant of the Key Agreement resistant to Key-Compromise Impersonation

Again this scheme requires one point scalar multiplication, one pairing exponentiation and one pairing exponentiation (pairing multiplication in not included in the efficiency analysis as it is extremely fast).

Unknown Key-Share Resilience: Alice cannot be coerced into sharing a key with Charlie thinking she is sharing a key with Bob. Again, this come from the fact that Alice explicitly uses Bob's public key in her contribution to the session key.

Forward Secrecy: Compromise of either Alice's private key or Bob's private key does not appear to allow an attacker to recover any past session keys. On the other hand, compromise of the KGC's master secret in the escrowed scheme allows all past agreed session keys to be recovered.

Key Control: Because both parties have an input into the key, neither entity is able to force the full session key to be a preselected value. However, Bob can set certain bits of the agreed session key by carefully selecting his ephemeral key x_b until be achieves the desired result. It does not appear possible for Bob to set any substantial number of bits in a reasonable time frame. Again, this key agreement is no less secure in this respect that most other key agreements. As with all key agreements a short timeout on a particular run of the protocol may be advisable.

8 Conclusion

We have presented a new ID-based key agreement protocol inspired on the Sakai-Kasahara key pair generation algorithm. The proposed scheme improves on the performance of the Smart and the Chen-Kudla key agreement protocols, can be instantiated in either escrowed or escrowless mode, and can be carried out by clients of distinct KGC's.

References

1. S. S. Al-Riyami and K. G. Paterson. Tripartite authenticated key agreement protocols from pairings. In *IMA Conference on Cryptography and Coding*, volume 2898 of *Lecture Notes in Computer Science*, pages 332–359. Springer-Verlag, 2003.
2. M. Bellare and P. Rogaway. Entity authentication and key distribution. In *Advances in Cryptology – Crypto'93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer-Verlag, 1994.
3. S. Blake-Wilson, D. Johnson, and A. Menezes. Key agreement protocols and their security analysis. In *IMA International Conference on Cryptography and Coding*, volume 1355 of *Lecture Notes in Computer Science*, pages 30–45. Springer-Verlag, 1997.
4. D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *Advances in Cryptology – Crypto'2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer-Verlag, 2001.
5. L. Chen and K. Harrison. Multiple trusted authorities in identifier based cryptography from pairings on elliptic curves. Trusted Systems Laboratory, HP, 2003. `http://www.hpl.hp.com/techreports/2003/HPL-2003-48.pdf`.
6. L. Chen and C. Kudla. Identity based authenticated key agreement from pairings. Cryptology ePrint Archive, Report 2002/184, 2002. `http://eprint.iacr.org/2002/184`.
7. Z. Chen. Security analysis on Nalla-Reddy's ID-based tripartite authenticated key agreement protocols. Cryptology ePrint Archive, Report 2003/103, 2003. `http://eprint.iacr.org/2003/103`.
8. Michael Cheng. University of Middlesex, London, UK. Personal Communication, 2004.
9. C. Cocks. An identity based encryption scheme based on quadratic residues. In *VIII IMA International Conference on Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer-Verlag, 2001. `"http://www.cesg.gov.uk/site/ast/idpkc/media/ciren.pdf`.
10. R. Dutta, R. Barua, and P. Sarkar. Pairing-based cryptography : A survey. Cryptology ePrint Archive, Report 2004/064, 2004. `http://eprint.iacr.org/2004/064`.
11. S. Galbraith. Personal communication, 2004.
12. S. Galbraith and V. Rotger. Easy decision-diffie-hellman groups. Cryptology ePrint Archive, Report 2004/070, 2004. `http://eprint.iacr.org/2004/070`.
13. A. Joux. A one round protocol for tripartite Diffie-Hellman. In *Proceedings of Algorithmic Number Theory Symposium*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer-Verlag, 2000.
14. D. Nalla. ID-based tripartite key agreement with signatures. Cryptology ePrint Archive, Report 2003/144, 2003. `http://eprint.iacr.org/2003/144`.
15. D. Nalla and K. C. Reddy. ID-based tripartite authenticated key agreement protocols from pairings. Cryptology ePrint Archive, Report 2003/004, 2003. `http://eprint.iacr.org/2003/004`.
16. Eun-Kyung Ryu, Eun-Yoon, and Kee-Young Yoo. An efficient ID-based autenticated key agreement protocol from pairings. In *NETWORKING 2004*, volume 3042 of *Lecture Notes in Computer Science*, pages 1458–1463. Springer-Verlag, 2004.
17. R. Sakai and M. Kasahara. ID based cryptosystems with pairing on elliptic curve. In *2003 Symposium on Cryptography and Information Security – SCIS'2003*, Hamamatsu, Japan, 2003. `http://eprint.iacr.org/2003/054`.

18. M. Scott. Authenticated ID-based key exchange and remote log-in with insecure token and PIN number. Cryptology ePrint Archive, Report 2002/164, 2002. http://eprint.iacr.org/2002/164/.
19. M. Scott and P. S. L. M. Barreto. Compressed pairings. In *Advances in Cryptology – Crypto'2004*, volume 3152 of *Lecture Notes in Computer Science*. Springer-Verlag, 2004. to appear.
20. A. Shamir. Identity based cryptosystems and signature schemes. In *Advances in Cryptology – Crypto'84*, volume 0196 of *Lecture Notes in Computer Science*, pages 47–53. Springer-Verlag, 1984.
21. K. Shim. Cryptanalysis of Al-Riyami-Paterson's authenticated three party key agreement protocols. Cryptology ePrint Archive, Report 2003/122, 2003. http://eprint.iacr.org/2003/122.
22. K. Shim. Cryptanalysis of ID-based tripartite authenticated key agreement protocols. Cryptology ePrint Archive, Report 2003/115, 2003. http://eprint.iacr.org/2003/115.
23. K. Shim. Efficient ID-based authenticated key agreement protocol based on Weil pairing. *Electronics Letters*, 39(8):653–654, 2003.
24. K. Shim. Efficient one round tripartite authenticated key agreement protocol from Weil pairing, 2003.
25. N. P. Smart. An identity based authenticated key agreement protocol based on the Weil pairing. *Electronics Letters*, 38:630–632, 2002.
26. H.-M. Sun and B.-T. Hsieh. Security analysis of Shim's authenticated key agreement protocols from pairings. Cryptology ePrint Archive, Report 2003/113, 2003. http://eprint.iacr.org/2003/113.
27. E. Verheul. Evidence that XTR is more secure than supersingular elliptic curve cryptosystems. In *Advances in Cryptology – Eurocrypt'2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 195–210. Springer-Verlag, 2001.
28. G. Xie. Cryptanalysis of noel mccullagh and paulo s.l.m. barreto's two party identity based key agreement. Cryptology ePrint Archive, Report 2004/308, 2004. http://eprint.iacr.org/2004/308.
29. Y. Yacobi. A note on the bilinear Diffie-Hellman assumption. Cryptology ePrint Archive, Report 2002/113, 2002. http://eprint.iacr.org/2002/113.
30. F. Zhang, R. Safavi-Naini, and W. Susilo. An efficient signature scheme from bilinear pairings and its applications. In *International Workshop on Practice and Theory in Public Key Cryptography – PKC'2004*, Lecture Notes in Computer Science, pages 277–290. Springer-Verlag, 2004.