Making the Diffie-Hellman Protocol Identity-Based*

Dario Fiore[1,**] and Rosario Gennaro[2]

[1] Dipartimento di Matematica ed Informatica – Università di Catania, Italy
fiore@dmi.unict.it
[2] IBM T. J. Watson Research Center – Hawthorne, New York 10532
rosario@us.ibm.com

Abstract. This paper presents a new identity based key agreement protocol. In id-based cryptography (introduced by Adi Shamir in [29]) each party uses its own identity as public key and receives his secret key from a master Key Generation Center, whose public parameters are publicly known.

The novelty of our protocol is that it can be implemented over any cyclic group of prime order, where the Diffie-Hellman problem is supposed to be hard. It does not require the computation of expensive bilinear maps, or additional assumptions such as factoring or RSA.

The protocol is extremely efficient, requiring only twice the amount of bandwith and computation of the *unauthenticated* basic Diffie-Hellman protocol. The design of our protocol was inspired by MQV (the most efficient authenticated Diffie-Hellman based protocol in the public-key model) and indeed its performance is competitive with respect to MQV (especially when one includes the transmission and verification of certificates in the MQV protocol, which are not required in an id-based scheme). Our protocol requires a single round of communication in which each party sends only 2 group elements: a very short message, especially when the protocol is implemented over elliptic curves.

We provide a full proof of security in the Canetti-Krawczyk security model for key exchange, including a proof that our protocol satisfies additional security properties such as forward secrecy, and resistance to reflection and key-compromise impersonation attacks.

1 Introduction

Identity-based cryptography was introduced in 1984 by Adi Shamir [29]. The goal was to simplify the management of public keys and in particular the association of a public key to the identity of its holder. Usually such binding of a public key to an identity is achieved by means of *certificates* which are signed statements by trusted third parties that a given public key belongs to a user. This requires users

* A full version of this paper is available at http://eprint.iacr.org/2009/174
** Work done while visiting NYU and IBM Research.

J. Pieprzyk (Ed.): CT-RSA 2010, LNCS 5985, pp. 165–178, 2010.

to obtain and verify certificates whenever they want to use a specific public key, and the management of public key certificates remains a technically challenging problem.

Shamir's idea was to allow parties to use their identities as public keys. An id-based scheme works as follows. A trusted *Key Generation Center* (KGC) generates a master public/secret key pair, which is known to all the users. A user with identity ID receives from the KGC a secret key S_{ID} which is a function of the string ID and the KGC's secret key (one can think of S_{ID} as a signature by the KGC on the string ID). Using S_{ID} the user can then perform cryptographic tasks. For example in the case of *id-based encryption* any party can send an encrypted message to the user with identity ID using the string ID as a public key and the user (and only the user and the KGC) will be able to decrypt it using S_{ID}. Note that the sender can do this even if the recipient has not obtained yet his secret key from the KGC. All the sender needs to know is the recipient's identity and the public parameters of the KGC. This is the major advantage of id-based encryption.

Id-Based Key Agreement and its Motivations. This paper is concerned with the task of *id-based key agreement*. Here two parties Alice and Bob, with identities A, B and secret keys S_A, S_B respectively, want to agree on a common shared key, in an *authenticated* manner (i.e. Alice must be sure that once the key is established, only Bob knows it – and viceversa). Since key agreement is inherently an interactive protocol (both parties are "live" and ready to establish a session) there is a smaller gain in using an id-based solution: indeed certificates and public keys can be easily sent as part of the protocol communication.

Yet the ability to avoid sending and verifying public key certificates is a significant practical advantage (see e.g. [32]). Indeed known shortcomings of the public key setting are the requirement of centralized certification authorities, the need for parties to cross-certify each other (via possibly long certificate chains), and the management of some form of large-scale coordination and communication (possibly on-line) to propagate certificate revocation information. Identity-based schemes significantly simplify identity management by bypassing the certification issues. All a party needs to know in order to generate a shared key is its own secret key, the public information of the KGC, and the identity of the communication peer (clearly, the need to know the peer's identity exists in any scheme including a certificate-based one).

Another advantage of identity-based systems is the versatility with which identities may be chosen. Since identities can be arbitrary string, they can be selected according to the function and attributes of the parties (rather than its actual "name"). For example in vehicular networks a party may be identified by its location ("the checkpoint at the intersection of a and b") or in military applications a party can be identified by its role ("platoon x commander"). This allows parties to communicate securely with the intended recipient even without knowing its "true" identity but simply by the definition of its function in the network.

Finally, identities can also include additional attributes which are temporal in nature: in particular an "expiration date" for an identity makes revocation of the corresponding secret key much easier to achieve.

For the reasons described above, id-based KA protocols are very useful in many systems where bandwith and computation are at a premium (e.g. sensor networks), and also in ad-hoc networks where large scale coordination is undesirable, if not outright impossible. Therefore it is an important question to come up with very efficient and secure id-based KA protocols.

Previous work on id-based key agreement. Following Shamir's proposal of the concept of id-based cryptography, some early proposals for id-based key agreement appeared in the literature: we refer in particular to the works of Okamoto [24] (later improved in [25]) and Gunther [18]. A new impetus to this research area came with the breakthrough discovery of bilinear maps and their application to id-based encryption in [4]: starting with the work of Sakai *et al.* [28] a large number of id-based KA protocols were designed that use pairings as tool. We refer the readers to [5] and [10] for surveys of these pairing-based protocols.

The main problem with the current state of the art is that many of these protocols lack a proof of security, and some have even been broken. Indeed only a few (e.g., [7,33]) have been proven according to a formal definition of security.

Our Contribution. By looking at prior work we see that provably secure id-based KAs require either groups that admit bilinear maps [7,33], or to work over a composite RSA modulus [25].

This motivated us to ask the following question: can we find an efficient and provably secure id-based KA protocol such that:

1. It that can be implemented over *any* cyclic group in which the Diffie-Hellman problem is supposed to be hard. The advantages of such a KA protocol would be several, in particular: (i) it would avoid the use of computationally expensive pairing computations; (ii) it could be implemented over much smaller groups (since we could use 'regular' elliptic curves, rather than the ones that admit efficient pairings computations for high security levels, or the group Z_N^* for a composite N needed for Okamoto-Tanaka).
2. It is more efficient than any KA protocols in the public key model (such as MQV [22]), when one includes the transmission and verification of certificates which are not required in an id-based scheme. This is a very important point since, as we pointed out earlier in this Section, id-based KA protocols are only relevant if they outperform PKI based ones in efficiency.

Our new protocol presented in this paper, achieves all these features. It can be implemented over any cyclic group over which the Diffie-Hellman problem is assumed to be hard. In addition it requires an amount of bandwith and computation similar to the *unauthenticated* basic Diffie-Hellman protocol. Indeed our new protocol requires a single round of communication in which each party sends just two group elements (as opposed to one in the Diffie-Hellman

protocol). Each party must compute four exponentiations to compute the session key (as opposed to two in the Diffie-Hellman protocol).

A similar favorable comparison holds with the Okamoto-Tanaka protocol in [25]. While that protocol requires only two exponentiations, it works over Z_N^* therefore requiring the use of a larger group size, which almost totally absorbs the computational advantage, and immediately implies a much larger bandwith requirement. Detailed efficiency comparisons to other protocols in the literature are discussed in Section 5.

We present a full proof of security of our protocol in the Canetti-Krawczyk security model. Our results hold in the random oracle model, under the Strong Diffie-Hellman Assumption. We also present some variations of our protocol that can be proven secure under the basic Computational Diffie-Hellman Assumption. Our protocol can be proven to satisfy additional desirable security properties such as perfect forward secrecy[1], and resistance to reflection and key-compromise impersonation attacks.

Our Approach. The first direction we took in our approach was to attempt to analyze the id-based KA protocols by Gunther [18] and Saeednia [27]. They also work over any cyclic group where the Diffie-Hellman problem is assumed to be hard, but lack a formal proof of security. While the original protocols cannot be shown to be secure, we were able to prove the security of modified versions of them. Nevertheless these two protocols were not very satisfactory solutions for the problem we had set out to solve, particularly for reasons of efficiency since they required a large number of exponentiations, which made them less efficient than say MQV with certificates. The security analysis of these modified Gunther and Saeednia protocols will be included in the final version.

Our protocol improves over these two protocols by using Schnorr's signatures [30], rather than ElGamal, to issue secret keys to the users. The simpler structure of Schnorr's signatures permits a much more efficient computation of the session key, resulting in less exponentiations and a single round protocol. Our approach was inspired by the way the MQV protocol [22] achieves *implicit authentication* of the session key. Indeed our protocol can be seen as an id-based version of the MQV protocol.

2 Preliminaries

In this section we present some standard definitions needed in the rest of the paper.

Let $\mathbb{N}$ the set of natural numbers. We will denote with $\ell \in \mathbb{N}$ the security parameter. The partecipants to our protocols are modeled as probabilistic Turing machines whose running time is bounded by some polynomial in ℓ. If S is a set, we

[1] We can prove PFS only in the case the adversary was passive in the session that he is attacking – though he can be active in other sessions. As proven by Krawczyk in [21], this is the best that can be achieved for 1-round protocols with *implicit* authentication, such as ours.

denote with $s \xleftarrow{\$} S$ the process of selecting an element uniformly at random from S. A function is said to be *negligible* if it vanishes faster than any polynomial.

The security of our protocol is based on the Strong Diffie-Hellman Assumption (SDH) [1], which is a variant of the standard Computational Diffie-Hellman (CDH) [13] where the adversary is provided with an oracle that solves the decisional problem.

Our new protocol is proven secure in the Canetti-Krawczyk (CK) [8,9] model for key agreement, adapted to the identity-based setting. For lack of space we defer the description of the assumptions and the model to the full version of the paper.

3 The New Protocol IB-KA

Protocol setup. The Key Generation Center (KGC) chooses a group $\mathbb{G}$ of prime order q (where q is ℓ-bits long), a random generator $g \in \mathbb{G}$ and two hash functions $H_1 : \{0,1\}^* \to \mathbb{Z}_q$ and $H_2 : \mathbb{Z}_q \times \mathbb{Z}_q \to \{0,1\}^\ell$. Then it picks a random $x \xleftarrow{\$} \mathbb{Z}_q$ and sets $y = g^x$. Finally the KGC outputs the public parameters $MPK = (\mathbb{G}, g, y, H_1, H_2)$ and keeps the master secret key $MSK = x$ for itself.

Key Derivation. A user with identity ID receives, as its secret key, a Schnorr's signature [30] of the message $m = ID$ under public key y. More specifically, the KGC after verifying the user's identity, creates the associated secret key as follows. First it picks a random $k \xleftarrow{\$} \mathbb{Z}_q$ and sets $r_{ID} = g^k$. Then it uses the master secret key x to compute $s_{ID} = k + H_1(ID, r_{ID})x$. (r_{ID}, s_{ID}) is the secret key returned to the user. The user can verify the correctness of its secret key by using the public key y and checking the equation $g^{s_{ID}} \stackrel{?}{=} r_{ID} \cdot y^{H_1(ID, r_{ID})}$.

A protocol session. Let's assume that Alice wants to establish a session key with Bob. Alice owns secret key (r_A, s_A) and identity A while Bob has secret key (r_B, s_B) and identity B.

Alice selects a random $t_A \xleftarrow{\$} \mathbb{Z}_q$, computes $u_A = g^{t_A}$ and sends the message $\langle A, r_A, u_A \rangle$ to Bob. Analogously Bob picks a random $t_B \xleftarrow{\$} \mathbb{Z}_q$, computes $u_B = g^{t_B}$ and sends $\langle B, r_B, u_B \rangle$ to Alice. After the parties have exchanged these two messages, they are able to compute the same session key $Z = H_2(z_1, z_2)$. In particular Alice computes

$$z_1 = (u_B r_B y^{H_1(B, r_B)})^{t_A + s_A} \quad \text{and} \quad z_2 = u_B^{t_A}.$$

On the other hand Bob computes

$$z_1 = (u_A r_A y^{H_1(A, r_A)})^{t_B + s_B} \quad \text{and} \quad z_2 = u_A^{t_B}.$$

It is easy to see that both the parties are computing the same values $z_1 = g^{(t_A + s_A)(t_B + s_B)}$ and $z_2 = g^{t_A t_B}$. The state of a user ID during a protocol session contains only the fresh random exponent t_{ID}. We assume that after a session is completed, the parties erase their state and keep only the session key.

Remark: In the next section we show that protocol IB-KA is secure under the Strong Diffie-Hellman Assumption. However, in the full version of the paper we

show how to modify IB-KA to obtain security under the basic CDH Assumption, at the cost of a slight degradation in efficiency.

4 Security Proof

We prove the security of the protocol by a usual reduction argument. More precisely we show how to reduce the existence of an adversary breaking the protocol into an algorithm that is able to break the SDH Assumption with non-negligible probability. The adversary is modeled as a CK attacker: in particular it will choose a test session among the complete and unexposed sessions and will try to distinguish between its real session key and a random one.

In our reduction we will make use of the General Forking Lemma, stated by Bellare and Neven in [2]. It follows the original forking lemma of Pointcheval and Stern [26], but, unlike that, it makes no mention of signature schemes and random oracles. In this sense it is more general and it can be used to prove the security of our protocol. We briefly recall it in the following.

Lemma 1 (General Forking Lemma [2]). *Fix an integer $Q \geq 1$ and a set H of size $|H| \geq 2$. Let $\mathcal{B}$ be a randomized algorithm that on input $x, h_1, \ldots, h_Q$ returns a pair (J, σ) where $J \in \{0, \ldots, Q\}$ and σ is referred as* side output*. Let IG be a randomized algorithm called the input generator. Let $acc_{\mathcal{B}} = \Pr[J \geq 1 : x \stackrel{\$}{\leftarrow} IG, h_1, \ldots, h_Q \stackrel{\$}{\leftarrow} H; (J, \sigma) \stackrel{\$}{\leftarrow} \mathcal{B}(x, h_1, \ldots, h_Q)]$ be the accepting probability of $\mathcal{B}$.*

The forking algorithm $F_{\mathcal{B}}$ associated to $\mathcal{B}$ is the randomized algorithm that takes in input x and proceeds as follows:

Algorithm $F_{\mathcal{B}}(x)$
Pick random coins ρ for $\mathcal{B}$
$h_1, \ldots, h_Q \stackrel{\$}{\leftarrow} H$
$(J, \sigma) \stackrel{\$}{\leftarrow} \mathcal{B}(x, h_1, \ldots, h_Q; \rho)$
If $J = 0$ then return $(0, \perp, \perp)$
$h'_1, \ldots, h'_Q \stackrel{\$}{\leftarrow} H$
$(J', \sigma') \stackrel{\$}{\leftarrow} \mathcal{B}(x, h_1, \ldots, h_{J-1}, h'_J, \ldots, h'_Q, ; \rho)$
If ($J = J'$ and $h_J \neq h'_J$) then return $(1, \sigma, \sigma')$
Else return $(0, \perp, \perp)$.

Let $frk = \Pr[b = 1 : x \stackrel{\$}{\leftarrow} IG; (b, \sigma, \sigma') \stackrel{\$}{\leftarrow} F_{\mathcal{B}}(x)]$. Then $frk \geq acc_{\mathcal{B}}(\frac{acc_{\mathcal{B}}}{Q} - \frac{1}{|H|})$.

Theorem 1. *Under the Strong-DH Assumption, if we model H_1 and H_2 as random oracles, then protocol* IB-KA *is a secure identity-based key agreement protocol.*

Proof. For sake of contradiction let us suppose there exists a PPT adversary $\mathcal{A}$ that has non-negligible advantage ϵ into breaking the protocol IB-KA , then we show how to build a solver algorithm S for the CDH problem.

In our reduction we will proceed into two steps. First, we describe an intermediate algorithm $\mathcal{B}$ (i.e. the simulator) that interacts with the IB-KA adversary $\mathcal{A}$ and returns a side output σ. Second, we will show how to build an algorithm S that exploits $F_{\mathcal{B}}$, the forking algorithm associated with $\mathcal{B}$, to solve the CDH problem under the Strong-DH Assumption.

$\mathcal{B}$ receives in input a tuple $(\mathbb{G}, g, U, V)$, where $U = g^u, V = g^v$ and u, v are random exponents in $\mathbb{Z}_q$, and a set of random elements $h_1, \ldots, h_Q \in \mathbb{Z}_q$. The simulator is also given access to a DH oracle $\mathsf{DH}(U, \cdot, \cdot)$ that on input $(\hat{V}, \hat{W})$ answers "yes" if $(U, \hat{V}, \hat{W})$ is a valid DDH tuple . The side output of $\mathcal{B}$ is $\sigma \in \mathbb{G} \times \mathbb{Z}_q$ or $\perp$. Let n be an upper bound to the number of sessions of the protocol run by the adversary $\mathcal{A}$ and Q_1 and Q_2 be the number of queries made by $\mathcal{A}$ to the random oracles H_1, H_2 respectively. Moreover, let Q_c be the number of users corrupted by $\mathcal{A}$ and $Q = Q_1 + Q_c + 1$.

Algorithm $\mathcal{B}^{\mathsf{DH}(U,\cdot,\cdot)}(\mathbb{G}, g, U, V, h_1, \ldots, h_Q)$
Initialize $ctr \leftarrow 0$; $bad \leftarrow false$; empty tables $\overline{H_1}, \overline{H_2}$;
Run $\mathcal{A}$ on input $(\mathbb{G}, g, y = U)$ as the public parameters of the protocol and simulates the protocol's environment for $\mathcal{A}$ as follows:

Guess the test session by choosing at random the user (let us call him Bob) and the order number of the test session. If n is an upper bound to the number of all the sessions initiated by $\mathcal{A}$ then the guess is right with probability at least $1/n$.

H_2 **queries** On input a pair (z_1, z_2):
If $\overline{H_2}[z_1, z_2] = \perp$: choose a random string $Z \in \{0,1\}^\ell$ and store $\overline{H_2}[z_1, z_2] = Z$
Return $\overline{H_2}[z_1, z_2]$ to $\mathcal{A}$

H_1 **queries** On input (ID, r):
If $\overline{H_1}[ID, r] = \perp$, then $ctr \leftarrow ctr + 1; \overline{H_1}[ID, r] = h_{ctr}$
Return $\overline{H_1}[ID, r]$ to $\mathcal{A}$

Party Corruption. When $\mathcal{A}$ asks to corrupt party $ID \neq B$, then:
$ctr \leftarrow ctr + 1; s \xleftarrow{\$} \mathbb{Z}_q; r = g^s y^{-h_{ctr}}$
If $\overline{H_1}[ID, r] \neq \perp$ then $bad \leftarrow true$
Store $\overline{H_1}[ID, r] = h_{ctr}$ and return (r, s) as ID's private key.
For the case of Bob, the simulator simply chooses the r_B component of Bob's private key by picking a random $k_B \xleftarrow{\$} \mathbb{Z}_q$ and setting $r_B = g^{k_B}$. We observe that in this case $\mathcal{B}$ is not able to compute the corresponding s_B. However, since Bob is the user guessed for the test session, we can assume that the adversary will not ask for his secret key.

Simulating sessions. First we describe how to simulate sessions different from the test session. Here the main point is that the adversary is allowed to ask session-key queries and thus the simulator must be able to produce the correct session key for each of these sessions. The simulator has full information about all the users' secret keys except Bob. Therefore $\mathcal{B}$ can easily simulate all the protocol sessions that do not include Bob, and answer any of the attacker's queries about these sessions. Hence we concentrate on describing how $\mathcal{B}$ simulates interactions with Bob.

Assume that Bob has a session with Charlie (whose identity is the string C). If Charlie is an uncorrupted party this means that $\mathcal{B}$ will generate the messages on behalf of him. In this case $\mathcal{B}$ knows Charlie's secret key and also has chosen his ephemeral exponent t_C. Thus it is trivial to see that $\mathcal{B}$ has enough information to compute the correct session key. The case when the adversary presents a message $\langle C, r_C, u_C \rangle$ to Bob as coming from Charlie is more complicated. Here is where $\mathcal{B}$ makes use of the oracle $\mathsf{DH}(y, \cdot, \cdot)$ to answer a session-key query about this session. The simulator replies with a message $\langle B, r_B, u_B = g^{t_B} \rangle$ where t_B is chosen by $\mathcal{B}$. Recall that the session key is $H_2(z_1, z_2)$ with $z_1 = g^{(s_C + t_C)(s_B + t_B)}$ and $z_2 = u_C^{t_B}$. So z_1 is the Diffie-Hellman result of the values $u_C g^{s_C}$ and $u_B g^{s_B}$, where $g^{s_C} = r_C y^{H_1(C, r_C)}$ and $g^{s_B} = r_B y^{H_1(B, r_B)}$ can be computed by the simulator. Notice also that the simulator knows t_B and k_B (the discrete log of r_B in base g). Therefore it checks if $\overline{H_2}[z_1, z_2] = Z$ where $z_2 = u_C^{t_B}$ and $\mathsf{DH}(y, u_C g^{s_C}, \bar{z_1}) =$ "yes" where $\overline{z_1} = \frac{z_1}{(u_C g^{s_C})^{(k_B + t_B) H_1(B, r_B)^{-1}}}$. If $\mathcal{B}$ finds a match then it outputs the corresponding Z as session key for Bob. Otherwise it generates a random $\zeta \xleftarrow{\$} \{0,1\}^\ell$ and gives it as response to the adversary. Later, for each query (z_1, z_2) to H_2, if (z_1, z_2) satisfies the equation above it sets $\overline{H_2}[z_1, z_2] = \zeta$ and answers with ζ. This makes oracle's answers consistent.

In addition observe that the simulator can easily answer to state reveal queries as it chooses the fresh exponents on its own.

Simulating the test session. Let $\langle B, \rho_B, u_B = g^{t_B} \rangle$ be the message from Bob to Alice sent in the test session. We notice that such message may be sent by the adversary who is trying to impersonate Bob. In this case $\mathcal{A}$ may use a value $\rho_B = g^{\lambda_B}$ of its choice as the public component of Bob's private key (i.e. different than $r_B = g^{k_B}$ which $\mathcal{B}$ simulated and for which it knows k_B). $\mathcal{B}$ responds with the message $\langle A, r_A, u_A = V \rangle$ as coming from Alice. Finally $\mathcal{B}$ provides $\mathcal{A}$ with a random session key.

Run until $\mathcal{A}$ halts and outputs its decision bit

If $\overline{H_1}[B, \rho_B] = \perp$ then set $ctr \leftarrow ctr + 1$ and $\overline{H_1}[B, \rho_B] = h_{ctr}$

If $bad = true$ then return $(0, \perp)$

Let $i \in \{1, \ldots, Q\}$ such that $H_1(B, \rho_B) = h_i$

Let $Z = H_2(z_1, z_2)$ be the correct session key for the test session where $z_1 = (u_A r_A y^{H_1(A, r_A)})^{(t_B + \lambda_B + x h_i)}$ and $z_2 = u_A^{t_B}$.

If $\mathcal{A}$ has success into distinguishing Z from a random value it must necessarily query the correct pair (z_1, z_2) to the random oracle H_2. This means that $\mathcal{B}$ can efficiently find the pair (z_1, z_2) in the table $\overline{H_2}$ using the Strong-DH oracle.

Compute $\tau = \frac{z_1}{z_2 (u_B \rho_B y^{h_i})^{s_A}} = \rho_B^v W^{h_i}$

Return $(i, (\tau, h_i))$

Let IG be the algorithm that generates a random Diffie-Hellman tuple $(\mathbb{G}, g, U, V)$ and $acc_{\mathcal{B}}$ be the accepting probability of $\mathcal{B}$.[2] Then we have that:

$$acc_{\mathcal{B}} \geq \frac{\epsilon}{n} - \Pr[bad = true].$$

The probability that $bad = true$ is the probability that the adversary has guessed the "right" r for a corrupted party ID before corrupting it, in one of the H_1 oracle queries beforehand. Since r is uniformly distributed the probability of guessing it is $1/q$, and since the adversary makes at most Q queries to H_1 and corrupts at most Q_c parties (and $q > 2^\ell$) we have that

$$acc_{\mathcal{B}} \geq \frac{\epsilon}{n} - \frac{Q_c(Q)}{2^\ell}.$$

which is still non-negligible, since ϵ is non-negligible.

Once we have described the algorithm $\mathcal{B}$ we can now show how to build a solver algorithm S that can exploit $F_{\mathcal{B}}$, the forking algorithm associated with the above $\mathcal{B}$.

The algorithm S plays the role of a CDH solver under the Strong-DH Assumption. It receives in input a CDH tuple $(\mathbb{G}, g, U, V)$ where $U = g^u, V = g^v$ and u, v are random exponents in $\mathbb{Z}_q$. S is also given access to a decision oracle $\mathsf{DH}(U, \cdot, \cdot)$ that on input $(\hat{V}, \hat{W})$ answers "yes" if $(U, \hat{V}, \hat{W})$ is a valid DH tuple.

Algorithm $S^{\mathsf{DH}(U,\cdot,\cdot)}(\mathbb{G}, g, U, V)$

$(b, \tau, \tau') \stackrel{\$}{\leftarrow} F_{\mathcal{B}}^{\mathsf{DH}(U,\cdot,\cdot)}(\mathbb{G}, g, U, V)$
If $b = 0$ then return 0 and halt
Parse σ as (τ, h) and σ' as (τ', h')
Return $W = (\tau/\tau')^{(h-h')^{-1}}$

If the forking algorithm $F_{\mathcal{B}}$ has success, this means that there exist random coins ρ, an index $J \geq 1$ and $h_1, \ldots, h_Q, h'_J, \ldots, h'_Q \in \mathbb{Z}_q$ with $h = h_J \neq h'_J = h'$ such that: the first execution of $\mathcal{B}(\mathbb{G}, g, U, V, h_1, \ldots, h_Q; \rho)$ outputs $\tau = \rho_B^v W^h$ where $\overline{H_1}[B, \rho_B] = h$; the second execution of $\mathcal{B}(\mathbb{G}, g, U, V, h_1, \ldots, h_{J-1}, h'_J, \ldots, h'_Q; \rho)$ outputs $\tau' = (\rho'_{B'})^v W^{h'}$ where $\overline{H_1}[B', \rho'_{B'}] = h'$. Since the two executions of $\mathcal{B}$ are the same until the response to the J-th query to H_1, then we must have $B = B'$ and $\rho_B = \rho'_{B'}$. Thus it is easy to see that S achieves its goal by computing $W = (\tau/\tau')^{\frac{1}{h-h'}} = g^{uv}$.

Finally, by the General Forking Lemma, we have that if $\mathcal{A}$ has non-negligible advantage into breaking the security of IB-KA , then S's success probability is also non-negligible.

4.1 Additional Security Properties of IB-KA

In addition to the notion of session key security, any key-agreement protocol should satisfy other important properties. Below we describe the additional security properties enjoyed by IB-KA .

[2] We say that $\mathcal{B}$ accepts if it outputs (J, σ) such that $J \geq 1$.

Forward secrecy. We say that a KA protocol has forward secrecy, if after a session is completed and its session key erased, the adversary cannot learn it *even if* it corrupts the parties involved in that session. In other words, learning the private keys of parties should not jeopardize the security of past completed sessions.

A relaxed notion of forward secrecy (which we call *weak*) assumes that only past sessions in which the adversary was passive (i.e. did not choose the messages) are not jeopardized.

The following theorem (whose proof is deferred to the full version of the paper) shows that the protocol IB-KA satisfies this notion of *weak forward secrecy.*

Theorem 2. *Let $\mathcal{A}$ be a PPT adversary that is able to break the weak forward secrecy of the* IB-KA *protocol with advantage ϵ. Let n be the an upper bound to the number of sessions of the protocol run by $\mathcal{A}$ and Q_1 and Q_2 be the number of queries made by the adversary to the random oracles H_1, H_2 respectively. Then we can solve the CDH problem with probability at least $\epsilon/(nQ_2)$.*

Resistance to reflection attacks. A *reflection attack* occurs when an adversary can compromise a session in which the two parties have the same identity (and the same private key). Though, at first glance, this seems to be only of theoretical interest, there are real-life situations in which this scenario occurs. For example consider the case when Alice is at her office and wants to establish a secure connection with her PC at home, therefore running a session between two computers with the same identity and private key.

The current proof actually does not work when the adversary sends a message with the same value r_B provided by the KGC (for which the simulator knows the discrete logarithm k_B, but cannot compute the corresponding s_B). The issue is that the knowledge of s_B would be needed to extract the solution of the CDH problem. We point out that a reflection attack using a value $\rho_B \neq r_B$ is captured by the current proof. Moreover it is reasonable to assume that a honest party refuses connections from itself that use a "wrong" key.

However it is possible to adapt the proof in this specific case. In particular we can show that a successful run of the adversary enables the simulator to compute g^{u^2} instead of g^{uv}. As showed in [23] by Maurer and Wolf, such an algorithm can be easily turned into a solver for CDH. For lack of space this is deferred to the full version of the paper.

Resistance to Key Compromise Impersonation. Suppose that the adversary learns Alice's private key. Then, it is trivial to see that this knowledge enables the adversary to impersonate Alice to other parties. A *key compromise impersonation* (KCI) attack can be carried out when the knowledge of Alice's private key allows the adversary to impersonate another party to Alice.

To see that the protocol IB-KA is resistant to KCI attacks it suffices to observe that in the proof of security, when the adversary tries to impersonate Bob to Alice, we are able to output Alice's private key whenever it is asked by the adversary. It means that the proof continues to be valid even in this case.

Ephemeral Key Compromise Impersonation. A recent paper by Cheng and Ma [12] shows that our protocol is susceptible to an ephemeral key compromise attack. Roughly speaking this attack considers the case when the adversary can make state-reveal queries (in order to learn the ephemeral key of a user) even in the test session. Though the paper is correct, we point out that this kind of attack is not part of the standard Canetti-Krawczyk security model that is considered in this paper.

5 Comparisons with Other IB-KA Protocols

In this section we compare IB-KA with other id-based KA protocols from the literature. In particular, we consider the protocol by Chen and Kudla [11] (SCK-2) (which is a modification of Smart's [31]) and two protocols proposed very recently by Boyd *et al.* [6] (BCNP1, BCNP2).

For our efficiency comparisons we consider a security parameter of 128 and implementations of SCK-2, BCNP1 and BCNP2 with Type 3 pairings[3], which are the most efficient pairings for this kind of security level (higher than 80). Our protocol is assumed to be implemented in an elliptic curves group $\mathbb{G}$ with the same security parameter. In this scenario elements of $\mathbb{G}$ and $\mathbb{G}_1$ need 256 bit to be represented, while 512 bits are needed for $\mathbb{G}_2$ elements and 3072 bits for an element of $\mathbb{G}_T$. We estimate the computational cost of all the protocols using the costs per operation for Type 3 pairings given by Chen *et al.* in [10]. The bandwidth cost is expressed as the amount of data in bits sent by each party to complete a session of the protocol[4]. According to the work of Chen *et al.* [10] SCK-2 is the most efficient protocol with a proof of security in the CK model for all types of pairings. It is proved secure using random oracles under the Bilinear Diffie-Hellman Assumption and requires one round of communication with only one group element sent by each party. To be precise, we point out that the protocol of Boyd *et al.* (BMP) [7] would appear computationally more efficient than SCK-2, but unfortunately it works only in type 1 and type 4 pairings and is proven secure only in symmetric pairings. BCNP1 and BCNP2 are generic constructions based on any CCA-secure IB-KEM. When implemented (as suggested by the authors of [6]) using one of the IB-KEMs by Kiltz [19], Kiltz-Galindo [20] or Gentry [17] they lead to a two-pass single-round protocol with (CK) security in the standard model. BCNP2 provides weak FS and resistance to KCI attacks, while BCNP1 satisfies only the former property.

The results are summarized in Table 1 assuming protocols BCNP1 and BCNP2 to be implemented with Kiltz's IB-KEM (the most efficient for this application according to the work of Boyd *et al.* [6]). We defer to the original papers of SCK-2 [11] and BCNP1, BCNP2 [6] for more details about these costs. As described in the table, our protocol has a reasonable bandwidth requirement and achieves the best computational efficiency among the other id-based KA protocols.

[3] This classification of pairing groups into several types is provided by Galbraith *et al.* in [15].

[4] We do not consider the identity string sent with the messages as it can be implicit and, in any way, appears in all the protocols.

Table 1. Comparisons between IB-KA protocols

	weak FS	KCI	Standard model	Efficiency	
				Bandwidth	Cost per party
BCNP1	✗	✓	✓	768	56
BCNP2	✓	✓	✓	1024	59
SCK-2	✓	✓	✗	256	43
IB-KA	✓	✓	✗	512	6

Comparison with PKI-based protocols. We also compare our protocol to MQV [22], and its provably secure version HMQV [21], which is the most efficient protocol in the public-key setting. When comparing our protocol to a PKI-based scheme, like MQV, we must also consider the additional cost of sending and verifying certificates.

We measure the computation costs of the protocols in terms of the number of exponentiations in the underlying group needed to compute the session key. If the exponentiations is done with an exponent that is half the length of the group size, then obviously we count it as 1/2 exponentiation. Also if an exponentiation is done over a fixed basis, we apply precomputation schemes to speed up the computation, e.g. [16].

Our protocol requires each party to send a single message consisting of two group elements. To compute the session key, the parties perform 2 full exponentiations over variable basis, and one half exponentiation over a fixed basis[5]. For our security parameter, following [16], the latter half exponentiation can be computed with less than 20 group multiplications, with a precomputation table of moderate size.

In MQV, each party sends a single message consisiting of one group element, and performs 1.5 exponentiations to compute the session key. Moreover, in HMQV certificates are sent and verified. Here we distinguish two cases: the certificate is based either on an RSA signature, or on a discrete-log signature, e.g. Schnorr's.

In the RSA case, a short exponent e.g. $e = 2^{16} + 1$, is typically used, and the verification cost is basically equivalent to the cost of the half exponentiation with precomputation in our protocol above. Therefore in this case, MQV is faster, but by a mere half exponentiation. The price to pay however is a massive increase in bandwidth to send the RSA signature (i.e. 3072 bits), and the introduction of the RSA Assumption in order to prove security of the entire scheme. If we use a Schnorr signature for the certificate, then MQV require sending two more group elements, and therefore its bandwidth requirement is already worse than our protocol (by one group element). The parties then must compute one full and one half exponentiation, both with fixed basis[6] to verify the certificate. This extra computational cost can be compared to an additional half exponentiation, making the computation requirement of MQV with Schnorr certificates equivalent to that of our protocol.

[5] Indeed since the input to the hash function H_1 is randomized, we can set its output length to be half of the length of the group size.

[6] Though different basis, which means that in order to apply precomputation techniques, the parties need to maintain two tables.

In conclusion, when comparing our protocol with MQV with certificates we find that our protocol: (i) has comparable computational cost; (ii) has better bandiwdth (by far in the case of RSA certificates) and (iii) simplifies protocol implementation by removing entirely the need to manage certificates and to interact with a PKI[7].

References

1. Abdalla, M., Bellare, M., Rogaway, P.: The oracle Diffie-Hellman assumptions and an analysis of DHIES. In: Naccache, D. (ed.) CT-RSA 2001. LNCS, vol. 2020, pp. 143–158. Springer, Heidelberg (2001)
2. Bellare, M., Neven, G.: New Multi-Signature Schemes and a General Forking Lemma. In: Proceedings of the 13th Conference on Computer and Communications Security – ACM CCS 2006. ACM Press, New York (2006)
3. Boneh, D., Boyen, X.: Short Signatures without Random Oracles. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 56–73. Springer, Heidelberg (2004)
4. Boneh, D., Franklin, M.K.: Identity-Based Encryption from the Weil Pairing. SIAM J. Comput. 32(3), 586–615 (2003); In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–615. Springer, Heidelberg (2001)
5. Boyd, C., Choo, K.-K.R.: Security of two-party identity-based key agreement. In: Dawson, E., Vaudenay, S. (eds.) Mycrypt 2005. LNCS, vol. 3715, pp. 229–243. Springer, Heidelberg (2005)
6. Boyd, C., Cliff, Y., Nieto, J.G., Paterson, K.G.: Efficient One-Round Key Exchange in the Standard Model. In: Mu, Y., Susilo, W., Seberry, J. (eds.) ACISP 2008. LNCS, vol. 5107, pp. 69–83. Springer, Heidelberg (2008)
7. Boyd, C., Mao, W., Paterson, K.G.: Key Agreement Using Statically Keyed Authenticators. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 248–262. Springer, Heidelberg (2004)
8. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 453–474. Springer, Heidelberg (2001)
9. Canetti, R., Krawczyk, H.: Universally Composable Notions of Key Exchange and Secure Channels. In: Knudsen, L.R. (ed.) EUROCRYPT 2002. LNCS, vol. 2332, pp. 337–351. Springer, Heidelberg (2002)
10. Chen, L., Cheng, Z., Nigel, P.: Smart. Identity-based key agreement protocols from pairings. Int. J. Inf. Sec. 6(4), 213–241 (2007)
11. Chen, L., Kudla, C.: Identity Based Authenticated Key Agreement Protocols from Pairings. In: 16th IEEE Computer Security Foundations Workshop - CSFW 2003, pp. 219–233. IEEE Computer Society Press, Los Alamitos (2003)
12. Cheng, Q., Ma, C.: Ephemeral Key Compromise Attack on the IB-KA protocol. Cryptology Eprint Archive, Report 2009/568 (2009), `http://eprint.iacr.org/2009/568`
13. Diffie, W., Hellman, M.: New Directions in Cryptography. IEEE Transactions on Information Theory 22(6), 644–654 (1976)

[7] In the above, we did not account for the cost of verifying group membership for the elements sent by the parties, which is necessary both in the case of MQV and our protocol, and is the same in both protocols.

14. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions of Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Galbraith, S.D., Paterson, K.G., Smart, N.P.: Pairings for Cryptographers. Cryptology ePrint Archive, Report 2006/165 (2006), http://eprint.iacr.org
16. Lim, C.H., Lee, P.J.: More Flexible Exponentiation with Precomputation. In: Desmedt, Y.G. (ed.) CRYPTO 1994. LNCS, vol. 839, pp. 95–107. Springer, Heidelberg (1994)
17. Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445–464. Springer, Heidelberg (2006)
18. Gunther, C.G.: An Identity-Based Key-Exchange Protocol. In: Quisquater, J.-J., Vandewalle, J. (eds.) EUROCRYPT 1989. LNCS, vol. 434, pp. 29–37. Springer, Heidelberg (1990)
19. Kiltz, E.: Direct Chosen-Ciphertext Secure Identity-Based Encryption in the Standard Model with short Ciphertexts. Cryptology Eprint Archive, Report 2006/122 (2006), http://eprint.iacr.org/2006/122
20. Kiltz, E., Galindo, D.: Direct Chosen-Ciphertext Secure Identity-Based Key Encapsulation Without Random Oracles. Cryptology Eprint Archive, Report 2006/034 (2006), http://eprint.iacr.org/2006/034
21. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 546–566. Springer, Heidelberg (2005)
22. Law, L., Menezes, A., Qu, M., Solinas, J., Vanstone, S.: An efficient Protocol for Authenticated Key Agreement. Designs, Codes and Cryptography 28, 119–134 (2003)
23. Maurer, U., Wolf, S.: Diffie-Hellman oracles. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 268–282. Springer, Heidelberg (1996)
24. Okamoto, E.: Key Distribution Systems Based on Identification Information. In: Pomerance, C. (ed.) CRYPTO 1987. LNCS, vol. 293, pp. 194–202. Springer, Heidelberg (1988)
25. Okamoto, E., Tanaka, K.: Key Distribution System Based on Identification. Information. IEEE Journal on Selected Areas in Communications 7(4), 481–485 (1989)
26. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology 13(3), 361–396 (2000)
27. Saeednia, S.: Improvement of Gunther's identity-based key exchange protocol. Electonics Letters 31(18), 1535–1536 (2000)
28. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: Symposium on Cryptography and Information Security, Okinawa, Japan (2000)
29. Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
30. Schnorr, C.P.: Efficient identification and signatures for smart cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
31. Smart, N.P.: An identity-based authenticated key-agreement protocol based on the Weil pairing. Electronics letters 38, 630–632 (2002)
32. Smetters, D.K., Durfee, G.: Domain-based Administration of Identity-Based Cryptosystems for Secure E-Mail and IPSEC. In: SSYM 2003: Proceedings of the 12th Conference on USENIX Security Symposium, p. 15. USENIX Association (2003)
33. Wang, Y.: Efficient Identity-Based and Authenticated Key Agreement Protocol. Cryptology ePrint Archive, Report 2005/108 (2005), http://eprint.iacr.org/2005/108/