Strongly Secure Certificateless Key Agreement

Georg Lippold, Colin Boyd, Juan González Nieto

July 28, 2010

Abstract

We introduce a formal model for certificateless authenticated key exchange (CL-AKE) protocols. Contrary to what might be expected, we show that the natural combination of an ID-based AKE protocol with a public key based AKE protocol cannot provide strong security. We provide the first one-round CL-AKE scheme proven secure in the random oracle model. We introduce two variants of the Diffie-Hellman trapdoor introduced by [CKS08]. The proposed key agreement scheme is secure as long as each party has at least one uncompromised secret. Thus, our scheme is secure even if the key generation centre learns the ephemeral secrets of both parties.

1 Introduction

Certificateless encryption introduced by Al-Riyami and Paterson [ARP03] is a variant of identity based encryption that limits the key escrow capabilities of the key generation centre, which is inherent in identity based encryption [BF03]. Dent [Den08] published a survey of more than twenty certificateless encryption schemes that focuses on the different security models and the efficiency of the respective schemes. In certificateless cryptography schemes, there are three secrets per party:

- The key issued by the key generation centre (Dent [Den08] calls it "partial private key"). We assume in the following that this key is ID-based, although it does not necessarily have to be ID-based.
- The user generated private key x_{ID} (Dent calls it "secret value").
- The ephemeral value chosen randomly for each session.

Key agreement schemes provide an efficient means for two parties to communicate over an adversarial controlled channel. An overview of almost twenty identity based key agreement protocols has been compiled by Chen, Cheng and Smart [CCS07]; they also provide security proofs for two of the surveyed protocols. Many ID-based schemes guarantee full privacy for both parties as long as the key generation centre (KGC) does not learn any of the ephemeral secrets used in computing the session key. But as Krawczyk [Kra05] points out, the leakage of ephemeral keys should not be neglected as they are usually precomputed and not stored in secure memory. In the context of identity based key agreement protocols, this means that as soon as the ephemeral key of either party leaks, a malicious KGC is able to compute the session key.

An overview of current certificateless key agreement schemes has been compiled by Swanson [Swa08]. Certificateless key agreement schemes attempt to provide full privacy even if the ephemeral secrets of the parties leak to the key generation centre or if the key generation centre actively interferes with the messages that are exchanged (e.g. does a man-in-the-middle attack). The first certificateless key agreement scheme was published by Al-Riyami and Paterson [ARP03] as a side note to their certificateless encryption scheme. However, they provided neither a security model for certificateless key agreement schemes nor a proof of security for the scheme. Other certificateless key agreement schemes were published by Mandt and Tan [MT06] and improved by Xia et al. [XWSX08], Wang, Cao and Wang [WCW06], and Shao Zu-hua [Zh05], but the respective authors gave only heuristic arguments as to why their schemes would be secure. Swanson [Swa08] analysed these certificateless schemes and showed generic attacks that break the notions of security claimed by the respective authors. Swanson also posed three open questions in the last chapter of her thesis that we will answer in this paper.

By combining an ID-based scheme with a public key based scheme, certificateless encryption [YL04a], [LQ06], certificateless signatures [YL04b], and certificateless key encapsulation mechanisms [BFMLS08] can be readily constructed from existing protocols. Contrary to what would be expected, we show that a certificateless key agreement protocol cannot be securely constructed by a natural combination of an ID-based key agreement protocol with a public key based key agreement protocol.

The security model is an extension of Swanson's [Swa08] modified version of the extended Canetti and Krawczyk model presented in [LLM07] for certificateless key agreement. In this paper, we strengthen the model further (thus giving more power to the adversary) and provide the first formal proof for a strongly secure certificateless key agreement scheme in the random oracle model. Moreover, the protocol we propose is a one round protocol that withstands all of Swanson's attacks, although the messages exchanged in our protocol are exactly the same messages as in Mandt and Tan's protocol [MT06]. To withstand the attacks we use a modified version of the technique presented by Xia et al. [XWSX08].

We prove that our certificateless key agreement protocol is secure even if the key generation centre actively tries to break the scheme: it may either reveal ephemeral secrets or reveal secret values / replace public keys but not both. In fact, we show that as long as each party still has at least one uncompromised secret, our scheme is still secure in the random oracle model assuming that the computational Diffie-Hellman assumption and the computational bilinear Diffie-Hellman assumption hold. Our proofs are in the strongest security model available for certificateless schemes, i.e. it corresponds to Dent's [Den08] Strong Type I and Strong Type II security where the adversary is allowed to replace certificateless public keys and the challenger still has to answer all oracle queries.

The main contributions of this paper are:

- Strongest formal model for secure authenticated certificateless key exchange protocols today. We provide the equivalent of a strong decryption oracle [Den08] for reveal queries.
- An analysis of why certificateless key establishment schemes (CL-AKE) cannot be readily composed by combining an ID-based authenticated key establishment (ID-AKE) scheme with a public key authenticated key establishment (PK-AKE) scheme in our security model.
- First one-round protocol for certificateless key agreement with a security proof in the random oracle model that fulfills all notions of security of our model and withstands recent attacks on certificateless key agreement protocols.

The organization of the paper is as follows: we introduce the security model in Section 2 and relate it to existing notions of security for key agreement schemes and certificateless encryption. We also show why a generic composition of ID-AKE with PK-AKE does not have sufficient security guarantees in our model. A description of the scheme is given in Section 3 on page 6. Section 5 on page 10 discusses the security proof of the new protocol. We conclude our paper by answering some open questions in Section 6 on page 18.

2 Security model for certificateless key agreement schemes

The following security properties are commonly required of key establishment protocols in general.

Resistance to basic impersonation attacks. An adversary who does not know the private key of party A should not be able to impersonate A.

Resistance to Unknown Key-Share (UKS) attacks. An adversary $\mathcal{M}$ interferes with two honest parties A and B such that both parties accept the session and compute the same key. However, while A thinks that the key is shared with B, B is convinced that the key is shared with $\mathcal{M}$.

Known key security. Each run of a key agreement protocol between two parties A and B should produce a unique session key. A protocol should not become insecure if the adversary has learned some of the session keys [LMQ+03].

Weak Perfect Forward Secrecy (wPFS). A key-exchange protocol provides *weak PFS (wPFS)* if an attacker $\mathcal{M}$ cannot distinguish from random a key of any session for which the session and

its matching session are clean[1] even if $\mathcal{M}$ has learned the private keys of both peers to the session [Kra05, Definition 22].

Resistance to Key-Compromise Impersonation (KCI) attacks. We say that a KE-attacker $\mathcal{M}$ that has learned the private key of party $\hat{A}$ succeeds in a *Key-compromise impersonation (KCI)* attack against $\hat{A}$ if $\mathcal{M}$ is able to distinguish from random the session key of a complete session at $\hat{A}$ for which the session peer is uncorrupted and the session and its matching session (if it exists) are clean [Kra05, Definition 20].

Resistance to disclosure of ephemeral secrets. The protocol should be resistant to the disclosure of ephemeral secrets. The disclosure of an ephemeral secret should not compromise the security of sessions where the ephemeral secret was not used.

ID-based protocols usually require the following property in addition to these properties:

KGC forward secrecy The key generation centre (KGC) should be unable to compute the session key knowing all publicly available information.

For certificateless protocols, we will additionally require the following property. Mandt & Tan [MT06] call this property "Resistance to known session-specific temporary information", but they provide only an informal definition. It is not possible to provide this property in an ID-based key agreement scheme since a KGC who knows the ephemeral secrets has all inputs to the session key.

Resistance to leakage of ephemeral secrets to the KGC. If a malicious KGC learns the ephemeral secrets of any session, the KGC should not be able to compute the session key.

2.1 Formal definition of the security model

We present a strengthened version of Swanson's [Swa08] model, which in turn is based on LaMacchia, Lauter & Mityagin's [LLM07] extended Canetti-Krawczyk (eCK) model. We discuss the changes to the respective models in Section 2.2 on page 5.

Let $\mathcal{U} = \{U_1, \ldots U_n\}$ be a set of parties. The protocol may be run between any two of these parties. For each party there exists an identity based public key that can be derived from its identifier. There is a key generation centre that issues identity based private keys to the parties through a secure channel. Additionally, the parties generate their own secret values and certificateless public keys.

The adversary is in control of the network over which protocol messages are exchanged. $\Pi^t_{i,j}$ represents the t^{th} protocol session which runs at party i with intended partner party j. Additionally, the adversary is allowed to replace certificateless public keys that are used to compute the session key. The adversary does not have to disclose the private key matching the replaced certificateless public key to the respective party.

A session $\Pi^t_{i,j}$ enters an *accepted state* when it computes a session key $SK^t_{i,j}$. Note that a session may terminate without ever entering into an accepted state. The information of whether a session has terminated with acceptance or without acceptance is assumed to be public. The session $\Pi^t_{i,j}$ is assigned a partner ID $pid = (ID_i, ID_j)$. The session ID sid of $\Pi^t_{i,j}$ at party i is the transcript of the messages exchanged with party j during the session. Two sessions $\Pi^t_{i,j}$ and $\Pi^u_{j,i}$ are considered matching if they have the same pid (and sid).

The game runs in two phases. During the first phase of the game, the adversary $\mathcal{M}$ is allowed to issue the following queries in any order:

Send$(\Pi^t_{i,j}, x)$**:** If the session $\Pi^t_{i,j}$ does not exist, it will be created as initiator at party i if $x = \lambda$, or as a responder at party j otherwise. If the participating parties have not been initiated before, the respective private and public keys are created. Upon receiving the message x, the protocol is executed. After party i has sent and received the last set of messages specified by the protocol, it outputs a decision indicating accepting or rejecting the session. In the case of one-round protocols, party i behaves as follows:

$x = \lambda$**:** Party i generates an ephemeral value and responds with an outgoing message only.

[1] Roughly speaking *clean* is the same as *fresh* in Definition 1 on the following page.

$x \neq \lambda$**:** If party i is a responder, it generates an ephemeral value for the session and responds with an outgoing message m and a decision indicating acceptance or rejection of the session. If party i as an initiator, it responds with a decision indicating accepting or rejecting the session.

In this work, we require $i \neq j$, i.e. a party will not run a session with itself.

Reveal master key The adversary is given access to the master secret key.

Session key reveal($\Pi^t_{i,j}$): If the session has not accepted, it returns $\perp$, otherwise it reveals the accepted session key.

Reveal ID-based secret(i): Party i responds with its ID-based private key, e.g. $sH_1(ID_i)$.

Reveal secret value(i): Party i responds with its secret value x_i that corresponds to its certificateless public key. If i has been asked the *replace public key* query before, it responds with $\perp$.

Replace public key(i, pk): Party i's certificateless public key is replaced with pk chosen by the adversary. Party i will use the new public key for all communication and computation.

Reveal ephemeral key($\Pi^t_{i,j}$): Party i responds with the ephemeral secret used in session $\Pi^t_{i,j}$

We can group the key reveal queries into three types: the *reveal master key* and *reveal ID-based secret* queries try to undermine the security of the ID-based part of the scheme, the *reveal secret value* and *replace public key* queries try to undermine the security of the public key based part of the scheme, and the *reveal ephemeral key* query tries to undermine the security of one particular session.

We define the state *fully corrupt* as a session that was asked all three types of reveal queries: the *reveal master key* or *reveal ID-based secret*, the *reveal secret value* or the *replace public key*, and the *reveal ephemeral key* query.

Once the adversary $\mathcal{M}$ decides that the first phase is over, it starts the second phase by choosing a *fresh session* $\Pi^t_{i,j}$ and issuing a *Test($\Pi^t_{i,j}$)* query, where the *fresh session* and *test query* are defined as follows:

Definition 1 (Fresh session)**.** *A session $\Pi^t_{i,j}$ is fresh if (1) $\Pi^t_{i,j}$ has accepted; (2) $\Pi^t_{i,j}$ is unopened (not being issued the* session key reveal *query); (3) the session state at neither party participating in this session is* fully corrupted*; (4) there is no opened session $\Pi^u_{j,i}$ which has a matching conversation to $\Pi^t_{i,j}$.*

Test($\Pi^t_{i,j}$) The input session $\Pi^t_{i,j}$ must be fresh. A bit $b \in \{0, 1\}$ is randomly chosen. If $b = 0$, the adversary is given the session key, otherwise it randomly samples a session key from the distribution of valid session keys and returns it to the adversary.

After the *test($\Pi^t_{i,j}$)* query has been issued, the adversary can continue querying except that the test session $\Pi^t_{i,j}$ should remain *fresh*. We emphasize here that partial corruption is allowed as this is a benefit of our security model. Additionally, *replace public key* queries may be issued to *any* party after the test session has been completed.

At the end of the game, the adversary outputs a guess $\hat{b}$ for b. If $\hat{b} = b$, we say that the adversary wins. The adversary's advantage in winning the game is defined as

$$Adv^{\mathcal{M}}(k) = \left| \Pr[\mathcal{M} \text{ wins}] - \frac{1}{2} \right|$$

Definition 2 (Strong Type I secure key agreement scheme)**.** *A certificateless key agreement scheme is Strong Type I secure if every probabilistic, polynomial-time adversary $\mathcal{M}$ has negligible advantage in winning the game described in Section 2.1 on the preceding page subject to the following constraints:*

- *$\mathcal{M}$ may corrupt at most two out of three types of secrets per party involved in the* test *session,*
- *$\mathcal{M}$ is allowed to replace public keys of any party; however, this counts as the corruption of one secret,*
- *$\mathcal{M}$ may not reveal the secret value of any identity for which it has replaced the certificateless public key,*

- $\mathcal{M}$ *is allowed to ask* session key reveal *queries even for session keys computed by identities where* $\mathcal{M}$ *replaced the identity's public key.*
- $\mathcal{M}$ *is allowed to replace public keys of any party after the* test *query has been issued.*

Definition 3 (Strong Type II secure key agreement scheme)**.** *A certificateless key agreement scheme is Strong Type II secure if every probabilistic, polynomial-time adversary* $\mathcal{M}$ *has negligible advantage in winning the game described in Section 2.1 on page 3 subject to the following constraints:*

- $\mathcal{M}$ *is given the master secret key* s *at the start of the game,*
- $\mathcal{M}$ *may corrupt at most one additional type of secret per party participating in the test query,*
- $\mathcal{M}$ *is allowed to replace public keys of any party; however, this counts as the corruption of one secret,*
- $\mathcal{M}$ *may not reveal the secret value of any identity for which it has replaced the certificateless public key,*
- $\mathcal{M}$ *is allowed to ask session key reveal queries even for session keys computed by identities where he replaced the identity's public key.*
- $\mathcal{M}$ *is allowed to replace public keys of any party after the test query has been issued.*

2.2 Relation to existing notions of security

Swanson's [Swa08] *replace public key* query is weaker in assuming that the party whose key was replaced continues to make its computations with its original (unreplaced) public key (and its matching private key). Although it seems that Swanson's model is more "natural" than our model, strong certificateless encryption has been the goal of many papers, a discussion of the benefits and drawbacks can be found in [DLP08]. As it gives more power to the adversary, we think that schemes that are strongly secure are preferable to those in a weaker security model.

When checking for a matching conversation, Swanson omits the certificateless public keys from the conversation transcript. This weakens the adversary compared to our model, as the adversary would not be allowed to replace public keys and try to replay the conversation with the replaced keys of the test query after the *test* query has been issued.

With respect to LaMacchia et al. [LLM07], the main difference of our definition is that instead of having only four pieces of secret information, in certificateless protocols there are six: the ID-based secret keys, the user's secret value, and the ephemeral private keys of both parties. We require a certificateless AKE to be secure as long as each party still holds at least one uncompromised secret.

We note that as the challenger has to answer *session key reveal* queries even for keys where the respective certificateless public keys have been replaced, the adversary has access to the equivalent of a "Strong Decrypt" oracle in certificateless encryption. Strong decryption oracles were first introduced by Al-Riyami and Paterson [ARP03]. Dent [Den08] defines the Strong Decryption Oracle as follows.

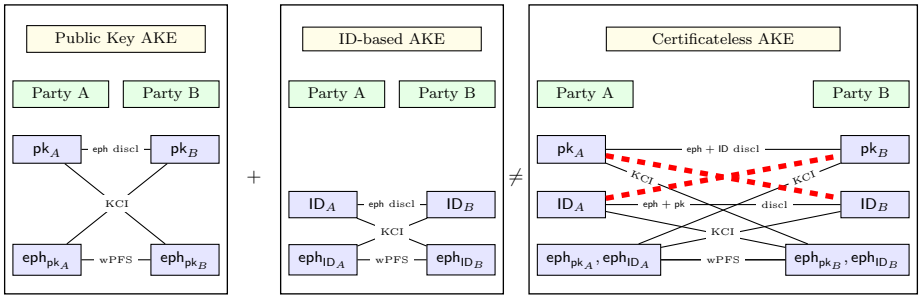
Definition 4 (Strong Decryption Oracle)**.** *The adversary supplies an identity* ID *and a ciphertext* C*, and the challenger responds with the decryption of* C *under the private key* sk_{ID}*. Note that if the attacker has replaced the public key for ID, then this oracle should return the correct decryption of* C *using the private key that inverts the public key* pk_{ID} *currently associated with the identity* ID *(or* $\perp$ *if no such private key exists).*

A strong decryption oracle in public key cryptography is able to return the plaintext for a given ciphertext (which does not necessarily mean that the plaintext has been decrypted using the correct key, as with double encryption). We note that in a *session key reveal* query the correct key for a given session has to be revealed, which is a stronger requirement. The scheme in Section 3 on the following page is both Strong Type I and Strong Type II secure with respect to Dent's definitions.

In the security proof in Section 5 on page 10 and Section 5.4 on page 14 we do not differentiate between these two types of adversarial behaviour but treat them together. If the adversary was split to be either Strong Type I or Strong Type II, then a Strong Type II adversary would be applicable only for the Strategies 1, 2, 3, and 4 in Section 5.1 on page 11. Being able to distinguish between Type I and Type II adversaries would thus increase the probability of success for the challenger.

2.3 Why a natural composition of CL-AKE from ID-AKE and PK-AKE is not possible in our model

In the security model, a session can only be fresh *as long as each party still has at least one uncompromised secret*. A composition of an ID-AKE with a PK-AKE is depicted in Figure 1. A natural way to achieve such a composition consists of running the two protocols in parallel and deriving the session key of the overall composition as a publicly known function of solely the two component session keys. This composition cannot offer the desired level of security, because no security guarantees exist if party A still has an uncompromised key in the PK-AKE and party B still has an uncompromised key in the ID-AKE (both AKE schemes are broken at this moment). This may explain why no CL-AKE schemes with a proof of security have been published before.



The lines indicate what combination of secrets gives resistance against which attack type. Examples for public key schemes applicable to this diagram would be NAXOS [LLM07] and CMQV [Ust08], an example for an ID-based scheme would be the ASIACCS09 [HC08] scheme. However, a combination of these schemes would not have any security guarantees about the dashed lines in the certificateless part of the diagram.

Figure 1: PK-AKE + ID-AKE ≠ CL-AKE

3 Description of the certificateless key agreement scheme

We describe the phases of our certificateless authenticated key exchange protocol in this section. Our protocol consists of three phases: setup, message exchange and key computation. We also briefly address the efficiency of the proposed protocol.

3.1 Setup

- The KGC publishes a generator $P \in \mathbb{G}$ and an admissible bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ that fulfills the following criteria:

  Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups of prime order p. A bilinear pairings map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ between the groups $\mathbb{G}$ and $\mathbb{G}_T$ satisfies the following properties:

  Bilinear We say that a map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is *bilinear* if $e(aP, bP) = e(P, P)^{ab}$ for all $P \in \mathbb{G}$ and $a, b \in \mathbb{Z}_p$.

  Non-degenerate We say that e is non-degenerate if it does not send all pairs in $\mathbb{G} \times \mathbb{G}$ to the identity in $\mathbb{G}_T$. Since $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime order p, it follows that if $P \in \mathbb{G}$ is a generator of $\mathbb{G}$, then $e(P, P)$ is a generator of $\mathbb{G}_T$.

  Computable There is an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in \mathbb{G}$.

  Suitable pairing groups for this protocol would be Type 1 and Type 4 pairings (see Chen, Cheng & Smart [CCS07] for a discussion). Asymmetric pairings are not possible because we use the non-interactive ID-based key agreement of Sakai, Ohgishi and Kasahara (SOK) [SOK00] as part of our protocol. This requires hashing to both $\mathbb{G}_1$ and $\mathbb{G}_2$. The SOK protocol has been proven by Dupont

and Enge [DE02] using gap assumptions. As an added benefit of our proof, we show how to prove the SOK protocol secure under the weaker computational bilinear Diffie-Hellman assumption using the twin bilinear Diffie-Hellman trapdoor [CKS08] in section 5.4 on page 14, Strategy 9.

- The KGC picks a random $s \in \mathbb{Z}_p$ as master secret key and sets its public key to sP
- The KGC selects three cryptographic hash functions

$$\begin{aligned} H_1 &: \quad \{0,1\}^* \to \mathbb{G} \\ H_2 &: \quad \{0,1\}^* \times \{0,1\}^* \times \mathbb{G}^8 \times {\mathbb{G}_T}^6 \to \{0,1\}^n \text{ for some integer } n > 0 \\ H_3 &: \quad \mathbb{G} \to \mathbb{G} \end{aligned}$$

  H_2 is the key derivation function for our scheme.

Each party participating in the key agreement protocol additionally computes a private key and a matching certificateless public key:

- Each user U generates a secret value $x_U \xleftarrow{\$} \mathbb{Z}_p$ and a public key $x_U P \in \mathbb{G}$
- Each user U gets an ID-based private key $\{sH_1(ID_U), sH_3(H_1(ID_U))\} \in \mathbb{G}^2$ from the key generation centre.

3.2 Message exchange

To establish a common key, user A generates the ephemeral secret $r_A \xleftarrow{\$} \mathbb{Z}_p$ and user B generates the ephemeral secret $r_B \xleftarrow{\$} \mathbb{Z}_p$. They exchange the following messages:

$$A \to B : E_A = (r_A P, x_A P) \qquad B \to A : E_B = (r_B P, x_B P)$$

We note that the certificateless public keys can be stripped from the messages if they are published in a public online directory. This will save bandwidth, but at the same time may make the scheme more vulnerable to the equivalent of denial of decryption attacks in certificateless encryption: an adversary may manipulate the entries of the directory more easily than the message exchange between two parties.

As we propose a one-round protocol, our protocol achieves only implicit authentication. Krawczyk [Kra05, Section 8] shows that explicit authentication is possible with three half rounds. To achieve explicit authentication, this protocol can be patched in the same way that HMQV is patched to HMQV-C.

In the following we require implicitly that each party always checks subgroup membership for all elements of messages that are exchanged in the protocol to defend against small subgroup attacks [LL97].

3.3 Key computation

To compute the certificateless session key, each user computes

$$\begin{aligned}
K_A &= e(H_1(ID_B), sP)^{r_A} e(sH_1(ID_A), r_B P) \\
&= e(H_1(ID_B), P)^{r_A s} e(H_1(ID_A), P)^{r_B s} \\
&= e(H_1(ID_A), sP)^{r_B} e(sH_1(ID_B), r_A P) \\
&= K_B = K \\
K'_A &= e(H_3(H_1(ID_B)), sP)^{r_A} e(sH_3(H_1(ID_A)), r_B P) \\
&= e(H_3(H_1(ID_B)), P)^{r_A s} \cdot e(H_3(H_1(ID_A)), P)^{r_B s} \\
&= e(sH_3(H_1(ID_B)), r_A P) \cdot e(H_3(H_1(ID_A)), sP)^{r_B} \\
&= K'_B = K' \\
L_A &= e(H_1(ID_B), sP)^{x_A} e(sH_1(ID_A), x_B P) \\
&= e(H_1(ID_B), P)^{x_A s} e(H_1(ID_A), P)^{x_B s} \\
&= e(sH_1(ID_B), x_A P)^{s} e(H_1(ID_A), sP)^{x_B} \\
&= L_B = L \\
L'_A &= e(H_3(H_1(ID_B)), sP)^{x_A} e(sH_3(H_1(ID_A)), x_B P) \\
&= e(H_3(H_1(ID_B)), P)^{x_A s} e(H_3(H_1(ID_A)), P)^{x_B s} \\
&= e(sH_3(ID_B), x_A P) e(H_3(ID_A), sP)^{x_B} \\
&= L'_B = L' \\
N_A &= e(H_1(ID_B), sH_1(ID_A)) = e(H_1(ID_B), H_1(ID_A))^s = N_B = N \\
N'_A &= e(H_3(H_1(ID_B)), sH_3(H_1(ID_A))) \\
&= e(H_3(H_1(ID_B)), H_3(H_1(ID_A)))^s \\
&= e(sH_3(H_1(ID_B)), H_3(H_1(ID_A))) \\
&= N'_B = N'
\end{aligned}$$

The session key is then computed as $SK = H_2(A, B, E_A, E_B, r_A r_B P, x_A x_B P, r_A x_B P, x_A r_B P, K, K', L, L', N, N')$. In Section 5 on page 10 and Section 5.4 on page 14 the challenger $\mathcal{B}$ uses the adversary $\mathcal{M}$ to solve either the computational Diffie-Hellman (CDH) or the computational bilinear Diffie-Hellman (CBDH) problem. K, L, and N are used in the proof to embed the input to the CBDH challenge into the test session. Each of these values is necessary to defend against one possible attack strategy of the adversary $\mathcal{M}$. K is the product of two encapsulated Boneh-Franklin session keys, L' is similar but with certificateless long-term keys. N' is the non-interactive ID-based key agreement scheme proposed by [SOK00]. K', L', and N' are needed to answer reveal queries of the adversary $\mathcal{M}$ consistently. To answer reveal queries, the challenger $\mathcal{B}$ makes use of the twin bilinear Diffie-Hellman problem as introduced by Cash, Kiltz and Shoup [CKS08]. The twin bilinear Diffie-Hellman "backdoor" is embedded in K', L' and N'.

3.4 Efficiency considerations

Although the protocol is one round, the computational overhead imposed on the parties is rather high: each party has to compute 5 exponentiations in $\mathbb{G}$ and 10 pairings. We would like to note that we need the H_3 hash function in the proof for full computational bilinear Diffie-Hellman security. If the gap bilinear Diffie-Hellman assumption is used (see Kudla and Paterson [KP05] for gap assumptions), the H_3 hash function can be omitted which saves 2 hash queries and reduces the complexity of the protocol to 3 exponentiations in $\mathbb{G}$ and 5 pairing computations (as K', L', and N' do not have to be computed). If there are multiple runs of the protocol between the same users (e.g. for rekeying in VPN's), then the complexity can be reduced by caching $x_A x_B P$, L, L', N, and N' in secure memory which then reduces the complexity for successive runs to 4 exponentiations and 4 pairing computations (or 2 exponentiations and 2 pairing computations if the gap bilinear Diffie-Hellman assumption is used). It may be possible to do better in terms of computational efficiency. However, the aim of this paper is to provide a strong model for certificateless key agreement and to show that schemes corresponding to the model exist.

We introduce the theorems that we later use as decisional oracles to be able to answer the H_2 queries of the adversary consistently (and to determine when the adversary submits the solution to a hard problem to the H_2 oracle). We continue then by embedding a hard problem in each of the uncorrupted secrets that are available in the respective strategies.

4 The Twin Bilinear Diffie-Hellman Trapdoor Theorems

The proof in section 5.4 on page 14 for Strategy 5 to 8 relies heavily on the following theorem:

Theorem 1 (Trapdoor Test)**.** *Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing, where $\mathbb{G}, \mathbb{G}_T$ are two cyclic groups of prime order p. Let $P \in \mathbb{G}$ be a generator of $\mathbb{G}$. Suppose $B_1 \in \mathbb{G}, y, z \in \mathbb{Z}_p$ are mutually independent random variables. Define $B_2 := yP - zB_1$. Further, suppose that A, C are random variables in $\mathbb{G}$ and T_1, T_2 are random variables in $\mathbb{G}_T$, each of which is defined as some function of B_1 and B_2. Then we have:*

1. *B_2 is uniformly distributed over $\mathbb{G}$.*
2. *B_1 and B_2 are independent.*
3. *If $B_1 = b_1 P$ and $B_2 = b_2 P$, then the probability that the truth value of*

$$T_1^z \cdot T_2 \stackrel{?}{=} e(A, C)^y \tag{1}$$

does not agree with the truth value of

$$T_1 \stackrel{?}{=} e(A, C)^{b_1} \wedge T_2 \stackrel{?}{=} e(A, C)^{b_2} \tag{2}$$

is at most $1/p$, moreover, if Equation 2 holds, then Equation 1 certainly holds.

See [CKS08], [HC08] for an explanation and a proof.

Additionally we need the "Additive double BDH Trapdoor Test" and the "Multiplicative double BDH Trapdoor Test" for Strategy 9:

Theorem 2 (Additive double BDH Trapdoor Test)**.** *Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing, where $\mathbb{G}, \mathbb{G}_T$ are two cyclic groups of prime order p. Let $P \in \mathbb{G}$ be a generator of $\mathbb{G}$. Suppose $B_1, D_1 \in \mathbb{G}, y_1, y_2, z \in \mathbb{Z}_p$ are mutually independent random variables. Define $B_2 := y_1 P - zB_1$ and $D_2 := y_2 P - zD_1$. Further, suppose that A, C are random variables in $\mathbb{G}$ and T_1, T_2 are random variables in $\mathbb{G}_T$, each of which is defined as some function of (A, C, B_1, D_1) and (A, C, B_2, D_2). Then we have:*

(i) *B_2 and D_2 are uniformly distributed over $\mathbb{G}$ (guaranteed by y_1 and y_2), as is $B_2 + D_2$.*

(ii) *B_1 and B_2 are independent and D_1 and D_2 are independent and B_2 and D_2 are independent, and $B_1 + D_1$ and $B_2 + D_2$ are independent (also due to y_1 and y_2).*

(iii) *If $B_1 = b_1 P, B_2 = b_2 P, D_1 = d_1 P, D_2 = d_2 P$, then the probability that the truth value of*

$$T_1^z T_2 \stackrel{?}{=} e(A, C)^{y_1 + y_2} \tag{3}$$

does not agree with the truth value of

$$T_1 \stackrel{?}{=} e(A, C)^{b_1} e(A, C)^{d_1} \wedge T_2 \stackrel{?}{=} e(A, C)^{b_2} e(A, C)^{d_2} \tag{4}$$

is at most $1/p$, moreover, if Equation 4 holds, then Equation 3 certainly holds.

Proof. This proof is a rewrite of Cash, Kiltz and Shoup's [CKS08] trapdoor test proof. Observe that $y_1 + y_2 = z(b_1 + d_1) + (b_2 + d_2)$. It is easy to verify that $B_2 + D_2$ is uniformly distributed over $\mathbb{G}$, and that $B_1 + D_1$, $B_2 + D_2$, z are mutually independent, from which *(i)* and *(ii)* follow. To prove *(iii)*, condition on fixed values of $B_1 + D_1$ and $B_2 + D_2$. In the resulting conditional probability space, z is uniformly distributed over $\mathbb{Z}_p$, while $(b_1 + d_1), (b_2 + d_2), e(A, C), T_1$ and T_2 are fixed. If Equation 4 holds, then by multiplying together the two equations in Equation 4, we see that Equation 3 certainly holds.

Conversely, if Equation 4 on the preceding page does not hold, we show that Equation 3 on the previous page holds with probability at most $1/p$. Observe that Equation 3 on the preceding page is equivalent to

$$\left(\frac{T_1}{e(A,C)^{b_1+d_1}}\right)^z = \frac{e(A,C)^{b_2+d_2}}{T_2}. \tag{5}$$

It is not hard to see that if $T_1 = e(A,C)^{b_1+d_1}$ and $T_2 \neq e(A,C)^{b_2+d_2}$, then Equation 5 certainly does not hold. This leaves us with the case $T_1 \neq e(A,C)^{b_1+d_1}$. But in this case, the left hand side of Equation 5 is a random element of $\mathbb{G}_T$ (since z is uniformly distributed in $\mathbb{Z}_p$), but the right hand side is a fixed element of $\mathbb{G}_T$. Thus, Equation 5 holds with probability $1/p$ in this case. □

Theorem 3 (Multiplicative double BDH Trapdoor Test)**.** [2] *Let $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing, where $\mathbb{G}, \mathbb{G}_T$ are two cyclic groups of prime order p. Let $P \in \mathbb{G}$ be a generator of $\mathbb{G}$. Suppose $B_1, C_1 \in \mathbb{G}, y_1, y_2, z \in \mathbb{Z}_p$ are mutually independent random variables. Define $B_2 := y_1P - zB_1$ and $C_2 := y_2P - zC_1$. Further, suppose that A is a random variables in $\mathbb{G}$ and T_1, T_2 are random variables in $\mathbb{G}_T$, each of which is defined as some function of (A, B_1, C_1) and (A, B_2, C_2). Then we have:*

(i) *B_2 and C_2 are uniformly distributed over $\mathbb{G}$ (guaranteed by y_1 and y_2), and $e(B_2, C_2)$ is uniformly distributed over $\mathbb{G}_T$.*

(ii) *B_1 and B_2 are independent and C_1 and C_2 are independent and B_2 and C_2 are independent, and $e(B_1, C_1)$ and $e(B_2, C_2)$ are independent (also due to y_1 and y_2).*

(iii) *If $B_1 = b_1P, B_2 = b_2P, C_1 = c_1P, C_2 = c_2P$, then the probability that the truth value of*

$$\frac{{T_1}^{z^2}}{T_2} \stackrel{?}{=} \frac{e(A,P)^{y_1y_2}}{e(A,C_2)^{y_1}e(A,B_1)^{y_2}} \tag{6}$$

does not agree with the truth value of

$$T_1 \stackrel{?}{=} e(A,P)^{b_1c_1} \wedge T_2 \stackrel{?}{=} e(A,P)^{b_2c_2} \tag{7}$$

is at most $2/p$, moreover, if Equation 7 holds, then Equation 6 certainly holds.

Proof. Observe that $y_1y_2 = (zb_1 + b_2)(zc_1 + c_2) = z^2b_1c_1 + zb_1c_2 + zb_2c_1 + b_2c_2$. It is easy to verify that $e(B_2, C_2)$ is uniformly distributed over $\mathbb{G}_T$, and that $e(B_1, C_1)$, $e(B_2, C_2)$, z are mutually independent, from which *(i)* and *(ii)* follow. To prove *(iii)*, condition on fixed values of $e(B_1, C_1)$ and $e(B_2, C_2)$. In the resulting conditional probability space, z is uniformly distributed over $\mathbb{Z}_p$, while b_1c_1, b_2c_2, A, T_1 and T_2 are fixed. If Equation 7 holds, then by multiplying together the two equations in Equation 7, we see that Equation 6 certainly holds. Conversely, if Equation 7 does not hold, we show that Equation 6 holds with probability at most $2/p$. Observe that Equation 6 is equivalent to

$$\left(\frac{T_1}{e(A,P)^{b_1c_1}}\right)^{z^2} = \frac{e(A,P)^{b_2c_2}}{T_2}. \tag{8}$$

It is not hard to see that if $T_1 = e(A,P)^{b_1c_1}$ and $T_2 \neq e(A,P)^{b_2c_2}$, then Equation 8 certainly does not hold. This leaves us with the case $T_1 \neq e(A,P)^{b_1c_1}$. But in this case, the left hand side of Equation 8 is the square of a random element of $\mathbb{G}_T$. Since z is uniformly distributed in $\mathbb{Z}_p$, z^2 is uniformly distributed over half of $\mathbb{Z}_p$ as half of the elements of $\mathbb{Z}_p$ are quadratic residues. On the other hand, the right hand side of 8 is a fixed element of $\mathbb{G}_T$. Thus, Equation 8 holds with probability $2/p$ in this case. □

5 Security proof for the certificateless key agreement scheme

We will prove that the certificateless key agreement scheme is a secure key agreement scheme in the random oracle model under the computational bilinear Diffie-Hellman (CBDH) assumption and the computational Diffie-Hellman (CDH) assumption. The CBDH the assumption states that given $\{aP, bP, cP\} \in \mathbb{G}^3$

[2] If this test was implemented with $B_2 = y_1P - z_1bP$ and $C_2 = y_2P - z_2cP$, then the probability that Equation 7 holds would be $\frac{1}{p^2}$. We use z instead of z_1 and z_2 because we need Theorem 2 on the previous page simultaneously.

it is hard to compute $e(P,P)^{abc} \in \mathbb{G}_T$. Let Z be an algorithm that takes as input a triple $\{aP, bP, cP\} \in \mathbb{G}^3$, and outputs an element $Z \in \mathbb{G}_T$. We define the CBDH advantage of Z to be

$$\Pr\left[a, b, c \xleftarrow{\$} \mathbb{Z}_p : Z(aP, bP, cP) = e(P,P)^{abc}\right]$$

The CDH assumption states that given $\{aP, bP\} \in \mathbb{G}^2$ it is hard to compute $abP \in \mathbb{G}$. Let Z be an algorithm that takes as input the pair $\{aP, bP\} \in \mathbb{G}^2$, and outputs an element $T \in \mathbb{G}$. We define the CDH advantage of Z to be

$$\Pr\left[a, b \xleftarrow{\$} \mathbb{Z}_p : Z(aP, bP) = abP\right]$$

To relate the advantage of an adversary against our protocol to the above assumptions, we use a classical reduction approach. We assume that an adversary $\mathcal{M}$ has an advantage in winning the game outlined in Section 2.1 on page 3. Additionally, the adversary $\mathcal{M}$ may query the random oracles H_1, H_2, and H_3. In the following, the challenger $\mathcal{B}$ is interested to use the adversary $\mathcal{M}$ to turn $\mathcal{M}$'s advantage in distinguishing a random session key from the correct session key in an advantage to solve either the computational Diffie-Hellman problem or the computational bilinear Diffie-Hellman problem. Let q_0 be the maximum number of sessions that any one party may have. We assume that the adversary $\mathcal{M}$ makes at most q_1 distinctive H_1 queries. The adversary may make any number of H_2 queries or H_3 queries. At the end of the game, $\mathcal{M}$ outputs its guess $\hat{b} \in \{0,1\}$ for b. Let $Adv^{\mathcal{M}}(k)[\Pi]$ be the advantage that the adversary $\mathcal{M}$ has against the protocol, i.e. the event that $\hat{b} = b$ and $\mathcal{M}$ wins the game.

Theorem 4. *If there exists an adversary that has an advantage against our certificateless key agreement scheme ($Adv^{\mathcal{M}}(k)[\Pi]$), the challenger $\mathcal{B}$ can use this adversary to solve either the computational Diffie-Hellman or the computational bilinear Diffie-Hellman problem. We show that the success probability of any adversary against the scheme is limited by*

$$Adv^{\mathcal{M}}(k)[\Pi] \leq 9 q_0 q_1^2 \max\left(Adv^{\mathcal{B}}(k)[CDH], Adv^{\mathcal{B}}(k)[CBDH]\right)$$

where $Adv^{\mathcal{B}}(k)[CDH]$ is the advantage that the challenger gets in solving the computational Diffie-Hellman problem given security parameter k using the adversary and $Adv^{\mathcal{B}}(k)[CBDH]$ is the advantage that the challenger gets in solving the computational bilinear Diffie-Hellman problem given security parameter k using the adversary.

We note that the CBDH problem is strictly weaker than the CDH problem. Thus, an adversary that is able to solve the CDH problem will also be able to solve the CBDH problem. We differentiate between these two problems because security against a Type II adversary is based solely on the CDH problem, whereas security against a Type I adversary is based on both the CDH problem and the CBDH problem.

5.1 Possible strategies for the challenger

Before the game starts, the challenger $\mathcal{B}$ tries to guess the test session. To this end, $\mathcal{B}$ randomly selects two indexes $I, J \in \{1, \ldots, q_1\} : I \neq J$ that represent the I^{th} and the J^{th} distinct query to the H_1 oracle. The probability that $\mathcal{B}$ chooses I and J correctly is (as there are at most q_1 entries in H_1)

$$\frac{1}{q_1(q_1-1)} > \frac{1}{q_1^2}$$

$\mathcal{B}$ chooses $T \in \{1, \ldots, q_0\}$ and thus determines the test oracle $\Pi^T_{I,J}$, which is correct with probability larger than $\frac{1}{q_0 q_1^2}$. If $\mathcal{B}$ did not guess the test session correctly, $\mathcal{B}$ aborts the game.

In order to use the adversary $\mathcal{M}$ to gain an advantage in computing the CBDH or the CDH challenge, the challenger $\mathcal{B}$ will guess the parts of the key in the session corresponding to the test query that the adversary may not learn. Depending on the chosen strategy, $\mathcal{B}$ aborts the game whenever $\mathcal{M}$'s queries target one of the forbidden elements. Otherwise, the game proceeds as usual. There are nine choices for $\mathcal{B}$ (see also Table 1 on the next page):

1. The adversary may learn neither the secret value of ID_I nor of ID_J.
2. The adversary may learn neither the ephemeral private key of ID_I nor of ID_J.

Strategy	1		2		3/4(mirr.)		5/6(mirr.)		7/8(mirr.)		9	
Value at party p	I	J	I	J	I	J	I	J	I	J	I	J
$sH_1(ID_p)$	c	c	c	c	c	c		c		c		
$sH_3(H_1(ID_p))$	c	c	c	c	c	c		c		c		
x_p / x_pP			c/r	c/r		c/r	c/r		c/r	c/r	c/r	c/r
r_p	c	c			c		c	c	c		c	c
Embedding in	x_Ix_JP		r_Ir_JP		r_Ix_JP/r_Jx_IP		K		L		N	
Problem type	CDH		CDH		CDH		CBDH		CBDH		CBDH	

c = corrupt, r = replace, mirr. = swap columns I and J

Strategy 1 - 4 are related to the computational Diffie-Hellman problem, Strategies 5 - 9 are related to the computational bilinear Diffie-Hellman problem. In the proof, the problem is always embedded in the values that the adversary may not corrupt or replace.

Table 1: Possible corrupt queries sorted by strategy

ID	$H_1(ID)$	$l \xleftarrow{\$} \mathbb{Z}_p$
ID_1	l_1P	l_1
...	...	...
ID_I	bP	$\perp$
...	...	...
ID_J	cP	$\perp$
...	...	...

Instead of choosing $H_1(ID_i)$ at random from $\mathbb{G}$, $\mathcal{B}$ chooses $l_i \in \mathbb{Z}_p$ at random, records it, and sets $H_1(ID_i)$ to l_iP. For Strategy 5, 7 and 9, the I^{th} entry is set to $H_1(ID_I) = bP$; for Strategy 6 and 8, the J^{th} entry is set to $H_1(ID_J) = bP$. For Strategy 9 the J^{th} entry is set to $H_1(ID_J) = cP$. bP and cP are taken from the inputs to the BDH challenge. As bP and cP are random in $\mathbb{G}$, this modification is indistinguishable for any adversary. The table above shows the H_1 oracle for Strategy 9 as an example.

Table 2: Modified H_1 oracle

3. The adversary may learn neither the secret value of ID_J nor replace the public key of ID_J and may also not learn the ID-based private key of ID_I.
4. The adversary may learn neither the secret value of ID_I nor replace the secret value of ID_I and may also not learn the ID-based private key of ID_J.
5. The adversary may learn neither the ephemeral private key of ID_J nor the secret value of ID_I.
6. The adversary may learn neither the ephemeral private key of ID_I nor the secret value of ID_J.
7. The adversary may learn neither the ephemeral private key of ID_J nor the ID-based private key of ID_I.
8. The adversary may learn neither the ephemeral private key of ID_I nor the ID-based private key of ID_J.
9. The adversary may learn neither the ID-based private key of ID_I nor of ID_J.

As there are nine strategies, the probability that $\mathcal{B}$ does not abort the game after $\mathcal{B}$ selected the strategy and the test session beforehand is now larger than $\frac{1}{9q_0q_1^2}$. The adversary may learn the key generation centre's master secret only in Strategy 1,2,3, and 4. Furthermore, $\mathcal{B}$ replaces the H_2 oracle by a table which records input/output pairs. If a query is made that matches one of the previous inputs, the corresponding output is returned, otherwise, a value from the respective output domain is chosen at

$g_i \in \mathbb{G}$	$H_3(g_i)$	$y_i \xleftarrow{\$} \mathbb{Z}_p$	$z \xleftarrow{\$} \mathbb{Z}_p$
$H_1(ID_I) = bP$	$y_{tbdh_1}P - zbP$	y_{tbdh_1}	z
$H_1(ID_J) = cP$	$y_{tbdh_2}P - zcP$	y_{tbdh_2}	z
g_1	$y_1 P$	y_1	$\perp$
...	...	...	$\perp$

Instead of choosing $H_3(g_i)$ for $g_i \in \mathbb{G}$ at random from $\mathbb{G}$, $\mathcal{B}$ chooses $y_i \in \mathbb{Z}_p$ at random, records it, and sets $H_3(g_i)$ to $y_i P$. For Strategy 5, 6, 7, 8 and 9, the oracle is patched before the game starts by setting $H_3(bP) = y_{tbdh_1}P - zbP$. For Strategy 9, the oracle is additionally patched before the game starts with $H_3(cP) = y_{tbdh_2}P - zcP$. bP and cP are taken from the inputs to the BDH challenge. As the pre-patched values are completely re-randomized, this modification is indistinguishable for any adversary. The table above shows the H_3 oracle for Strategy 9 as an example.

Table 3: Modified H_3 oracle suitable for twin bilinear Diffie-Hellman

random, the new input/output pair is added to the list and the value is returned. The H_1 and H_3 oracle operate as explained in Table 2 on the preceding page and Table 3 respectively.

5.2 Behaviour of the challenger based on the chosen strategy

To solve the computational DH problem using $\mathcal{M}$, $\mathcal{B}$ is given the values (aP, bP) and $\mathcal{B}$'s task is to compute abP. To solve this problem, $\mathcal{B}$ uses the H_2 oracle. The bilinear pairing is used for consistency checks.

To solve the computational BDH problem using $\mathcal{M}$, $\mathcal{B}$ is given the values (aP, bP, cP) and $\mathcal{B}$'s task is to compute $e(P,P)^{abc}$. To solve this problem, $\mathcal{B}$ uses the H_2 and the H_1 oracle. The H_3 oracle is used for consistency checks and operates as in Table 3.

The session key SK is generated by querying H_2 on (ID_i, ID_j, r_iP, x_iP, r_jP, x_jP, r_ir_jP, x_ix_jP, r_ix_jP, r_jx_iP, K, K', L, L', N, N') where

$$K = \underbrace{e(H_1(ID_j), P)^{r_i s}}_{K_1} \cdot \underbrace{e(H_1(ID_i), P)^{r_j s}}_{K_2},$$

$$L = \underbrace{e(H_1(ID_j), P)^{s x_i}}_{L_1} \cdot \underbrace{e(H_1(ID_i), P)^{s x_j}}_{L_2}, \qquad N = e(H_1(ID_i), H_1(ID_j))^s$$

Depending on the chosen strategy, $\mathcal{B}$ embeds the challenge in the test query and answers the test query as specified in Section 2.1 on page 3.

5.2.1 Patching the H_2 oracle

$\mathcal{B}$ has to maintain consistency between the H_2 oracle and *session key reveal* queries, as $\mathcal{B}$ will not be able to compute all data necessary to query the H_2 oracle for a valid session key in some instances (e.g. if certificateless public keys have been replaced by the adversary). If $\mathcal{B}$ has been asked on the H_2 oracle first and is then later asked a matching *session key reveal* query, $\mathcal{B}$ is always able to answer these requests correctly (it uses its decisional oracles that are explained in the proofs for respective strategies, see Section 5.4 on the following page). However, if $\mathcal{B}$ is asked a *session key reveal* query for which no matching H_2 query exists yet, B proceeds as follows: $\mathcal{B}$ inserts all available data and all data that $\mathcal{B}$ is able to compute (see also section 5.3 on the next page) into the H_2 oracle but may have to leave some fields (like K and K' or L and L' or N and N') empty. $\mathcal{B}$ chooses a random value from H_2's output domain as the session key and records that value together with the incomplete H_2 query data. For the following H_2 queries, $\mathcal{B}$ first checks if one of the incomplete entries of the H_2 oracle matches $\mathcal{M}$'s query data by using the respective decisional oracle(s). If that is the case, $\mathcal{B}$ records the complete information submitted by $\mathcal{M}$ and returns the H_2 entry. $\mathcal{B}$ additionally fills up all long term values that it can determine (even if it is not able to fill a H_2 entry completely). If $\mathcal{B}$ finds no matching entry, $\mathcal{B}$ simply generates a new H_2 entry as usual.

5.3 Handling a *session key reveal* query for sessions $\Pi_{i,j}^t$ where party i and j are not participating in the *test* query

Without loss of generality, we assume that i is the initiator of the session. Given party i that has incoming message $(r_{\mathcal{M}_j}P, x_{\mathcal{M}_j}P)$ (where $\mathcal{M}_j$ indicates that the values may be adversarial controlled) and that thus accepts, the challenger knows at least the identity based private keys and the ephemeral private key of party i, i.e. the challenger knows $sH_1(ID_i), sH_3(H_1(ID_i)), r_i$. The adversary may have replaced the certificateless public key of party i with $x_{\mathcal{M}_i}P$. To obtain a session key, party i has to query the H_2 oracle with the session data (as explained in Section 3.3 on page 8) on the following elements:

$$SK = H_2(i, j, r_iP, x_{\mathcal{M}_i}P, r_{\mathcal{M}_j}P, x_{\mathcal{M}_j}P, r_ir_{\mathcal{M}_j}P, x_{\mathcal{M}_i}x_{\mathcal{M}_j}P, r_ix_{\mathcal{M}_j}P, x_{\mathcal{M}_i}r_{\mathcal{M}_j}P, K, K', L, L', N, N')$$

Besides the public values $i, j, r_iP, x_{\mathcal{M}_i}P, r_{\mathcal{M}_j}P, x_{\mathcal{M}_j}P$ that are part of the H_2 query, the challenger acting as party i is able to compute the following values knowing its (possibly corrupted) private information $sH_1(ID_i), sH_3(H_1(ID_i)), r_i$:

$r_ir_{\mathcal{M}_j}P$ trivially, by computing $r_i(r_{\mathcal{M}_j}P)$

$r_ix_{\mathcal{M}_j}P$ by computing $r_i(x_{\mathcal{M}_j}P)$

K due to the patched H_1 oracle (see Table 2 on page 12), the challenger knows $\log_P H_1(ID_i) = l_i$ and $\log_P H_1(ID_j) = l_j$. Thus K can be computed as

$$K = e(H_1(ID_j), sP)^{r_i} e(l_isP, r_{\mathcal{M}_j}P)$$

K' just like for K, the challenger knows $\log_P H_3(H_1(ID_i)) = y_i$ and $\log_P H_3(\ H_1(ID_j)) = y_j$ (see Table 3 on the preceding page). Thus K' can be computed as

$$K' = e(H_3(H_1(ID_j)), sP)^{r_i} e(y_isP, r_{\mathcal{M}_j}P)$$

L Knowing l_i and l_j from the H_1 oracle computing L is easy:

$$L = e(l_j(x_{\mathcal{M}_i}P), sP)e(l_isP, x_{\mathcal{M}_j}P)$$

L' can be computed similarly, just like K' above.

N **and** N' are easy as the ID-based private keys are known.

The only missing values are $x_{\mathcal{M}_i}x_{\mathcal{M}_j}P$ and $x_{\mathcal{M}_i}r_{\mathcal{M}_j}P$ which cannot be computed by the challenger. However, as we point out in the proof for Strategy 1 in Section 5.4, the challenger is still able to answer *session state reveal* and H_2 queries consistently: If the challenger is asked a H_2 query first and then later asked a matching *session state reveal* query, the challenger can identify the corresponding H_2 entry by checking for all entries if $e(x_iP, x_jP) = e(x_ix_jP, P)$ and if $e(x_{\mathcal{M}_i}P, r_{\mathcal{M}_j}P) = e(x_{\mathcal{M}_i}r_{\mathcal{M}_j}P)$. If the challenger is asked a *session state reveal* query, but there is no matching H_2 entry, the challenger can create a new random value from the output domain of H_2 and assign it to the incomplete entry. The challenger checks the subsequent queries of the adversary to the H_2 oracle and is able to answer the queries correctly by using the pairing as above.

In the following, we will split the challenger's behaviour based on the strategy chosen in Section 5.1 on page 11. Additionally, we omit the indices $^t_{i,j}$ with respect to key computations for specific sessions to increase readability. Usually it is evident for which particular session the computations are needed. For the proof we assume that the adversary $\mathcal{M}$ does not get an advantage in outputting its guess $\hat{b}$ for b unless $\mathcal{M}$ queries the H_2 oracle on the session key.

5.4 Proofs for Strategy 1 to 9

5.4.1 Strategy 1

The allowed corrupt queries for the adversary are listed in Table 1 on page 12. The challenger $\mathcal{B}$ wants to use the adversary $\mathcal{M}$ to solve the computational Diffie-Hellman problem. The input for $\mathcal{B}$ is

$(aP, bP) \in \mathbb{G}^2$ and $\mathcal{B}$'s goal is to compute abP. To this end, $\mathcal{B}$ sets the certificateless public key of ID_I to aP and the certificateless public key of ID_J to bP. $\mathcal{B}$ uses the pairing to check whether the queries of the adversary to the H_2 oracle are valid: by computing $e(aP, bP) = e(abP, P)$, $\mathcal{B}$ is able to identify valid queries. As soon as $\mathcal{B}$ finds such a query, $\mathcal{B}$ aborts the game and returns abP as solution of the CDH challenge.

The probability that $\mathcal{B}$ is able to find a solution to the CDH challenge is

$$Adv^{\mathcal{B}}(k)[CDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9q_0q_1^2}$$

$\mathcal{B}$ is able to compute all other elements (x_Ix_JP, K, K', L, L', N, N') that are necessary for H_2 queries as the respective private values are under $\mathcal{B}$'s control. If $\mathcal{M}$ is a Type II adversary as explained in Section 2.1 on page 3, $\mathcal{B}$ gives s to $\mathcal{M}$ at the start of the game. We note that as $\mathcal{B}$ knows s, $\mathcal{B}$ is able to generate ID-based private keys for any identity; thus the game does not have to be changed for Type II adversaries. We note that $\mathcal{M}$ is allowed to replace the certificateless public key of ID_I and/or ID_J after the test query has been issued.

If $\mathcal{M}$ replaces the certificateless public keys of other identities and asks reveal queries, $\mathcal{B}$ first uses the pairing to check for matching queries to the H_2 oracle. If no matching query is found, $\mathcal{B}$ first generates a random value v of the output domain of H_2, inserts the available session data together with v into the H_2 table as described in Section 5.2.1 on page 13 (i.e. everything including the certificateless public keys; except x_ix_jP which $\mathcal{B}$ cannot compute) and returns v. If $\mathcal{B}$ is then later asked H_2 queries containing the correct x_ix_jP and the certificateless keys x_iP and x_jP, $\mathcal{B}$ is able to tell so by using the pairing computation and completes the entries in the H_2 table wherever possible.

5.4.2 Strategy 2

The allowed corrupt queries for the adversary are listed in Table 1 on page 12. The challenger $\mathcal{B}$ wants to use the adversary $\mathcal{M}$ to solve the computational Diffie-Hellman problem. The input for $\mathcal{B}$ is $(aP, bP) \in \mathbb{G}^2$ and $\mathcal{B}$'s goal is to compute abP. To this end, $\mathcal{B}$ sets the ephemeral key of ID_I to aP and the ephemeral key of ID_J to bP in the test query. $\mathcal{B}$ uses the pairing to check whether the queries of the adversary to the H_2 oracle are valid: by computing $e(aP, bP) = e(abP, P)$, $\mathcal{B}$ is able to identify valid queries. As soon as $\mathcal{B}$ find such a query, $\mathcal{B}$ aborts the game and returns abP as solution of the CDH challenge.

The probability that $\mathcal{B}$ is able to find a solution to the CDH challenge is

$$Adv^{\mathcal{B}}(k)[CDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9q_0q_1^2}$$

As $\mathcal{M}$ is allowed to replace the certificateless public keys of any identity, $\mathcal{B}$ uses the technique described in Strategy 1 to decide how to answer *reveal* queries and H_2 queries.

5.4.3 Strategy 3 and 4

The allowed corrupt queries for the adversary are listed in Table 1 on page 12. For Strategy 3, we want to embed the CDH challenge in r_Ix_JP, because the input to other values used in the key derivation function can be corrupted by the adversary. Here, $\mathcal{B}$ selects the master private key $s \xleftarrow{\$} \mathbb{Z}_p$. $\mathcal{B}$ is able to provide ID-based secret keys for all identities, as $\mathcal{B}$ is in possession of the master secret key. Furthermore, $\mathcal{B}$ sets the certificateless public key of ID_I to $x_IP = aP$ and the ephemeral public key of party ID_J to $r_JP = bP$ in session $\Pi^T_{I,J}$. If the adversary is a Type II adversary as described in Section 2.1 on page 3, then $\mathcal{B}$ gives s to $\mathcal{M}$ at the start of the game.

Similar to Strategy 1 and 2, $\mathcal{B}$ checks the H_2 queries for entries where

$$e(P, r_Jx_IP) \stackrel{?}{=} e(aP, bP)$$

As soon as $\mathcal{B}$ finds such an entry, $\mathcal{B}$ aborts the game and returns r_Jx_IP as solution to the BDH challenge. The probability that this happens is lower bounded by

$$Adv^{\mathcal{B}}(k)[CDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9q_0q_1^2}$$

$\mathcal{B}$ uses the techniques described in Strategy 1 to deal with replaced certificateless keys of identities other than ID_I. We note that $\mathcal{M}$ is allowed to replace the certificateless public key of ID_I after the test query has been issued.

We note that as Strategy 4 is symmetric to Strategy 3, its probability of success is equal to the probability of success for Strategy 3. Only ID_I and ID_J are exchanged and the computational BDH challenge is embedded in $r_I x_J P$ instead of $r_J x_I P$.

5.4.4 Strategy 5 and 6

The allowed corrupt queries for Strategy 5 for the adversary are listed in Table 1 on page 12. The BDH challenge can only be embedded in L_2 if Strategy 5 is chosen, because the input to all other values used in the key derivation function can be corrupted by the adversary. To accomplish this, the challenger $\mathcal{B}$ sets the master public key to aP and implements the H_1 oracle as described in Table 2 on page 12, thus $H_1(ID_I) = bP$. $\mathcal{B}$ patches the H_3 oracle as described in Table 3 on page 13, thus $H_3(H_1(ID_I)) = H_3(bP) = y_{tbdh_1}P - zbP$. $\mathcal{B}$ can still generate private keys for all identities except ID_I by computing $sH(ID_i) = l_i aP$ and $sH_3(H_1(ID_i)) = y_i aP$. Additionally, $\mathcal{B}$ sets the certificateless public key of ID_J to cP.

A problem for $\mathcal{B}$ arises when the adversary asks *session key reveal* queries for other sessions than the test session that include ID_I and ID_J, or for sessions that include ID_I and another party for which the adversary issued a *replace public key* query. Whenever $\mathcal{B}$ is asked a reveal query, $\mathcal{B}$ first checks if the key derivation function H_2 was asked with a matching session string involving both ID_I and ID_J. As $\mathcal{B}$ is unable to compute L, $\mathcal{B}$ uses the twin bilinear Diffie-Hellman trapdoor (see Theorem 1 on page 9) to check if $\mathcal{M}$ submitted a valid query, i.e. if the query should be answered with a record from H_2 (if such a record exists). The challenger extracts the discrete logarithm for ID_J's private keys, l_J and y_J from the H_1 and H_3 oracle respectively ($H_3(H_1(ID_J)) = H_3(l_J P) = y_J P$ and $\mathcal{B}$ is able to extract both l_J and y_J). Then, $\mathcal{B}$ extracts L and L' from each entry that matches the session for which the reveal query is being asked, computes $L_1 = e(l_J aP, x_I P)$, $L'_1 = e(y_J aP, x_I P)$ and checks if

$$\begin{aligned}
\left(\frac{L}{L_1}\right)^z \cdot \frac{L'}{L'_1} &= \left(\frac{e(H_1(ID_J), P)^{sx_I} \cdot e(H_1(ID_I), P)^{sx_J}}{e(l_J aP, x_I P)}\right)^z \\
&\quad \cdot \frac{e(H_3(H_1(ID_J)), P)^{sx_I} \cdot e(H_3(H_1(ID_I)), P)^{sx_J}}{e(y_J aP, x_I P)} \\
&= \left(\frac{e(l_J P, aP)^{x_I} \cdot e(bP, P)^{ac}}{e(l_J aP, x_I P)}\right)^z \\
&\quad \cdot \frac{e(y_J P, P)^{ax_I} \cdot e(y_{tbdh_1}P - zbP, P)^{ac}}{e(y_J aP, x_I P)} \\
&= e(bP, P)^{acz_1} \cdot e(y_{tbdh_1}P - zbP, P)^{ac} \\
&= e(P, P)^{z_1 abc} e(P, P)^{y_{tbdh_1} ac - z_1 abc} = e(P, P)^{y_{tbdh_1} ac} \\
&\stackrel{?}{=} e(aP, cP)^{y_{tbdh_1}}
\end{aligned}$$

As soon as $\mathcal{M}$ submits such an entry to the H_2 oracle, $\mathcal{B}$ aborts the game and returns

$$\begin{aligned}
\frac{L}{L_1} &= \frac{e(H_1(ID_J), P)^{sx_I} \cdot e(H_1(ID_I), P)^{sx_J}}{e(l_J aP, x_I P)} = \frac{e(l_J P, aP)^{x_I} \cdot e(bP, P)^{ac}}{e(l_J aP, x_I P)} \\
&= e(P, P)^{abc}
\end{aligned}$$

as solution to the BDH challenge.

$\mathcal{B}$ uses the same strategy for reveal queries to sessions of ID_I where the adversary replaced the certificateless public key of ID_j, except that $\mathcal{B}$ does not abort the game if a matching H_2 query is found but returns the correct H_2 value. If no matching H_2 query is found, $\mathcal{B}$ proceeds as in Section 5.2.1 on page 13. If the adversary replaces the certificateless public key of ID_I, $\mathcal{B}$ additionally uses the strategy described in Strategy 1. We note that $\mathcal{M}$ is allowed to replace the certificateless public key of ID_J after the test query has been issued.

The probability that $\mathcal{B}$ is able to find a solution to the CBDH challenge is

$$Adv^{\mathcal{B}}(k)[CBDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9 q_0 q_1^2}$$

Strategy 6 is symmetric to Strategy 5, so it has the same probability (only ID_I and ID_J are exchanged). The BDH challenge is embedded in L_1 instead of L_2.

5.4.5 Strategy 7 and 8

The allowed corrupt queries for the adversary are listed in Table 1 on page 12. The BDH challenge can only be embedded in K_2, because the input to all other values used in the key derivation function can be corrupted by the adversary. Using this strategy, the challenger sets the master public key sP to aP (notice that $\mathcal{B}$ does not know s). $\mathcal{B}$ changes the mode of operation of the H_1 oracle so that H_1 operates as in Table 2 on page 12, thus $H_1(ID_I) = bP$. $\mathcal{B}$ patches the H_3 oracle as described in Table 3 on page 13, thus $H_3(H_1(ID_I)) = H_3(bP) = y_{tbdh_1}P - zbP$. $\mathcal{B}$ can still generate private keys for all identities except ID_I by computing $sH(ID_i) = l_iaP$ and $sH_3(H_1(ID_i)) = y_iaP$. As queries for ID_I's private keys were ruled out, this does not affect the overall success probability. Additionally, $\mathcal{B}$ sets the ephemeral public key of party $J \neq I$ that participates in the T^{th} oracle $\Pi^T_{I,J}$ to cP.

If the adversary has an advantage in this strategy, then $\mathcal{M}$ needs to query the H_2 oracle on the session key. To distinguish this entry from other H_2 queries, $\mathcal{B}$ re-computes $K_1 = e(aP, P)^{l_J r_I}$ and similarly the $K_1' = e(aP, P)^{y_J r_I}$. Then, $\mathcal{B}$ searches in the table of the H_2 oracle for an entry where

$$\left(\frac{K}{K_1}\right)^z \cdot \frac{K'}{K_1'} = e(aP, cP)^{y_{tbdh_1}}$$

$\mathcal{B}$ aborts the game as soon as such an entry is submitted to the H_2 oracle and returns K/K_1 as solution to the computational bilinear Diffie-Hellman challenge. The probability that this happens is lower bounded by

$$Adv^{\mathcal{B}}(k)[CBDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9q_0q_1^2}$$

A problem for $\mathcal{B}$ occurs if $\mathcal{M}$ replaces certificateless public keys. As $\mathcal{B}$ knows the ID-based private keys for all identities except ID_I, $\mathcal{B}$ can compute K, K', L, L', N and N' for any session except for sessions involving ID_I. $\mathcal{B}$ may be unable to compute x_ix_jP if $\mathcal{M}$ replaced both x_iP and x_jP but can use the pairing as described in Strategy 1. For reveal queries involving ID_I and replaced certificateless public keys, $\mathcal{B}$ uses the H_3 oracle as described in Strategy 5.

Strategy 8 is symmetric to Strategy 7, so it has the same probability (only ID_I and ID_J are exchanged). The BDH challenge is embedded in K_1 instead of K_2.

5.4.6 Strategy 9

The allowed corrupt queries for the adversary are listed in Table 1 on page 12. The BDH challenge will be embedded in N. To accomplish this, the challenger sets the master secret key to aP, $H_1(ID_I) = bP$, and $H_1(ID_J) = cP$. Additionally, the H_3 oracle (see Table 3 on page 13) is modified before the game starts so that $H_3(H_1(ID_I)) = H_3(bP) = y_{tbdh_1}P - zbP$ and $H_3(H_1(ID_J)) = H_3(cP) = y_{tbdh_2}P - zcP$.

A problem for $\mathcal{B}$ arises when the adversary asks *session key reveal* queries for other sessions than the test session that include ID_I and ID_J, or for sessions where the adversary $\mathcal{M}$ replaces the certificateless public keys of any of the target identities. In these cases the challenger is unable to computer neither N nor L. Whenever $\mathcal{B}$ is asked a *session key reveal* query, $\mathcal{B}$ first checks if H_2 was asked with a matching session string involving both ID_I and ID_J. As $\mathcal{B}$ is generally unable to compute either L or N, $\mathcal{B}$ uses the trapdoor as explained in Theorem 3 on page 10 for N and Theorem 2 on page 9 for L to check if $\mathcal{M}$ submitted a valid query, i.e. if the query should be answered with a record from H_2 (if such a record exists). To this end, $\mathcal{B}$ extracts L, L', N and N' from each entry that matches the session for which the reveal query is being asked, and checks if $\frac{N'}{N^{z^2}} = \frac{e(aP,P)^{y_{tbdh_1}y_{tbdh_2}}}{e(cP,aP)^{zy_{tbdh_1}}e(bP,aP)^{zy_{tbdh_2}}}$ and if

$L^z L' = e(aP, x_I P)^{y_{tbdh_2}} e(aP, x_J P)^{y_{tbdh_1}}$.

$$\begin{aligned} N &= e(P,P)^{abc} \\ N' &= e(P,P)^{a(y_{tbdh_1}-bz)(y_{tbdh_2}-cz)} \\ &= e(P,P)^{a(y_{tbdh_1}y_{tbdh_2}-y_{tbdh_1}cz-bzy_{tbdh_2}+bcz^2)} \\ &= e(P,P)^{abcz^2+ay_{tbdh_1}y_{tbdh_2}-y_{tbdh_1}zac-y_{tbdh_2}zab} \end{aligned}$$

$$\begin{aligned} \Rightarrow N^{-z^2}\cdot N' &= (e(P,P)^{abc})^{-z^2}\cdot e(P,P)^{abcz^2+ay_{tbdh_1}y_{tbdh_2}-y_{tbdh_1}zac-y_{tbdh_2}zab} \\ &= e(P,P)^{ay_{tbdh_1}y_{tbdh_2}-y_{tbdh_1}zac-y_{tbdh_2}zab} \\ &= \frac{e(aP,P)^{y_{tbdh_1}y_{tbdh_2}}}{e(aP,cP)^{y_{tbdh_1}z}\cdot e(aP,bP)^{y_{tbdh_2}z}}. \end{aligned}$$

$$\begin{aligned} L &= e(H_1(ID_J),P)^{sx_I} e(H_1(ID_I),P)^{sx_J} \\ &= e(cP,P)^{ax_I} e(bP,P)^{ax_J} = e(P,P)^{x_I ac} e(P,P)^{x_J ab} \\ L' &= e(H_3(H_1(ID_J)),P)^{sx_I} e(H_3(H_1(ID_I)),P)^{sx_J} \\ &= e(y_{tbdh_2}P - zcP,P)^{ax_I} e(y_{tbdh_1}P - zbP,P)^{ax_J} = e(P,P)^{ax_I y_{tbdh_2} - ax_I zc} e(P,P)^{ax_J y_{tbdh_1} - ax_J zb} \\ L^z\cdot L' &= (e(P,P)^{x_I ac} e(P,P)^{x_J ab})^z \cdot e(P,P)^{ax_I y_{tbdh_2} - ax_I zc} e(P,P)^{ax_J y_{tbdh_1} - ax_J zb} \\ &= e(P,P)^{x_I acz} e(P,P)^{x_J abz} \cdot \frac{e(P,P)^{ax_I y_{tbdh_2}}}{e(P,P)^{ax_I zc}} \frac{e(P,P)^{ax_J y_{tbdh_1}}}{e(P,P)^{ax_J zb}} \\ &= e(P,P)^{x_I acz - x_I zac} e(P,P)^{x_J abz - x_J zab} \cdot e(P,P)^{ax_I y_{tbdh_2}} e(P,P)^{ax_J y_{tbdh_1}} \\ &= e(P,P)^{ax_I y_{tbdh_2}} e(P,P)^{ax_J y_{tbdh_1}} = e(aP, x_I P)^{y_{tbdh_2}} e(aP, x_J P)^{y_{tbdh_1}} \end{aligned}$$

If no matching record exists, $\mathcal{B}$ patches the H_2 oracle as explained in Section 5.2.1 on page 13. As soon as $\mathcal{M}$ submits such an entry to the H_2 oracle, $\mathcal{B}$ aborts the game and returns N as solution to the BDH challenge. The probability that this happens is lower bounded by

$$Adv^{\mathcal{B}}(k)[CBDH] \geq \frac{Adv^{\mathcal{M}}(k)[\Pi]}{9q_0 q_1^2}$$

$\mathcal{B}$ is able to distinguish between H_2 queries that have correct session data and H_2 queries that have invalid session data and is thus able to operate the H_2 oracle consistently. $\mathcal{B}$ may have to use the techniques explained in Strategy 1 and Strategy 5 to operate the H_2 oracle.

Theorem 4 follows from the above strategies. □

6 Conclusion

We give the strongest security model for certificateless encryption and relate it to Type I and Type II adversaries [Den08]. We give the first construction for a strongly secure one round certificateless key agreement scheme that is proven to be secure in the random oracle model, if the computational bilinear Diffie-Hellman and the computational Diffie-Hellman assumptions hold. This enables us to positively answer Swanson's [Swa08, Chapter 7] first question, whether it is even possible to construct a certificateless key agreement scheme that meets the extended eCK model. The protocol is compatible with existing certificateless key infrastructures and can thus be deployed easily. It is furthermore a natural complement to certificateless encryption, which brings us to Swanson's second question: We show that a practical protocol for CL-AKE exists, although it is computationally expensive. We also show how the computational cost can be reduced if we use gap assumptions. We prove our scheme to be more secure than ID-based schemes, in the sense that the KGC can be more actively trying to learn secrets. To answer Swanson's third question, whether the flexibility of certificateless schemes is worth the increased likeliness of vulnerabilities, we note that the ability of the adversary to replace public keys does not necessarily have to introduce vulnerabilites. CL-AKE schemes therefore combine user flexibility with enhanced privacy.

It remains to devise computationally more efficient one round protocols for certificateless key agreement proven secure with respect to standard computational problems such as DH or BDH. Furthermore, a proof for a certificateless key agreement scheme in the standard model would be very interesting.

References

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless Public Key Cryptography. In Chi-Sung Laih, editor, *ASIACRYPT*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer, 2003. Online available at http://eprint.iacr.org/2003/126.pdf. 1, 5

[BF03] Dan Boneh and Matthew Franklin. Identity based encryption from the Weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003. Online available at http://crypto.stanford.edu/~dabo/papers/bfibe.pdf. 1

[BFMLS08] Kamel Bentahar, Pooya Farshim, John Malone-Lee, and Nigel P. Smart. Generic Constructions of Identity-Based and Certificateless KEMs. *J. Cryptology*, 21(2):178–199, 2008. 2

[CCS07] L. Chen, Z. Cheng, and Nigel P. Smart. Identity-based key agreement protocols from pairings. *Int. J. Inf. Sec.*, 6(4):213–241, 2007. 1, 6

[CKS08] David Cash, Eike Kiltz, and Victor Shoup. The Twin Diffie-Hellman Problem and Applications. In Nigel P. Smart, editor, *EUROCRYPT*, volume 4965 of *Lecture Notes in Computer Science*, pages 127–145. Springer, 2008. 1, 7, 8, 9

[DE02] Régis Dupont and Andreas Enge. Practical non-interactive key distribution based on pairings. Cryptology ePrint Archive, Report 2002/136, 2002. http://eprint.iacr.org/2002/136. 7

[Den08] Alexander W. Dent. A survey of certificateless encryption schemes and security models. *International Journal of Information Security*, 7(5):349–377, October 2008. 1, 2, 5, 18

[DLP08] Alexander W. Dent, Benoît Libert, and Kenneth G. Paterson. Certificateless encryption schemes strongly secure in the standard model. In Ronald Cramer, editor, *Public Key Cryptography*, volume 4939 of *Lecture Notes in Computer Science*, pages 344–359. Springer, 2008. 5

[HC08] Hai Huang and Zhenfu Cao. An ID-based Authenticated Key Exchange Protocol Based on Bilinear Diffie-Hellman Problem. Cryptology ePrint Archive, Report 2008/224, 2008. http://eprint.iacr.org/2008/224, to be published in ASIACCS'09. 6, 9

[KP05] Caroline Kudla and Kenneth G. Paterson. Modular Security Proofs for Key Agreement Protocols. In Bimal K. Roy, editor, *ASIACRYPT*, volume 3788 of *Lecture Notes in Computer Science*, pages 549–565. Springer, 2005. 8

[Kra05] Hugo Krawczyk. HMQV: A High-Performance Secure Diffie-Hellman Protocol. Cryptology ePrint Archive, Report 2005/176, 2005. http://eprint.iacr.org/2005/176. 1, 3, 7

[LL97] Chae Hoon Lim and Pil Joong Lee. A Key Recovery Attack on Discrete Log-based Schemes Using a Prime Order Subgroup. In *CRYPTO '97: Proceedings of the 17th Annual International Cryptology Conference on Advances in Cryptology*, pages 249–263, London, UK, 1997. Springer-Verlag. 7

[LLM07] Brian A. LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger Security of Authenticated Key Exchange. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *ProvSec*, volume 4784 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2007. 2, 3, 5, 6

[LMQ+03] Laurie Law, Alfred Menezes, Minghua Qu, Jerry Solinas, and Scott Vanstone. An Efficient Protocol for Authenticated Key Agreement. *Des. Codes Cryptography*, 28(2):119–134, 2003. 2

[LQ06] Benoît Libert and Jean-Jacques Quisquater. On Constructing Certificateless Cryptosystems from Identity Based Encryption. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography*, volume 3958 of *Lecture Notes in Computer Science*, pages 474–490. Springer, 2006. 2

[MT06] Tarjei K. Mandt and Chik How Tan. Certificateless Authenticated Two-Party Key Agreement Protocols. In Mitsu Okada and Ichiro Satoh, editors, *ASIAN*, volume 4435 of *Lecture Notes in Computer Science*, pages 37–44. Springer, 2006. 1, 2, 3

[SOK00] R. Sakai, K. Oghishi, and M. Kasahara. Cryptosystems based on pairing. In *Proceedings of Symposium on Cryptography and Information Security (SCIS 2000)*, pages 233–238, 2000. 6, 8

[Swa08] Colleen Marie Swanson. Security in Key Agreement: Two-Party Certificateless Schemes. http://uwspace.uwaterloo.ca/bitstream/10012/4156/1/Swanson_Colleen.pdf, 2008. Master Thesis, University of Waterloo. Download 2009-01-29. 1, 2, 3, 5, 18

[Ust08] Berkant Ustaoglu. Obtaining a secure and efficient key agreement protocol from (H)MQV and NAXOS. *Des. Codes Cryptography*, 46(3):329–342, 2008. 6

[WCW06] Shengbao Wang, Zhenfu Cao, and Licheng Wang. Efficient Certificateless Authenticated Key Agreement Protocol from Pairings. *Wuhan University Journal of Natural Sciences*, 11(5):1278–1282, 2006. 1

[XWSX08] Liang Xia, Shengbao Wang, Jiajun Shen, and Guoming Xu. Breaking and repairing the certificateless key agreement protocol from ASIAN 2006. *Wuhan University Journal of Natural Sciences*, 13(5):562–566, October 2008. 1, 2

[YL04a] Dae Hyun Yum and Pil Joong Lee. Generic Construction of Certificateless Encryption. In Antonio Laganà, Marina L. Gavrilova, Vipin Kumar, Youngsong Mun, Chih Jeng Kenneth Tan, and Osvaldo Gervasi, editors, *ICCSA (1)*, volume 3043 of *Lecture Notes in Computer Science*, pages 802–811. Springer, 2004. 2

[YL04b] Dae Hyun Yum and Pil Joong Lee. Generic Construction of Certificateless Signature. In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *ACISP*, volume 3108 of *LNCS*, pages 200–211. Springer, 2004. 2

[Zh05] Shao Zu-hua. Efficient authenticated key agreement protocol using self-certified public keys from pairings. *Wuhan University Journal of Natural Sciences*, 10(1):262–270, 2005. 1