

# On the relations between non-interactive key distribution, identity-based encryption and trapdoor discrete log groups

Kenneth G. Paterson · Sriramkrishnan Srinivasan

Received: 28 July 2008 / Revised: 12 February 2009 / Accepted: 16 February 2009 /  
Published online: 15 March 2009  
© Springer Science+Business Media, LLC 2009

**Abstract** This paper investigates the relationships between identity-based non-interactive key distribution (ID-NIKD) and identity-based encryption (IBE). It provides a new security model for ID-NIKD, and a construction that converts a secure ID-NIKD scheme satisfying certain conditions into a secure IBE scheme. This conversion is used to explain the relationship between the ID-NIKD scheme of Sakai, Ohgishi and Kasahara and the IBE scheme of Boneh and Franklin. The paper then explores the construction of ID-NIKD and IBE schemes from general trapdoor discrete log groups. Two different concrete instantiations for such groups provide new, provably secure ID-NIKD and IBE schemes. These schemes are suited to applications in which the Trusted Authority is computationally well-resourced, but clients performing encryption/decryption are highly constrained.

**Keywords** Identity-based encryption · Identity-based non-interactive key distribution · Trapdoor discrete logs

**Mathematics Subject Classification (2000)** 94A60

## 1 Introduction

The concept of identity-based cryptography (IBC) was first introduced by Shamir [35]. In IBC, arbitrary identifying strings such as e-mail addresses or IP addresses can be used to form public keys for users, with the corresponding private keys being created by a trusted authority (TA) who is in possession of a system-wide master secret. Then a party Alice who wishes, for example, to encrypt to a party Bob needs only know Bob's identifier and

---

Communicated by S. Galbraith.

---

K. G. Paterson (✉) · S. Srinivasan  
Information Security Group, Royal Holloway, University of London, Egham, Surrey TW20 0EX, UK  
e-mail: kenny.paterson@rhul.ac.uk

S. Srinivasan  
e-mail: s.srinivasan@rhul.ac.uk

the system-wide public parameters. This approach eliminates certificates and the associated processing and management overheads from public key cryptography.

Shamir [35] was able to construct an identity-based signature scheme, but the first fully secure and efficient identity-based encryption (IBE) schemes were not forthcoming until much later [6, 34]. Since then there has been an explosion of interest in IBC and the related field of cryptography based on pairings. A major open problem in IBC has been to construct an efficient IBE scheme whose security is *not* based on hardness assumptions in pairing-friendly groups (c.f. [32]). A candidate solution is the Cocks IBE scheme [11], whose security is based on the hardness of the quadratic residuosity problem modulo an RSA composite  $N$ . But this scheme suffers from ciphertext expansion. More recently, Boneh et al. gave an IBE scheme with security also based on the quadratic residuosity problem [7]. This scheme has short ciphertexts but rather large private keys, and both the encryption and decryption algorithms require non-trivial computational effort. An entirely different approach to IBE, based on lattices, was recently introduced by Gentry et al. [16]. Their IBE scheme enjoys efficient encryption and decryption, but has large public parameters and private keys. It also remains to be seen how parameters should be selected in practice for this scheme in order to attain a given level of security.

In a separate strand of work that pre-dates the pairing-based approach to IBE, Maurer and Yacobi [25–27] explored the use of special RSA moduli to construct groups in which knowledge of trapdoor information makes the discrete log problem (DLP) in the group easier to solve than when the trapdoor is not available. They used these groups to construct identity-based non-interactive key distribution (ID-NIKD) schemes. Such a scheme allows two entities who have each received a private key from a TA to compute a shared key without exchanging any messages; this key can then be used to derive a key for symmetric encryption, for example. According to [29], around the same time, Murakami and Kasahara adopted a similar approach as Maurer and Yacobi. Unfortunately, the schemes of Maurer and Yacobi have a somewhat troubled history, with a series of attacks and fixes having been presented for them [26, 24, 27, 21, 29]. Put simply, none of the schemes in [25–27] are secure in any reasonable security model for ID-NIKD, a situation that is yet to be rectified. Moreover, these schemes are not actually IBE schemes. This is because there is a subtle but important distinction between IBE and ID-NIKD: in the former, any party can send Bob an encrypted message knowing only Bob's identifier and the system parameters, while in the latter, both parties need to be enrolled in the system and in possession of their private keys in order to establish a shared key for encryption.

## 1.1 Contributions

In this paper, we revisit the early work of Maurer and Yacobi, investigating how general trapdoor discrete log (TDL) groups can be used to obtain ID-NIKD and IBE schemes. Our historical perspective allows us to shed light on the relationship between the ID-NIKD and IBE primitives. It also allows us to obtain new, provably secure IBE schemes that provide a different set of trade-offs between efficiency and bandwidth consumption compared to existing schemes.

We begin by establishing a new security model for ID-NIKD that more accurately captures the security desired from this primitive. Our new model is stronger than the existing model of Dupont and Enge [13] and is inspired by earlier models for (interactive) authenticated key exchange [2, 5, 3, 9]. Since ID-NIKD is itself a very useful cryptographic primitive with a wide range of applications in cryptography (see for example [37, 8, 1]), our new model should

be of independent interest. We show that the well-known ID-NIKD scheme of Sakai et al. [33] is secure in our model, thus strengthening the main result of [13].

Our next contribution is to show how to construct a secure ID-NIKD scheme from *any* TDL group  $G$ , under the assumptions that the Computational Diffie-Hellman (CDH) problem is hard in  $G$  and that there exists an efficient hash function  $H_1 : \{0, 1\}^* \rightarrow G$ . This hash function is modelled as a random oracle in our security analysis. Our construction generalises the approach taken by Maurer and Yacobi [25–27] to arbitrary TDL groups and provably fixes the flaws in these earlier schemes by appropriate use of hashing.

We then explore the relationship between IBE and ID-NIKD. We give a conversion that takes an ID-NIKD scheme that is secure in our model and that satisfies some mild technical conditions, and produces from it an IBE scheme that is secure in the IND-ID-CPA security model of [6]. The conversion itself works in the standard model. Chosen-ciphertext security in the random oracle model can easily be obtained by applying a secondary conversion, for example the conversions of [20, 38]. Our ID-NIKD-to-IBE conversion provides a framework within which existing ID-NIKD and IBE schemes can be related. For example, our conversion allows us to build an IBE scheme from the above-mentioned ID-NIKD scheme based on TDL groups. It also allows us to show how the Boneh–Franklin IBE scheme [6] arises from the ID-NIKD scheme of Sakai et al. [33]. Thus these quite different IBE schemes can be seen to arise in a uniform way from a single construction applied to well-known ID-NIKD schemes.

Finally, we investigate the ID-NIKD and IBE schemes that result when these constructions are instantiated in two different candidate TDL groups. The first instantiation yields provably secure ID-NIKD and IBE schemes for the RSA setting that was introduced by Maurer and Yacobi. Security for these schemes is directly related to the hardness of factoring the RSA modulus. The ID-NIKD scheme can be seen as a provably secure version of the Maurer–Yacobi scheme, while the IBE scheme appears to be new. The second instantiation uses a TDL group construction that is based on combining Weil descent and disguising certain elliptic curves via isogenies. This approach to constructing TDL groups was first sketched in [14] and fully developed in [36], but the application to identity-based cryptography is new. Security for the resulting ID-NIKD and IBE schemes depends on the hardness of the CDH problem for a particular class of elliptic curves.

At an 80-bit security level, our two IBE schemes are quite practical, the only issue being that private key generation is rather expensive for the TA. For example, for the IBE scheme built from elliptic curves and isogenies, the TA faces a one-time setup cost of about  $2^{48}$  bit operations, after which individual private key extractions are straightforward. This IBE scheme operates on general elliptic curves defined over  $F_{2^{161}}$ , enjoys compact public parameters and private keys, and has fast encryption and decryption. Thus we obtain IBE schemes whose security is not based on hardness assumptions in pairing-friendly groups and whose implementation does not require expensive pairing calculations. These characteristics make our new schemes ideal for use in applications where the TA can be assumed to have large amounts of computational power (such as government or military settings), but where the clients who perform encryption and decryption operations are constrained (as is the case in, for example, sensor networks or mobile ad hoc networks).

## 1.2 Related work

Murakami and Kasahara [29] recently reexamined the Maurer–Yacobi approach to constructing ID-NIKD schemes in the RSA trapdoor discrete log setting. They gave a careful exposition

of the options for constructing schemes and emphasized the need for appropriate hashing to avoid the square root attacks that plague the Maurer–Yacobi schemes. However, their paper does not contain any formal security analysis and does not consider IBE schemes. In fact, we will establish the security of one of the ID-NIKD schemes from [29] in this paper. In [22], Kunihiro et al. also revisited the Maurer–Yacobi approach and considered its practicality. However, they studied only versions of the Maurer–Yacobi schemes already known to be insecure.

We are not aware of previous work pointing out the relationship between ID-NIKD and IBE. Indeed, these two different primitives are sometimes confused in the literature. In the particular case of the ID-NIKD scheme of Sakai et al. [33] and the Boneh–Franklin IBE scheme [6], it has been noted by many authors that the same algebraic setting and keying method is used. But, as far as we are aware, their exact relationship has not been clarified until now. The fact that ID-NIKD schemes are known to be useful in constructing other cryptographic primitives is evident from [37, 8, 1], for example.

Several papers [17, 19, 30] sketch how TDL groups can be used to build ID-NIKD or IBE schemes. But to our knowledge, none of the previous work properly formalises this idea, nor provides any security analysis. In this sense some of our constructions and security results can be regarded as formalising folklore using the techniques of provable security. More generally, the utility of TDL groups in cryptography is widely acknowledged—see for example [31, 12]. Teske [36] suggests the use of TDL groups for escrowed encryption applications: each user selects his own public key and provides the relevant trapdoor information to the escrow authority. A virtue is then made of the fact that solving the DLP is *not* easy for the authority—this prevents the agency from decrypting individual user’s communications too easily. In our application of the ideas of [14, 36] to IBE, we use a single TDL group  $G$  as part of the public parameters, and then exploit the fact that many discrete logs in  $G$  can be extracted relatively cheaply after a significant pre-computation. We note that no mention of ID-based applications is made in [14, 36].

Rivest [32] outlines a construction for IBE based on the existence of trapdoor groups with infeasible inverses (TGIIs) that is attributed to Vaikuntanathan, and asks whether this is the only known sufficient condition for IBE outside of bilinear maps (i.e. pairing-friendly groups). Our work, in particular our construction of IBE from any TDL group in which the CDH problem is hard and for which hashing onto the group is possible, answers this question. Moreover, in both of our concrete constructions, taking inverses in the relevant group is easy, showing that the notion of a TDL group is distinct from that of a TGII. Indeed, there do not currently appear to be any good candidates for TGIIs.

Several authors [18, 23] have considered the construction of IBE schemes secure against collusions involving a limited number of participants. The benefit of their approach is that any group in which DDH is hard can be used, but the drawback is that the schemes are not secure in the accepted security model for IBE [6]. Moreover, the complexity of the encryption and decryption algorithms of the schemes is linear in  $k$ , the threshold of cheating participants that can be tolerated.

## 2 Background and definitions

In this section, we provide basic definitions needed for the remainder of the paper.

**Definition 1** (*Trapdoor discrete log generator*) A trapdoor discrete log group generator (TDL group generator) is defined by a pair algorithms  $\text{TDLGen}$  and  $\text{SolVeDL}$ :

- **TDLGen**: An algorithm that takes a security parameter  $1^k$  as input and outputs  $(G, r, g, T)$  where  $G$  is a (description of a) cyclic group of some order  $r$  with generator  $g$  and  $T$  denotes trapdoor information.
- **SolveDL**: A polynomial-time algorithm which takes as input  $1^k, (G, r, g, T)$  and a group element  $h$  and outputs  $a \in \mathbb{Z}_r$  such that  $h = g^a$ .

Note that we do not insist that the order  $r$  of the group  $G$  in the above definition be prime. This allows us to handle both RSA and elliptic curve settings. In the RSA setting,  $r$  must be kept secret by the party running the TDLGen algorithm, and, for technical reasons, we will need to assume that a suitable bound  $R$  on the group order is available as part of the description of  $G$ . We will see that a large multiple of the modulus  $N$  can play the role of  $R$  in the RSA setting. When disclosure of  $r$  does not impact on security we may include  $r$  directly as part of the group description. For now we insist that SolveDL run in time polynomial in  $k$ , but we will relax this in Sect. 5 in order to handle situations where private key generation may require time that is super-polynomial or even exponential in  $k$ , but that is still small enough for practical application for specific values of  $k$ . In this situation, extra care would need to be taken in the selection of scheme parameters to ensure that our security results to follow remain meaningful in practice.

**Definition 2** Let  $G$  with generator  $g$  be the output of algorithm TDLGen as in Definition 1. We define the advantage of an algorithm  $\mathcal{A}$  in solving the Computational Diffie-Hellman (CDH) problem in  $G$  to be

$$\text{Adv}_{\mathcal{A}}^{\text{CDH}}(k) = \Pr(\mathcal{A}(G, g, g^a, g^b) = g^{ab})$$

where  $a, b \xleftarrow{\$} \mathbb{Z}_r$ .

**Definition 3** Let  $G$  with generator  $g$  be the output of algorithm TDLGen as in Definition 1. We define the advantage of an algorithm  $\mathcal{A}$  in solving the Decisional Diffie-Hellman (DDH) problem  $G$  to be

$$\text{Adv}_{\mathcal{A}}^{\text{DDH}}(k) = \frac{1}{2} \left| \Pr(\mathcal{A}(G, g, g^a, g^b, g^{ab}) = 1) - \Pr(\mathcal{A}(G, g, g^a, g^b, Z) = 1) \right|$$

where  $a, b \xleftarrow{\$} \mathbb{Z}_r$  and  $Z \xleftarrow{\$} G$ .

We also want to work with pairing-friendly groups, and introduce a second generator for this setting:

**Definition 4** A pairing-friendly group generator PairingGen is a polynomial time algorithm with input  $1^k$  and output a tuple  $(G, G_T, e, q, P)$ . Here  $G, G_T$  are groups of prime order  $q$ ,  $P$  generates  $G$ , and  $e : G \times G \rightarrow G_T$  is a bilinear, non-degenerate and efficiently computable map. By convention,  $G$  is an additive group and  $G_T$  multiplicative.

For ease of presentation, we work exclusively in the setting where  $e$  is symmetric; our definitions and results can be generalised to the asymmetric setting where  $e : G_1 \times G_2 \rightarrow G_T$ , with  $G_1$  and  $G_2$  being different groups. Further details concerning the basic choices that are available when using pairings in cryptography can be found in [15].

**Definition 5** We define the advantage of an algorithm  $\mathcal{A}$  in solving the Bilinear Diffie-Hellman (BDH) problem in  $(G, G_T)$  to be

$$\text{Adv}_{\mathcal{A}}^{\text{BDH}}(k) = \Pr(\mathcal{A}(aP, bP, cP) = e(P, P)^{abc})$$

where  $a, b, c \xleftarrow{\$} \mathbb{Z}_q$ . Here, we implicitly assume that parameters  $(G, G_T, e, q, P)$  are given to  $\mathcal{A}$  as additional inputs.

**Definition 6** We define the advantage of an algorithm  $\mathcal{A}$  in solving the Decisional Bilinear Diffie-Hellman (DBDH) problem in  $(G, G_T)$  to be:

$$\text{Adv}_{\mathcal{A}}^{\text{DBDH}}(k) = \frac{1}{2} \left| \Pr(\mathcal{A}(aP, bP, cP, e(P, P)^{abc}) = 1) - \Pr(\mathcal{A}(aP, bP, cP, Z) = 1) \right|$$

where  $a, b, c \xleftarrow{\$} \mathbb{Z}_q$  and  $Z \xleftarrow{\$} G_T$ . Again, we assume that  $\mathcal{A}$  has the expected additional inputs.

**Definition 7** A function  $\epsilon(k)$  is said to be *negligible* if, for every  $c$ , there exists  $k_c$  such that  $\epsilon(k) \leq k^{-c}$  for every  $k \geq k_c$ .

**Definition 8** Consider a family of functions  $f = (f_k)_{k \in \mathbb{N}}$  where  $f_k : \{0, 1\}^* \rightarrow \mathcal{T}_k$  for some set  $\mathcal{T}_k$ . We define the advantage of an algorithm  $\mathcal{A}$  in breaking the one-wayness of function  $f_k$  to be:

$$\text{Adv}_{\mathcal{A}}^{\text{OW}}(k) = \Pr(f_k(x) = y : x' \xleftarrow{\$} \{0, 1\}^*, y = f_k(x'), x \leftarrow \mathcal{A}(y)).$$

Here,  $\mathcal{A}$  is given a description of  $f_k$  as part of its input.

The family  $(f_k)_{k \in \mathbb{N}}$  is said to be *one-way* if  $\text{Adv}_{\mathcal{A}}^{\text{OW}}(k)$  is a negligible function for all algorithms  $\mathcal{A}$  running in time polynomial in  $k$ .

### 3 ID-based non-interactive key distribution

We formally define an ID-based non-interactive key distribution (ID-NIKD) scheme by three distinct algorithms: *Setup*, *Extract* and *SharedKey*. Algorithms *Setup* and *Extract* are executed by the Trusted Authority (TA), while *SharedKey* can be executed by any entity in possession of its private key and the identifier of any other entity with which it wishes to generate a shared key.

- *Setup*: On input  $1^k$ , outputs a master public key (or system parameters)  $mpk$  and master secret key  $msk$ .
- *Extract*: On input  $mpk, msk$  and identifier  $ID \in \{0, 1\}^*$ , returns a private key  $S_{ID}$  from some space of private keys  $\mathcal{SK}$ .
- *SharedKey*: On input  $mpk$ , a private key  $S_{ID_A}$  and an identifier  $ID_B \in \{0, 1\}^*$ , where  $ID_B \neq ID_A$ , this algorithm returns a key  $K_{A,B}$  from some space of shared keys  $\mathcal{SHK}$  specified in  $mpk$ .

We require that, for any pair of identities  $ID_A, ID_B$ , and corresponding private keys  $S_{ID_A}, S_{ID_B}$ , *SharedKey* satisfies the constraint:

$$\text{SharedKey}(mpk, S_{ID_A}, ID_B) = \text{SharedKey}(mpk, S_{ID_B}, ID_A).$$

This ensures that entities  $A$  and  $B$  can indeed generate a shared key without any interaction. We will normally assume that  $\mathcal{SHK}$ , the space of shared keys, is  $\{0, 1\}^{n(k)}$  for some function  $n(k)$ . In practice, this can be arranged by hashing a “raw” key.

### 3.1 Definition of security for ID-based non-interactive key distribution

Dupont and Enge [13] introduced the first formal security model for ID-NIKD. We present our new model, discuss a variant of it, and then explain how it strengthens the model of [13].

Our model is stated in terms of a game between an adversary  $\mathcal{A}$  and a challenger  $\mathcal{C}$ .  $\mathcal{C}$  takes as input the security parameter  $1^k$ , runs algorithm  $\text{Setup}$  of the ID-NIKD scheme and gives  $\mathcal{A}$   $mpk$ . It keeps  $msk$  to itself.  $\mathcal{A}$  then makes queries of the following three types:

- **Extract**(ID):  $\mathcal{C}$  responds by running algorithm  $\text{Extract}$  of the ID-NIKD scheme with input  $(mpk, msk, \text{ID})$  to generate a private key  $S_{\text{ID}}$ .  $\mathcal{C}$  gives  $S_{\text{ID}}$  to  $\mathcal{A}$ .
- **Reveal**( $\text{ID}_A, \text{ID}_B$ ):  $\mathcal{C}$  responds by running  $\text{Extract}(mpk, msk, \text{ID}_A)$  to obtain a private key  $S_A$  and then  $\text{SharedKey}(mpk, S_A, \text{ID}_B)$  to generate a shared key  $K_{A,B}$ .  $\mathcal{C}$  gives  $K_{A,B}$  to  $\mathcal{A}$ .
- **Test**( $\text{ID}_A, \text{ID}_B$ ):  $\mathcal{C}$  responds by calculating  $K_{A,B}$  as above.  $\mathcal{C}$  then selects  $b \xleftarrow{\$} \{0, 1\}$ . If  $b = 0$  then  $\mathcal{C}$  gives  $K_{A,B}$  to  $\mathcal{A}$ ; if  $b = 1$ , then  $\mathcal{C}$  gives  $\mathcal{A}$  a random element from  $\mathcal{SK}$ .

$\mathcal{A}$ 's queries may be made adaptively and are arbitrary in number, except that  $\mathcal{A}$  is allowed to make only one **Test** query. A straightforward hybrid argument can be used to relate our model with a single **Test** query to a model allowing multiple **Test** queries for a fixed bit  $b$ . In our model, no query to the **Reveal** oracle is allowed on the pair of identities selected for the **Test** query (in either order), and no **Extract** query is allowed on either of the identities involved in the **Test** query. These last two conditions are necessary to prevent the adversary from trivially winning the security game.

Finally,  $\mathcal{A}$  outputs a bit  $b'$ , and wins the game if  $b' = b$ .  $\mathcal{A}$ 's advantage in this IND-SK (indistinguishability of shared key) security game is defined to be

$$\text{Adv}_{\mathcal{A}}^{\text{IND-SK}}(k) = |\Pr[b = b'] - 1/2|.$$

We say that an ID-NIKD scheme is IND-SK secure if for any polynomial time adversary  $\mathcal{A}$ , the function  $\text{Adv}_{\mathcal{A}}^{\text{IND-SK}}(k)$  is negligible.

A weaker “computational” version of this model can be obtained by removing the **Test** oracle and changing the win condition for the game to require that  $\mathcal{A}$  output the actual shared key  $K_{A,B}$  for two identities  $\text{ID}_A, \text{ID}_B$  neither of which is the subject of an **Extract** query, and that are not together the subject of a **Reveal** query. We refer to COMP-SK security in this case.

#### 3.1.1 Comparison to the model of Dupont and Enge:

We detail a number of differences between our security model for ID-NIKD and that of Dupont and Enge [13].

The model of [13] gives the adversary access to the **Extract** oracle, but not to a **Reveal** oracle. Thus it captures collusion attacks, but does not give the adversary any direct oracle access to shared keys. This is somewhat analogous to a “No Reveals” adversary that is sometimes considered in weak security models for (interactive) key exchange. This restriction means that the model of [13] does not even capture the very natural requirement that the adversary, even after capturing keys shared between some pairs of entities, should still not be able to compute further shared keys. An adversary can, of course, compute any given shared key after extracting the private key of one of the relevant entities. But the different key types (private keys and shared keys) may be afforded different levels of protection in practice, so differentiating between the different types of compromise that are possible gives

us a more refined model that may better represent real applications. For example, private keys may be stored and used only in tamper-resistant hardware, while shared keys may be used as transport keys and so be exposed to cryptanalysis or other forms of attack.

We note that, in our model, queries to the **Reveal** oracle can be *simulated* by making appropriate access to the **Extract** oracle. This leads to a reduction from our security model to a model in which the adversary does not have access to a **Reveal** oracle. However, this reduction requires a correct guess as to which identities will be involved in the **Test** query, implying that it is not tight.

The model of [13] requires the adversary to compute the shared key held between two entities in order to be judged successful, whereas an indistinguishability-based definition is stronger, and more closely aligned with existing models for key exchange. Thus the model of [13] is analogous to our COMP-SK security model, though it is still weaker than even that model since we provide access to a **Reveal** oracle.

Finally, for the avoidance of confusion, we note that no non-interactive key distribution scheme can meet the notion of *forward security* that is enjoyed by many interactive key distribution protocols.

### 3.2 ID-based non-interactive key distribution from trapdoor discrete log groups

We specify how to obtain an ID-NIKD scheme from any TDL group generator. We need to assume the existence of efficient hash functions  $H_1 : \{0, 1\}^* \rightarrow G$  and  $H_2 : G \rightarrow \{0, 1\}^n$  in our construction; these are modelled as random oracles in our security proof. The component algorithms of our TDL-based ID-NIKD scheme are defined as follows:

- **Setup**: On input  $1^k$ , this algorithm runs `TDLGen` of the TDL generator and obtains a tuple  $(G, r, g, T)$ . It outputs  $mpk = (G, g, H_1, H_2, n)$  where  $H_1 : \{0, 1\}^* \rightarrow G$  and  $H_2 : G \rightarrow \{0, 1\}^n$  are hash functions, and  $SHK = \{0, 1\}^n$ . It also outputs  $msk = (G, g, H_1, H_2, n, r, T)$ . Here  $n = n(k)$  will be the bit-length of shared keys in the ID-NIKD scheme.
- **Extract**: On input  $mpk, msk$  and identifier  $ID \in \{0, 1\}^*$ , this algorithm runs algorithm `SolveDL` on input  $H_1(ID)$  to obtain a value  $S_{ID} \in \mathbb{Z}_r$  such that  $g^{S_{ID}} = H_1(ID)$ . The algorithm then outputs  $S_{ID}$ .
- **SharedKey**: On input  $mpk$ , a private key  $S_{ID_A}$  and an identifier  $ID_B \in \{0, 1\}^*$ , where  $ID_B \neq ID_A$ , this algorithm outputs  $H_2(H_1(ID_B)^{S_{ID_A}}) = H_2(g^{S_{ID_A} S_{ID_B}}) \in \{0, 1\}^n$

It is clear that `SharedKey` defined in this way satisfies the requirement that entities  $A$  and  $B$  are able to compute a common key. Note that  $g$  need not be included in the public parameters. However, including it slightly simplifies our later presentation.

**Theorem 1** *The TDL-based ID-NIKD scheme is secure assuming the hardness of the CDH problem in groups  $G$  produced by the TDL group generator. In more detail, for any IND-SK adversary  $A$  against the TDL-based ID-NIKD scheme that makes  $q_i$  queries to hash function  $H_i$  for  $i = 1, 2$ , there is an algorithm  $B$  that solves the CDH problem in groups  $G$  produced by the TDL group generator with*

$$Adv_B^{CDH}(k) \geq Adv_A^{IND-SK}(k)/q_1^2 q_2.$$

*Moreover,  $B$  runs in time  $O(\text{time}(A))$ . The proof is in the random oracle model, i.e.  $H_1, H_2$  are modelled as random oracles.*

*Proof* Suppose there is an adversary  $\mathcal{A}$  against the TDL-based ID-NIKD scheme with advantage  $\epsilon$  and running time  $t$ . We show how to construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the CDH problem in groups  $G$  produced by the TDL generator.  $\square$

$\mathcal{B}$ 's input is  $(G, g, g^a, g^b)$  where  $G$  is a cyclic group produced by  $\text{TDLGen}$ ,  $g$  is a generator of  $G$  and  $(g^a, g^b)$  is an instance of the CDH problem in  $G$ .  $\mathcal{B}$ 's task is to compute  $g^{ab}$ , and it does this by acting as a challenger for  $\mathcal{A}$ . We assume here that  $r$ , the group order, is part of the description of  $G$ . After the proof, we sketch the modifications necessary to handle the case where only a bound  $R$  on  $r$  is available.

$\mathcal{B}$  gives  $\mathcal{A}$   $mpk = (G, g, H_1, H_2, n)$  where  $H_1$  and  $H_2$  are random oracles controlled by  $\mathcal{B}$ . Let  $q_1$  be a bound on the number of queries made to  $H_1$  by  $\mathcal{A}$  in the course of its attack; similarly let  $q_2$  be a bound on the number of queries made to  $H_2$ .  $\mathcal{B}$  chooses two distinct indices  $I$  and  $J$  uniformly at random from  $\{1, 2, \dots, q_1\}$  and a third index  $L$  uniformly at random from  $\{1, 2, \dots, q_2\}$ .  $\mathcal{A}$  makes a series of queries which  $\mathcal{B}$  answers as follows:

- **$H_1$  queries:**  $\mathcal{B}$  maintains a table to handle  $\mathcal{A}$ 's  $H_1$  queries. If ID already appears in an entry of the form  $(\text{ID}, c, h)$  in the table, then  $\mathcal{B}$  returns  $h$  in response to  $\mathcal{A}$ 's query. Otherwise, if  $\mathcal{A}$ 's  $i$ -th distinct query to  $H_1$  is on  $\text{ID}_i$ , then  $\mathcal{B}$  proceeds as follows:
  1. If  $i = I$ , then  $\mathcal{B}$  adds  $(\text{ID}_I, \perp, g^a)$  to the  $H_1$  table and returns  $g^a$ .
  2. If  $i = J$ , then  $\mathcal{B}$  adds  $(\text{ID}_J, \perp, g^b)$  to the  $H_1$  table and returns  $g^b$ .
  3. Otherwise,  $\mathcal{B}$  chooses  $c_i$  uniformly at random from  $\mathbb{Z}_r$ , adds  $(\text{ID}_i, c_i, g^{c_i})$  to the  $H_1$  table, and returns  $g^{c_i}$ .
  
- $\mathcal{B}$ 's responses to  $H_1$  queries are uniformly and independently generated, so  $\mathcal{B}$ 's simulation of  $H_1$  is indistinguishable from that in the real attack environment. Notice how knowledge of the group order  $r$  is used to ensure this.
- **$H_2$  queries:**  $\mathcal{B}$  maintains a table to handle  $\mathcal{A}$ 's  $H_2$  queries. If the query  $s$  already appears in an entry of the form  $(s, K)$  in the table, then  $\mathcal{B}$  returns  $K$  in response to  $\mathcal{A}$ 's query. Otherwise, if  $\mathcal{A}$ 's  $i$ -th distinct query to  $H_2$  is on  $s_i$ , then  $\mathcal{B}$  selects  $K_i \xleftarrow{\$} \{0, 1\}^n$ , adds  $(s_i, K_i)$  to the table, and returns  $K_i$  to  $\mathcal{A}$ . Again,  $\mathcal{B}$ 's responses to  $H_2$  queries are uniformly and independently generated.
- **Extract queries:** If  $\mathcal{A}$  makes an query on an **Extract** identifier ID,  $\mathcal{B}$  first makes an  $H_1$  query on ID if this has not already been done. If  $\text{ID} \in \{\text{ID}_I, \text{ID}_J\}$  then  $\mathcal{B}$  aborts the simulation. Otherwise,  $\mathcal{B}$  finds an entry  $(\text{ID}, c, h)$  in the  $H_1$  table and outputs  $c$ .
- **Reveal queries:** When  $\mathcal{A}$  makes a **Reveal** query on a pair of identities  $\{\text{ID}_i, \text{ID}_j\}$ ,  $\mathcal{B}$  first makes  $H_1$  queries on  $\text{ID}_i$  and  $\text{ID}_j$  if this has not already been done. If  $\{\text{ID}_i, \text{ID}_j\} = \{\text{ID}_I, \text{ID}_J\}$  then  $\mathcal{B}$  aborts the simulation. Otherwise, suppose  $|\{\text{ID}_i, \text{ID}_j\} \cap \{\text{ID}_I, \text{ID}_J\}| \leq 1$ . Then  $\mathcal{B}$  can obtain from the  $H_1$  table two entries  $(\text{ID}_i, c_i, h_i)$  and  $(\text{ID}_j, c_j, h_j)$ , where either  $c_i \neq \perp$  or  $c_j \neq \perp$ . If  $c_i \neq \perp$ , then  $\mathcal{B}$  responds with the value  $H_2(h_i^{c_j})$ , first making the  $H_2$  query if necessary. If  $c_j \neq \perp$ , then  $\mathcal{B}$  responds with the value  $H_2(h_j^{c_i})$ .
- **Test query:** At some point during the simulation  $\mathcal{A}$  makes a single **Test** query on a pair of identities. If  $\mathcal{A}$  does not choose  $\text{ID}_I$  and  $\text{ID}_J$  as the identities in this query, then  $\mathcal{B}$  aborts its interaction with  $\mathcal{A}$  and fails. Otherwise,  $\mathcal{B}$  outputs a randomly generated value  $K \in \{0, 1\}^n$ . Notice that the "correct" key that would be computed by  $\mathcal{B}$  in responding to this query is equal to  $H_2(g^{ab})$ .

This completes our description of  $\mathcal{B}$ 's simulation. When  $\mathcal{A}$  terminates by outputting a bit  $b'$  (or if  $\mathcal{A}$  exceeds its normal running time), then  $\mathcal{B}$  outputs the value  $s_L$  held in the  $L$ -th entry of the  $H_2$  list.

We now assess  $\mathcal{B}$ 's success probability. Let  $\mathcal{F}$  denote the event that  $\mathcal{B}$  is not forced to abort during its simulation and let  $\mathcal{G}$  denote the event that a query to  $H_2$  on input  $g^{ab}$  is made at some point during  $\mathcal{B}$ 's simulation. It is easy to see that  $\Pr(\mathcal{F}) \geq 1/q_1^2$ , and that, conditioned on event  $\mathcal{F}$  occurring, then up to the point where  $\mathcal{G}$  occurs,  $\mathcal{B}$ 's simulation is indistinguishable from that seen when  $\mathcal{A}$  interacts with a true challenger. Moreover, if  $\mathcal{F}$  occurs, but  $\mathcal{G}$  does not occur, then it is easy to see that  $\mathcal{A}$ 's advantage is zero. For then the shared key that should be held between the two identities involved in the **Test** query (namely  $ID_I$  and  $ID_J$ ) is equal to  $H_2(g^{ab})$ , and this is independent of  $\mathcal{A}$ 's view unless event  $\mathcal{G}$  occurs. Thus, conditioned on  $\mathcal{F}$  occurring, we have:

$$\Pr[b = b' | \neg \mathcal{G}] = \frac{1}{2}.$$

Hence, assuming event  $\mathcal{F}$  occurs, we have

$$\begin{aligned} \text{Adv}_{\mathcal{A}}^{\text{IND-SK}}(k) &= \left| \Pr[b = b'] - \frac{1}{2} \right| \\ &= \left| \Pr[b = b' | \mathcal{G}] \Pr[\mathcal{G}] + \Pr[b = b' | \neg \mathcal{G}] \Pr[\neg \mathcal{G}] - \frac{1}{2} \right| \\ &= \left| \Pr[b = b' | \mathcal{G}] \Pr[\mathcal{G}] + \Pr[b = b' | \neg \mathcal{G}] (1 - \Pr[\mathcal{G}]) - \frac{1}{2} \right| \\ &= \left| (\Pr[b = b' | \mathcal{G}] - \Pr[b = b' | \neg \mathcal{G}]) \Pr[\mathcal{G}] + \Pr[b = b' | \neg \mathcal{G}] - \frac{1}{2} \right| \\ &\leq \Pr[\mathcal{G}] + \left| \Pr[b = b' | \neg \mathcal{G}] - \frac{1}{2} \right| \\ &= \Pr[\mathcal{G}] \end{aligned}$$

On the other hand, when events  $\mathcal{F}$  and  $\mathcal{G}$  do occur,  $\mathcal{B}$  is successful in outputting  $g^{ab}$  with probability at least  $1/q_2$ , since in this case we know that  $g^{ab}$  is on the  $H_2$  list in some position, and  $\mathcal{B}$  selects as its output a random element from the list. Combining these facts, we see that  $\text{Adv}_{\mathcal{B}}^{\text{CDH}}(k) \geq \epsilon/q_1^2 q_2$ . This completes the proof.

The above proof uses  $\mathcal{B}$ 's knowledge of  $r$  to produce uniformly sampled elements from  $G$  with known discrete logs. When only an upper bound  $R$  on  $r$  is available (because the group order is hidden),  $\mathcal{B}$  cannot so easily achieve this. However, if we place  $\mathbb{Z}_r$  by  $\mathbb{Z}_R$  wherever it occurs in the proof, then  $\mathcal{B}$ 's simulation of  $H_1$  deviates from being perfectly uniform by a fraction of at most  $r/R$ , and this can be made negligibly small simply by increasing the size of  $R$ . For example, in the RSA setting, we can take  $R = 2^k N$  where  $1^k$  is the security parameter and  $N$  the modulus. Although the private keys  $c$  returned by  $\mathcal{B}$  to  $\mathcal{A}$  as the result of **Extract** queries are now no longer in  $\mathbb{Z}_r$ , they are still valid keys (satisfying  $g^c = H_1(\text{ID})$  in  $G$ ). Strictly, the definition of the scheme should then be altered slightly to ensure that the outputs of the **Extract** algorithm are also from  $\mathbb{Z}_R$ .  $\square$

A variant scheme whose security is based on the hardness of the DDH problem in the TDL group is obtained simply by omitting the hash function  $H_2$ , so that the shared key is defined to be the "raw" value  $g^{S_A S_B} \in G$ . We can obtain COMP-SK security based on the hardness of the CDH problem in the TDL group for this variant scheme. The proofs of these results use similar techniques as the proof of Theorem 1.

Our TDL-based ID-NIKD scheme presented here is a generalisation of the schemes given in [25] from specific TDL groups in the RSA setting to the setting of general TDL groups.

As we noted in the introduction, the schemes of [25] and their successors [26, 27] lack formal security analysis and all suffer from attacks of various kinds [24, 21, 29]. Our analysis above shows that what is needed to prevent the various attacks on the Maurer–Yacobi ID-NIKD schemes is the introduction of appropriate hash functions. We return to the issue of how to select parameters and construct appropriate hash functions in Sect. 5, where we examine RSA-based instantiations of our TDL-based schemes. A scheme of this type in the setting of anomalous elliptic curves over  $\mathbb{Z}_n$  was also sketched in [30], but the specific scheme was soon found to be insecure by the authors.

### 3.3 Security of the ID-NIKD scheme of Sakai, Ohgishi and Kasahara

In this section, we prove the security of the Sakai–Ohgishi–Kasahara (SOK) ID-NIKD scheme [33] in our extended security model. This strengthens the main result of [13].

The SOK ID-NIKD scheme makes use of a pairing-friendly group generator `PairingGen` and has the following algorithms:

- **Setup:** On input  $1^k$ , this algorithm runs `PairingGen` to obtain a tuple  $(G, G_T, e, q, P)$  with the usual properties. It selects  $s \xleftarrow{\$} \mathbb{Z}_q$  and outputs  $mpk = (G, G_T, e, q, P, P_0 = sP, H_1, H_2, n)$  where  $H_1 : \{0, 1\}^* \rightarrow G$  and  $H_2 : G_T \rightarrow \{0, 1\}^n$  are hash functions, and  $\mathcal{SK} = \{0, 1\}^n$ . It also outputs  $msk = s$ .
- **Extract:** On input  $mpk, msk$  and identifier  $ID \in \{0, 1\}^*$ , this algorithm outputs  $S_{ID} = sH_1(ID)$ .
- **SharedKey:** On input  $mpk$ , a private key  $S_{ID_A}$  and an identifier  $ID_B \in \{0, 1\}^*$ , where  $ID_B \neq ID_A$ , this algorithm outputs  $H_2(e(S_{ID_A}, H_1(ID_B))) \in \{0, 1\}^n$ .

It is clear from bilinearity of the map  $e$  that `SharedKey` defined in this way satisfies the requirement that entities  $A$  and  $B$  are able to compute a common key. It also should be pointed out that Dupont and Enge [13] analyse a slight generalisation of the SOK ID-NIKD scheme that operates in the more general setting of asymmetric pairings. Our analysis can also be transferred to this setting. Note too that, strictly speaking, there is no need to include the value  $P_0 = sP$  in the public parameters of the scheme. However, including it simplifies our later presentation.

**Theorem 2** *The SOK ID-NIKD scheme is secure assuming the hardness of the BDH problem in groups  $(G, G_T)$  produced by the pairing-friendly group generator `PairingGen`. In more detail, for any IND-SK adversary  $\mathcal{A}$  against the SOK ID-NIKD scheme that makes  $q_i$  queries to hash function  $H_i$  for  $i = 1, 2$ , there is an algorithm  $\mathcal{B}$  that solves the BDH problem in groups  $(G, G_T)$  produced by `PairingGen` with*

$$Adv_{\mathcal{B}}^{BDH}(k) \geq Adv_{\mathcal{A}}^{IND-SK}(k)/q_1^2q_2.$$

*Moreover,  $\mathcal{B}$  runs in time  $O(\text{time}(\mathcal{A}))$ . The proof is in the random oracle model, i.e.  $H_1, H_2$  are modelled as random oracles.*

*Proof* Suppose there is an adversary  $\mathcal{A}$  against the SOK ID-NIKD scheme with advantage  $\epsilon$  and running time  $t$ . We show how to construct an algorithm  $\mathcal{B}$  that uses  $\mathcal{A}$  to solve the BDH problem in groups  $(G, G_T)$  produced by `PairingGen`.

$\mathcal{B}$ 's input is  $(G, G_T, e, q, P, aP, bP, cP)$  where  $G, G_T$  are cyclic groups of prime order  $q$ ,  $P$  generates  $G$ ,  $e : G \times G \rightarrow G_T$  is a bilinear map, and  $(aP, bP, cP)$  is an instance of

the BDH problem in  $G$ .  $\mathcal{B}$ 's task is to compute  $e(P, P)^{abc}$ , and it does this by acting as a challenger for  $\mathcal{A}$ .

$\mathcal{B}$  gives  $\mathcal{A}$   $mpk = (G, G_T, e, q, P, P_0 = cP, H_1, H_2, n)$  where  $H_1$  and  $H_2$  are random oracles controlled by  $\mathcal{B}$ . Let  $q_1$  be a bound on the number of queries made to  $H_1$  by  $\mathcal{A}$  in the course of its attack; similarly let  $q_2$  be a bound on the number of queries made to  $H_2$ .  $\mathcal{B}$  chooses two distinct indices  $I$  and  $J$  uniformly at random from  $\{1, 2, \dots, q_1\}$  and a third index  $L$  uniformly at random from  $\{1, 2, \dots, q_2\}$ .  $\mathcal{A}$  makes a series of queries which  $\mathcal{B}$  answers as follows:

- **$H_1$  queries:**  $\mathcal{B}$  maintains a table to handle  $\mathcal{A}$ 's  $H_1$  queries. If  $ID$  already appears in an entry of the form  $(ID, d, h)$  in the table, then  $\mathcal{B}$  returns  $h$  in response to  $\mathcal{A}$ 's query. Otherwise, if  $\mathcal{A}$ 's  $i$ -th distinct query to  $H_1$  is on  $ID_i$ , then  $\mathcal{B}$  proceeds as follows:
  1. If  $i = I$ , then  $\mathcal{B}$  adds  $(ID_I, \perp, aP)$  to the  $H_1$  table and returns  $aP$ .
  2. If  $i = J$ , then  $\mathcal{B}$  adds  $(ID_J, \perp, bP)$  to the  $H_1$  table and returns  $bP$ .
  3. Otherwise,  $\mathcal{B}$  chooses  $d_i$  uniformly at random from  $\mathbb{Z}_q$ , adds  $(ID_i, d_i, d_i P)$  to the  $H_1$  table, and returns  $d_i P$ .

$\mathcal{B}$ 's responses to  $H_1$  queries are uniformly and independently generated.

- **$H_2$  queries:**  $\mathcal{B}$  maintains a table to handle  $\mathcal{A}$ 's  $H_2$  queries. If the query  $s$  already appears in an entry of the form  $(s, K)$  in the table, then  $\mathcal{B}$  returns  $K$  in response to  $\mathcal{A}$ 's query. Otherwise, if  $\mathcal{A}$ 's  $i$ -th distinct query to  $H_2$  is on  $s_i$ , then  $\mathcal{B}$  selects a random element  $K_i$  from  $G$ , adds  $(s_i, K_i)$  to the table, and returns  $K_i$  to  $\mathcal{A}$ . Again,  $\mathcal{B}$ 's responses to  $H_2$  queries are uniformly and independently generated.
- **Extract queries:** If  $\mathcal{A}$  makes an **Extract** query on an identifier  $ID$ ,  $\mathcal{B}$  first makes an  $H_1$  query on  $ID$  if this has not already been done. If  $ID \in \{ID_I, ID_J\}$  then  $\mathcal{B}$  aborts the simulation. Otherwise,  $\mathcal{B}$  finds an entry  $(ID, d, h)$  in the  $H_1$  table and outputs  $d(cP)$ .
- **Reveal queries:** When  $\mathcal{A}$  makes a **Reveal** query on a pair of identities  $\{ID_i, ID_j\}$ ,  $\mathcal{B}$  first makes  $H_1$  queries on  $ID_i$  and  $ID_j$  if this has not already been done. If  $\{ID_i, ID_j\} = \{ID_I, ID_J\}$  then  $\mathcal{B}$  aborts the simulation. Otherwise, suppose  $|\{ID_i, ID_j\} \cap \{ID_I, ID_J\}| \leq 1$ . Then  $\mathcal{B}$  can obtain from the  $H_1$  table two entries  $(ID_i, d_i, h_i)$  and  $(ID_j, d_j, h_j)$ , where either  $d_i \neq \perp$  or  $d_j \neq \perp$ . If  $d_i \neq \perp$ , then  $\mathcal{B}$  responds with the value  $H_2(e(h_j, d_i(cP)))$ , first making the  $H_2$  query if necessary. If  $d_j \neq \perp$ , then  $\mathcal{B}$  responds with the value  $H_2(e(h_i, d_j(cP)))$ .
- **Test query:** At some point during the simulation  $\mathcal{A}$  makes a single **Test** query on a pair of identities. If  $\mathcal{A}$  does not choose  $ID_I$  and  $ID_J$  as the identities in this query, then  $\mathcal{B}$  aborts its interaction with  $\mathcal{A}$  and fails. Otherwise,  $\mathcal{B}$  outputs a randomly generated value  $K \in \{0, 1\}^n$ . Notice that because of the way in which the simulation is set up, the "correct" key that would be computed by  $\mathcal{B}$  in responding to this query is equal to  $H_2(e(P, P)^{abc})$ .

This completes our description of  $\mathcal{B}$ 's simulation. When  $\mathcal{A}$  terminates by outputting a bit  $b'$  (or if  $\mathcal{A}$  exceeds its normal running time), then  $\mathcal{B}$  outputs the value  $s_L$  held in the  $L$ -th entry of the  $H_2$  list. Then the assessment of  $\mathcal{B}$ 's success probability is almost identical to that in the proof of Theorem 1, and we omit the details. □

A variant scheme whose security is based on the hardness of the DBDH problem is obtained simply by omitting the hash function  $H_2$ , so that the shared key is defined to be the "raw" value  $e(S_{ID_A}, H_1(ID_B)) \in G_T$ . We can also obtain COMP-SK security based on the hardness of the BDH problem for this variant scheme.

### 4 From ID-based non-interactive key distribution to identity-based encryption

We show how to construct an IBE scheme from any ID-NIKD scheme that meets two additional technical conditions. We will then show that if the ID-NIKD scheme is IND-SK secure, then the IBE scheme that results from our conversion is IND-ID-CPA secure in the sense of [6]. We note here that we think that a generic construction in the reverse direction is unlikely to be possible, since it seems difficult to obtain a non-interactive primitive from one that is intrinsically interactive. A KEM formulation of our construction is also possible. In some ways this would be a more natural approach to take, since both ID-NIKD schemes and KEMs are concerned with keys, and one can generically obtain IBE from suitable KEMs using results of [4]. However, we are interested in exploring the relationship between existing ID-NIKD and IBE schemes, and so have focussed here on an IBE formulation instead.

We begin by recalling the formal definitions of IBE and its security, then give our conversion.

An IBE scheme is defined in terms of four algorithms:

- **Setup**: On input  $1^k$ , outputs a master public key (or system parameters)  $mpk$  and master secret key  $msk$ .
- **Extract**: On input  $mpk, msk$  and identifier  $ID \in \{0, 1\}^*$ , returns a private key  $S_{ID}$ .
- **Encrypt**: On input  $mpk$ , identifier  $ID \in \{0, 1\}^*$  and message  $M$  (from some message space), returns a ciphertext  $C$ .
- **Decrypt**: On input  $mpk$ , a private key  $S_{ID}$  and a ciphertext  $C$ , returns either a message  $M$  or a failure symbol  $\perp$ .

We have the obvious correctness requirement that decryption “undoes” encryption.

Security models for IBE were first established in [6]. Chosen-ciphertext security is defined in terms of the following IND-ID-CCA game between an adversary  $\mathcal{A}$  and a challenger  $\mathcal{C}$ .  $\mathcal{C}$  takes as input the security parameter  $1^k$ , runs algorithm **Setup** of the IBE scheme and gives  $Ampk$ . It keeps  $msk$  to itself.  $\mathcal{A}$  then runs in two phases:

- **Phase 1**:  $\mathcal{A}$  issues a series of adaptively selected **Extract** and **Decrypt** queries on identities  $ID$  and identifier/ciphertext combinations  $(ID, C)$  of its choice. These are replied to by  $\mathcal{C}$  by using algorithms **Extract** and **Decrypt** and knowledge of  $msk$ .
- **Challenge**: After  $\mathcal{A}$  decides to end Phase 1, it outputs two equal length messages  $M_0, M_1$  and a challenge identifier  $ID^*$ .  $\mathcal{C}$  selects  $b \xleftarrow{\$} \{0, 1\}$ , sets  $C^* = \text{Encrypt}(mpk, ID^*, M_b)$  and gives  $C^*$  to  $\mathcal{A}$ . We require that  $ID^*$  not be the subject of any **Extract** query in Phase 1.
- **Phase 2**: This phase proceeds as in Phase 1, with the constraints that  $ID^*$  not be the subject of any **Extract** query and that the pair  $(ID^*, C^*)$  not be the subject of any **Decrypt** query.
- **Guess**: Finally  $\mathcal{A}$  outputs a bit  $b' \in \{0, 1\}$ .

The advantage of  $\mathcal{A}$  against the IBE scheme in the above IND-ID-CCA security game is defined to be:

$$Adv_{\mathcal{A}}^{\text{IND-ID-CCA}}(k) = \left| \Pr[b = b'] - \frac{1}{2} \right|$$

where the probability is measured over the random choices of coins of  $\mathcal{A}$  and  $\mathcal{C}$ . An IBE scheme is said to be IND-ID-CCA secure if  $Adv_{\mathcal{A}}^{\text{IND-ID-CCA}}(k)$  is negligible for all polynomial time adversaries  $\mathcal{A}$ .

A weaker notion of IND-ID-CPA security for IBE is obtained by removing the adversary’s access to the **Decrypt** oracle. The advantage of  $\mathcal{A}$  against the IBE scheme in the resulting IND-ID-CPA security game is defined to be:

$$Adv_{\mathcal{A}}^{\text{IND-ID-CPA}}(k) = \left| \Pr[b = b'] - \frac{1}{2} \right|$$

and an IBE scheme is said to be IND-ID-CPA secure if  $Adv_{\mathcal{A}}^{\text{IND-ID-CPA}}(k)$  is negligible for all polynomial time adversaries  $\mathcal{A}$ .

#### 4.1 The conversion

We are now ready to present our conversion from ID-NIKD to IBE. We require the ID-NIKD scheme to satisfy the following requirements:

- The `Extract` and `SharedKey` algorithms of the ID-NIKD scheme should, as a first step, hash the input identifier `ID` using a member  $h = h_k$  of a one-way hash function family  $(h_k)_{k \in \mathbb{N}}$  to produce an element  $U_{\text{ID}}$  in some set  $\mathcal{PK}$ , with all further computations in the `Extract` and `SharedKey` algorithms depending only on  $U_{\text{ID}}$  and not on `ID`. We assume that  $h$  is described in the public parameters of the scheme. We may think of  $U_{\text{ID}}$  as being the public key corresponding to the string `ID`. If this condition is satisfied, then, from the algorithm `SharedKey` with inputs  $mpk, S_{\text{ID}_A}, \text{ID}_B$  we can construct a new algorithm `SharedKey'` with inputs  $mpk, S_{\text{ID}_A}, U_{\text{ID}_B} = h(\text{ID}_B)$  that has the same output as `SharedKey`.
- There should exist a randomized algorithm `Sample` that on input  $mpk$ , outputs pairs  $(S, U) \in \mathcal{SK} \times \mathcal{PK}$  with  $S$  being a private key corresponding to public key  $U$  and  $U \xrightarrow{\$} \mathcal{PK}$ . Note that an identifier corresponding to  $U$  will not be obtainable from  $U$  when it is generated in this way without inverting the one-way hash function  $h$ .

These conditions are certainly satisfied for the TDL-based and the SOK ID-NIKD schemes considered in Sect. 3. We do not know of any other ID-NIKD schemes. In both schemes, a hash function  $H_1$  is first used to convert identifiers into group elements before any further processing. Modelling  $H_1$  as a random oracle in the security analysis is a stronger assumption than  $H_1$  being one-way. For the second condition for the TDL-based scheme, we can define `Sample` as setting  $S \xrightarrow{\$} \mathbb{Z}_r$  and  $U = g^S$ . If  $r$  is not part of  $mpk$ , we may set  $S \xrightarrow{\$} \mathbb{Z}_R$  instead. For the SOK scheme, we can define `Sample` as setting  $b \xrightarrow{\$} \mathbb{Z}_q, S = bP_0$  and  $U = bP$ . Here we see the need for including  $g$  and  $P_0$  in the public parameters of the ID-NIKD schemes.

Assuming these two conditions are met for an ID-NIKD scheme  $S$ , we construct an IBE scheme  $\text{IBE}(S)$  as follows:

- `Setup`: On input  $1^k$ , this algorithm runs the `Setup` of  $S$  (with the same input) to obtain  $mpk, msk$ . The master public key of  $\text{IBE}(S)$  is set to  $mpk$  and the master secret key is set to  $msk$ . We assume that  $mpk$  contains a description of the shared key space  $\mathcal{SHK}$ , and that  $\mathcal{SHK} = \{0, 1\}^n$ ; this will also define the message space of  $\text{IBE}(S)$ .
- `Extract`: On input  $(mpk, msk, \text{ID})$ , this algorithm runs the `Extract` algorithm of the ID-NIKD scheme  $S$  with the same input to obtain a private key  $S_{\text{ID}}$ . The output is  $S_{\text{ID}}$ .
- `Encrypt`: Let the input be  $mpk, \text{ID}, M \in \{0, 1\}^n$ . This algorithm runs `Sample` to obtain a pair  $(S, U)$ . It then runs `SharedKey` of scheme  $S$  on input  $(mpk, S, \text{ID})$  to obtain a key  $K \in \{0, 1\}^n$ . The output is  $C = (U, V)$  where  $V = M \oplus K$ .

- **Decrypt**: Let the input be  $(mpk, S_{ID}, C)$  with  $C = (U, V)$ . This algorithm runs  $\text{SharedKey}'$  (derived from  $\text{SharedKey}$  of  $S$ ) on input  $(mpk, S_{ID}, U)$  to obtain a key  $K$ . The output is  $M = V \oplus K$ .

Note that the **Setup** and **Extract** algorithms of the IBE scheme  $\mathcal{IBE}(S)$  are almost identical to those of the ID-NIKD scheme  $S$ . The main idea of the construction is that the encrypting party  $A$  can generate an ‘‘on-the-fly’’ key-pair  $(S, U)$  for each encryption to  $B$ ; running  $\text{SharedKey}(mpk, S, ID_B)$  allows  $A$  to generate a shared key with  $B$  having identifier  $ID_B$ , and sending the public key  $U$  to  $B$  as part of the ciphertext allows  $B$  to compute the same shared key by running  $\text{SharedKey}'(mpk, S_{ID_B}, U)$ . This shared key is then used to protect the message. This can be seen as an identity-based analogue of the classical conversion that turns the Diffie-Hellman key agreement protocol into the ElGamal encryption scheme.

The scheme can be generalised to handle an ID-NIKD scheme whose shared key space  $\mathcal{SK}$  is any set equipped with a group operation; messages are then constrained to also lie in  $\mathcal{SK}$ . Alternatively, the key  $K$  produced during **Encrypt** can be used to derive a key for a symmetric encryption scheme. We have the following security result:

**Theorem 3** *Suppose the ID-NIKD scheme  $S$  is IND-SK secure and satisfies the two conditions above. Suppose also that  $(h_k)_{k \in \mathbb{N}}$  used in the construction of  $S$  is a one-way function family. Then the IBE scheme  $\mathcal{IBE}(S)$  is IND-ID-CPA secure. More precisely, for any adversary  $\mathcal{A}$  against  $\mathcal{IBE}(S)$ , there are adversaries  $\mathcal{B}_1$  against the one-wayness of  $(h_k)$  and  $\mathcal{B}_2$  against the IND-SK security of  $S$  such that:*

$$Adv_{\mathcal{A}}^{\text{IND-ID-CPA}}(k) \leq Adv_{\mathcal{B}_1}^{\text{OW}}(k) + 2 \cdot Adv_{\mathcal{B}_2}^{\text{IND-SK}}(k).$$

Here,  $\mathcal{B}_1, \mathcal{B}_2$  have running time roughly the same as  $\mathcal{A}$ .

*Proof* Let  $\mathcal{A}$  be an adversary against  $\mathcal{IBE}(S)$  and let  $Adv_{\mathcal{A}}^{\text{IND-ID-CPA}}(k)$  denote its advantage. We construct from  $\mathcal{A}$  two distinct algorithms  $\mathcal{B}_1, \mathcal{B}_2$ . Algorithm  $\mathcal{B}_1$  attempts to break the one-wayness of hash function  $h$ , while algorithm  $\mathcal{B}_2$  attempts to break the ID-NIKD scheme. The subsequent joint analysis of these two algorithms will then give us our result.

$\mathcal{B}_1$  receives from its challenger  $\mathcal{C}_1$  a value  $U \in \mathcal{PK}$  and uses  $\mathcal{A}$  in an attempt to find a string  $ID' \in \{0, 1\}^*$  such that  $h(ID') = U$ . Clearly, if  $\mathcal{B}_1$  is successful, then it breaks the one-wayness of  $h$ .  $\mathcal{B}_1$  runs **Setup** of the scheme  $\mathcal{IBE}(S)$ , obtaining  $msk, mpk$ . Note that  $h$  is assumed to be described in  $mpk$ . Now  $\mathcal{B}_1$  gives  $mpk$  to  $\mathcal{A}$ .  $\mathcal{B}_1$  handles  $\mathcal{A}$ 's **Extract** queries using its knowledge of  $msk$  to run algorithm **Extract** of  $\mathcal{IBE}(S)$ . When  $\mathcal{A}$  submits challenge messages  $M_0, M_1$  and a challenge identifier  $ID^*$ ,  $\mathcal{B}_1$  sets  $C^* = (U, V)$  where  $V = M_b \oplus K$  and  $K = \text{SharedKey}'(mpk, S_{ID^*}, U)$ . Here,  $S_{ID^*}$  is obtained by running **Extract** and  $b \xleftarrow{\$} \{0, 1\}$ . Eventually  $\mathcal{A}$  outputs its bit  $b'$ . If at any point in the game,  $\mathcal{A}$  made a query to its **Extract** oracle on an identifier  $ID$  satisfying  $h(ID) = U$ , then  $\mathcal{B}_1$  now outputs  $ID$ . If  $ID^*$  satisfies  $h(ID^*) = U$ , then  $\mathcal{B}_1$  outputs  $ID^*$ . Otherwise,  $\mathcal{B}_1$  fails when  $\mathcal{A}$  outputs its bit. It is clear that the attack environment provided by  $\mathcal{B}_1$  to  $\mathcal{A}$  is indistinguishable from that provided by a real challenger. Moreover, if  $\mathcal{A}$  does at any point make a query (either an **Extract** query or during the challenge phase) involving an identifier  $ID' \in \{0, 1\}^*$  such that  $h(ID') = U$ , then  $\mathcal{B}_1$  breaks the one-wayness of  $h$ .

$\mathcal{B}_2$  receives from its challenger  $\mathcal{C}_2$  the public parameters  $mpk$  of the ID-NIKD scheme  $S$  and uses  $\mathcal{A}$  in an attempt to win against  $\mathcal{C}_2$  in the IND-SK security game for  $S$ . Let  $d$  denote the hidden bit used by  $\mathcal{C}_2$  in responding to  $\mathcal{B}_2$ 's **Test** query.  $\mathcal{B}_2$  begins by selecting  $ID' \xleftarrow{\$} \{0, 1\}^*$  and computing  $U = h(ID')$ .  $\mathcal{B}_2$  then passes  $mpk$  to  $\mathcal{A}$ .  $\mathcal{A}$ 's **Extract** queries

on identifiers  $ID$  are handled by  $\mathcal{B}_2$  by passing them to  $\mathcal{C}_2$  as **Extract** queries in the IND-SK game. However, if  $h(ID) = U$  for any of these queries, then  $\mathcal{B}_2$  aborts. When  $\mathcal{A}$  submits challenge messages  $M_0, M_1$  and a challenge identifier  $ID^*$ ,  $\mathcal{B}_2$  makes its **Test** query to  $\mathcal{C}_2$  on input  $(ID', ID^*)$ , receiving a value  $K$  in return.  $\mathcal{B}_2$  then selects  $b \xleftarrow{\$} \{0, 1\}$  and sets  $C^* = (U, V)$  where  $V = M_b \oplus K$ . However, if  $h(ID^*) = U$ , then  $\mathcal{B}_2$  aborts. Eventually  $\mathcal{A}$  outputs its bit  $b'$ . If  $b' = b$  then  $\mathcal{B}_2$  outputs bit  $d' = 0$ ; otherwise  $\mathcal{B}_2$  outputs  $d' = 1$ . The attack environment provided by  $\mathcal{B}_2$  to  $\mathcal{A}$  is indistinguishable from that provided by a real challenger, provided that  $\mathcal{B}_2$  does not abort. Moreover, when  $\mathcal{B}_2$  does not abort,  $\mathcal{B}_2$  makes only legal queries to its challenger  $\mathcal{C}_2$  – we can be sure that  $ID'$  is distinct from  $ID^*$  and that  $ID'$  is not involved in any **Extract** query. In this situation,  $\mathcal{B}_2$ 's advantage can, via a standard argument, be expressed as:

$$\text{Adv}_{\mathcal{B}_2}^{\text{IND-SK}}(k) = \frac{1}{2} |\Pr[d' = 0|d = 0] - \Pr[d' = 0|d = 1]|.$$

When  $d = 1$ , the key  $K$  returned by  $\mathcal{C}_2$  to  $\mathcal{B}_2$  as a result of the **Test** query is random in  $\mathcal{SK} = \{0, 1\}^n$ , and then the ciphertext  $C^*$  received by  $\mathcal{A}$  is independent of the bit  $b$ . Hence in this case,  $\mathcal{A}$  has zero advantage and  $\Pr[b = b'] = 1/2$ . So  $\Pr[d' = 0|d = 1] = 1/2$ . On the other hand, when  $d = 0$ , the key  $K$  returned by  $\mathcal{C}_2$  to  $\mathcal{B}_2$  is equal to the shared key for identifiers  $ID', ID^*$ , and hence  $C^*$  is a proper encryption of  $M_b$ . Then  $\Pr[d' = 0|d = 0] = \Pr[b = b'|\neg\mathcal{F}]$ , where  $\mathcal{F}$  denotes the event that  $\mathcal{B}_2$  aborts and  $\neg\mathcal{F}$  its complement. Combining this information, we obtain

$$\text{Adv}_{\mathcal{B}_2}^{\text{IND-SK}}(k) = \frac{1}{2} \cdot |\Pr[b = b'|\neg\mathcal{F}] - 1/2|.$$

Notice that  $\mathcal{A}$ 's view is identical when playing against either  $\mathcal{B}_1$  or  $\mathcal{B}_2$ , unless  $\mathcal{B}_2$  aborts. So the occurrence of  $\mathcal{F}$  is independent of which algorithm  $\mathcal{A}$  plays against. Notice too that if  $\mathcal{F}$  occurs, then  $\mathcal{B}_1$  successfully inverts  $h$ . Then we have

$$\begin{aligned} \text{Adv}_{\mathcal{A}}^{\text{IND-ID-CPA}}(k) &= |\Pr[b = b'] - 1/2| \\ &= |\Pr[b = b'|\mathcal{F}] \cdot \Pr[\mathcal{F}] + \Pr[b = b'|\neg\mathcal{F}] \cdot (1 - \Pr[\mathcal{F}]) - 1/2| \\ &= |(\Pr[b = b'|\mathcal{F}] - \Pr[b = b'|\neg\mathcal{F}]) \cdot \Pr[\mathcal{F}] + \Pr[b = b'|\neg\mathcal{F}] - 1/2| \\ &\leq \Pr[\mathcal{F}] + |\Pr[b = b'|\neg\mathcal{F}] - 1/2| \\ &= \text{Adv}_{\mathcal{B}_1}^{\text{OW}}(k) + 2 \cdot \text{Adv}_{\mathcal{B}_2}^{\text{IND-SK}}(k). \end{aligned}$$

This, together with the observation that the running times of  $\mathcal{B}_1$  and  $\mathcal{B}_2$  are roughly the same as that of  $\mathcal{A}$ , completes the proof. □

The above construction provides only IND-ID-CPA security for the scheme  $\mathcal{IBE}(S)$ . In the random oracle model, we may apply the generic conversions of [20, 38] to obtain an IBE scheme with IND-ID-CCA security, provided  $\mathcal{IBE}(S)$  satisfies a mild technical condition ( $\gamma$ -uniformity). The resulting scheme (using either conversion) will be only a little less efficient than  $\mathcal{IBE}(S)$ . In turn,  $\mathcal{IBE}(S)$  has roughly the same performance characteristics as the ID-NIKD scheme that it is built from. In fact, the generic conversion of [38] only requires a one-way security notion for the starting IBE scheme in order to obtain IND-ID-CCA security. We can achieve this notion of security for our ID-NIKD-to-IBE conversion under the weaker requirement that the ID-NIKD scheme be COMP-SK secure. This can allow a slightly more efficient overall construction for an IND-ID-CCA secure IBE scheme, since we can typically

obtain COMP-SK security using one less hash function than is needed for IND-SK security, and with a slightly tighter reduction. Moreover, it can be seen from the proof of Theorem 3 that no access to the **Reveal** oracle is needed during the simulation. This means that security in the weaker model for ID-NIKD originally proposed in [13] is actually sufficient for our application to IBE.

In the next sections, we will examine some specific IBE schemes that result from our conversion.

### 4.2 Applying the conversion

We have already explained how the TDL-based ID-NIKD scheme and the SOK ID-NIKD scheme meet the requirements for our conversion to be applicable.

In the TDL case, we obtain an IBE scheme with the following algorithms:

- **Setup**: On input  $1^k$ , this algorithm runs  $\text{TDLGen}$  of the TDL generator and obtains a tuple  $(G, r, g, T)$ . It outputs  $mpk = (G, g, H_1, H_2, n)$  where  $H_1 : \{0, 1\}^* \rightarrow G$  and  $H_2 : G \rightarrow \{0, 1\}^n$  are hash functions. It also outputs  $msk = (G, g, H_1, H_2, n, r, T)$ . Here  $n$  is the size of plaintext messages.
- **Extract**: On input  $mpk, msk$  and identifier  $ID \in \{0, 1\}^*$ , this algorithm runs algorithm  $\text{SolveDL}$  on input  $H_1(ID)$  to obtain a value  $S_{ID} \in \mathbb{Z}_r$  such that  $g^{S_{ID}} = H_1(ID)$ . The algorithm then outputs  $S_{ID}$ .
- **Encrypt**: On input  $mpk$ , identifier  $ID \in \{0, 1\}^*$  and message  $M$ , this algorithm returns a ciphertext  $C = (U, V)$  where  $S \xleftarrow{\$} \mathbb{Z}_r, U = g^S$  and  $V = M \oplus H_2(H_1(ID)^S)$ .
- **Decrypt**: On input  $mpk$ , a private key  $S_{ID}$  and a ciphertext  $C = (U, V)$ , this algorithm outputs  $M = V \oplus H_2(U^{S_{ID}})$ .

The IND-ID-CPA security of this IBE scheme is guaranteed by Theorems 1 and 3, assuming the hardness of the CDH problem in groups  $G$  produced by  $\text{TDLGen}$ . The scheme is an ID-based analogue of the ElGamal encryption scheme; indeed it can be constructed and its security analysed directly without going via our ID-NIKD-to-IBE conversion. However, we are not aware of this IBE scheme having been explicitly presented in the literature before. A variant of it without explicit hash functions and without any security analysis was sketched in [19]. We will explore specific instantiations of this TDL-based scheme in more detail in the next section.

In the SOK case, it is easy to see that the IND-ID-CPA secure IBE scheme that results from applying our conversion is nothing other than the  $\text{BasicIdent}$  IBE scheme of Boneh and Franklin [6]: sampling produces a pair  $S = bP_0$  and  $U = bP$ , and the ciphertext has the form  $C = (U, V)$  where  $V = M \oplus H_2(e(S, H_1(ID)))$ . Here,

$$e(S, H_1(ID)) = e(bP_0, H_1(ID)) = e(bP, sH_1(ID)) = e(U, sH_1(ID)) = e(U, S_{ID}),$$

showing how the recipient may perform decryption.

Thus our approach provides a “new” proof of security for this scheme (and since the IND-ID-CCA secure scheme  $\text{FullIdent}$  of [6] is effectively obtained via the later generic conversion of [38], a proof for  $\text{FullIdent}$  too). More importantly, our conversion demonstrates how seemingly quite different IBE schemes can be seen as arising in a uniform way from a common underlying primitive, namely ID-NIKD. Specifically, it explains in a new way the relationship between the SOK ID-NIKD scheme and the IBE schemes of Boneh and Franklin [6].

### 5 Instantiating the TDL-based schemes

In this section, we consider two different instantiations of our TDL-based ID-NIKD and IBE schemes. Since the relevant schemes are already described and proved secure in Sects. 3.2 and 4.2 in the context of general TDL groups, our focus here is on two different proposals for such groups. In both cases, the algorithm `SolveDL` no longer runs in polynomial time, but its execution represents a still feasible if challenging computation. However, this means that the groups no longer strictly comply with the formal definition of a TDL group in Sect. 2. This in turn means that care is needed in selecting security parameters so as to obtain schemes that are efficient for the TA and still secure against well-equipped adversaries. In much the same way as in, for example, the well-known paper of Boneh and Franklin [6], we do not seek to use the security results from Sects. 3.2 and 4.2 to directly perform a concrete security analysis. Instead, as in [6], we use the hardness of the identified computational problem as a guide to selecting parameters. In our case, this problem is the CDH problem in the relevant TDL group. If a concrete security analysis is desired, then the “cost” (in terms of looseness) of the security reductions in Theorems 1 and 3 must be taken into account. This will necessitate the selection of different security parameters, which will in turn impact on the efficiency and practicality of the schemes.

#### 5.1 The RSA setting

We present and evaluate an instantiation of a TDL group generator for the RSA setting. Our presentation borrows in part from [29] (but note that [29] contains only a heuristic security analysis). We assume the reader is familiar with basic number theory and algorithms for factorisation and discrete logs. See [28] for further background.

Let  $N = pq$  where  $p \equiv 3 \pmod 4$ ,  $q \equiv 1 \pmod 4$  and  $\gcd(p - 1, q - 1) = 2$ . Then the maximal order of any element in  $\mathbb{Z}_N^*$  is  $(p - 1)(q - 1)/2$ . For  $x \in \mathbb{Z}_N$ , let  $(\frac{x}{N})$  denote the Jacobi symbol. Let  $g$  be an element of  $\mathbb{Z}_N^*$  such that  $g_p = g \pmod p$  is primitive in  $\mathbb{Z}_p$  and  $g_q = g \pmod q$  is primitive in  $\mathbb{Z}_q$ . Then  $g$  has maximal order in  $\mathbb{Z}_N^*$  and  $(\frac{g}{N}) = 1$ . Let  $G$  denote the subgroup of  $\mathbb{Z}_N^*$  generated by  $g$ ;  $G$  will be our trapdoor discrete log group. We next explain how to hash onto  $G$ , and then discuss how to select  $p$  and  $q$  in order to build a trapdoor for extracting discrete logs in  $G$ .

It is easy to verify that  $G$  is exactly the set of all elements of  $\mathbb{Z}_N$  with Jacobi symbol 1. Moreover, from the choice of  $p$  and  $q$ , we have  $(\frac{-1}{N}) = -1$ . Let  $H$  be a hash function from  $\{0, 1\}^*$  onto  $\mathbb{Z}_N$ , and define

$$H_1(\text{ID}) = \left( \frac{H(\text{ID})}{N} \right) \cdot H(\text{ID}).$$

Since  $-1$  has Jacobi symbol  $-1$  in  $\mathbb{Z}_N^*$ , we see that outputs of  $H_1$  have Jacobi symbol 1, and so lie in  $G$  (unless a hash output lying in  $\mathbb{Z}_N \setminus \mathbb{Z}_N^*$  can be found, in which case  $N$  can be factored). If  $H : \{0, 1\}^* \rightarrow \mathbb{Z}_N$  is modelled as a random oracle, then  $H_1$  is a random oracle with output in  $G$ . In practice, it is easy to instantiate  $H$  with output in  $\mathbb{Z}_N$  using a hash function such as SHA-256.

We next discuss in more detail how  $p$  and  $q$  should be chosen in order to obtain a suitable trapdoor for the DLP. With the information  $p$  and  $q$ , discrete logs can be extracted in  $G$  by finding discrete logs in  $\mathbb{Z}_p$  and  $\mathbb{Z}_q$  and combining the results using Chinese remaindering. The idea, as originally sketched in [25], is to choose  $p$  and  $q$  so that  $p - 1$  and  $q - 1$  are

both  $B$ -smooth, where  $B$  is some bound to be determined. Then the Pohlig-Hellman algorithm can be combined with Pollard's rho algorithm to find discrete logs in  $\mathbb{Z}_p$  and  $\mathbb{Z}_q$  in time roughly  $O(\ell B^{1/2})$  where  $\ell$  is the number of prime factors in  $p - 1$  and  $q - 1$ , using essentially no storage. This algorithm can be parallelised. Now consider the case when the trapdoor information is not known. It is shown in [27, Lemma 2] that extracting discrete logs to the base  $g$  in this situation is at least as difficult as factoring  $N$ . On the other hand when  $N$  is large enough, the best known algorithm for factoring  $N$  will either be the Number Field Sieve (with running time  $L_N(1/3, c)$  for some constant  $c$ ) or Pollard's  $p - 1$  factoring algorithm. The latter algorithm requires running time  $O(B \log N / \log B)$  if  $p - 1$  and  $q - 1$  are both  $B$ -smooth and if both do have a prime factor of size comparable to  $B$ . It does not seem to be capable of parallelisation.

Thus we see that, provided  $N$  is chosen so that the Number Field Sieve is not effective, solving the DLP in  $G$  with the trapdoor information takes time  $O(\ell B^{1/2})$  while, without the trapdoor information, time  $O(B \log N / \log B)$  is needed. Roughly speaking, then, with the trapdoor, finding discrete logs takes time  $2^k$ , and without it, time  $2^{2k}$ , for some value  $k$  that can be selected at will. This gives us the basis of a trapdoor for the DLP in our group  $G$ , albeit where knowledge of the trapdoor only gives a square-root speed-up rather than a polynomial-time algorithm for discrete logs. For example, suppose  $N = pq$  with  $p$  and  $q$  as above has 1024 bits and  $B = 2^{80}$ . Then the NFS has complexity roughly  $2^{80}$  operations, as does Pollard's  $p - 1$  algorithm. So without the trapdoor, the DLP in  $G$  takes effort roughly  $2^{80}$ , while with the trapdoor, it takes effort roughly  $2^{40}$ .

The above discussion can be formalised in the form of a trapdoor discrete log generator. Applying our constructions, we obtain concrete ID-NIKD and IBE schemes that are provably secure in the random oracle model, assuming the hardness of the CDH problem in the group  $G$  consisting of elements in  $\mathbb{Z}_N$  with Jacobi symbol 1. It is known that the CDH problem in this group is at least as hard as factoring (see [28, Fact 3.80]). This security guarantee is, to the best of our knowledge, in contrast to all previously presented versions of these schemes. The IBE scheme is fairly efficient, with encryption requiring two exponentiations modulo  $N$  and decryption requiring one. Public parameters and private keys are compact. However, private key extraction is somewhat inefficient, requiring about  $2^{40}$  effort for each private key at the 80-bit security level. This is because, with the Pohlig-Hellman and Pollard rho algorithms, it does not seem possible to re-use any of the effort expended in extracting one log to find another. On the other hand, this effort can be spread across multiple processors and is storage-free. Re-use of effort would be possible if one instead used an index-calculus style algorithm for discrete logs in  $\mathbb{Z}_p$  and  $\mathbb{Z}_q$ . For parameters of practical interest, this would effectively trade a large pre-computation of the order of  $L_p(1/3, c)$  for rapid extraction of individual discrete logs thereafter. In our example where  $p$  has roughly 512 bits, this pre-computation is significant but feasible. The factorisation of RSA-512 [10] took 8000 mips years using the Number Field Sieve and one would expect a roughly similar pre-computation cost for finding discrete logs in  $\mathbb{Z}_p$  with  $p$  having 512 bits.

Finally, we note that an alternative approach to constructing  $N$  using more than 2 prime factors appears to give a greater asymmetry for the DLP with and without the trapdoor, at least for parameters of practical interest. Analysis in [25] and more recently [22] proceeds on this basis. However, we do not know how to securely and efficiently hash onto a large cyclic subgroup of  $Z_N^*$  in that case, and as the attacks on earlier ID-NIKD instantiations [25–27] and our security analysis show, proper hashing is vital for security. Thus this alternative approach does not seem to be compatible with security.

## 5.2 Trapdoor discrete logs from isogenies and weil descent

Galbraith et al. [14] proposed a method by which elliptic curves on which the DLP can be solved using Weil descent can be disguised using isogenies to look like random curves. Teske [36] developed these ideas, giving a more detailed complexity and security analysis. In essence, the construction is as follows.

We start with a special elliptic curve  $E$  defined over  $\mathbb{F}_{2^{161}}$  for which it is possible to construct an explicit group homomorphism  $\Phi$  from  $E$  into  $J_C(\mathbb{F}_{2^{23}})$ , the Jacobian of a hyperelliptic curve of genus 7 or 8 defined over  $\mathbb{F}_{2^{23}}$ . Let  $\langle P \rangle$  of order  $r$  be a large cyclic subgroup of the group of points on  $E$ , and assume  $P \notin \ker(\Phi)$ . Then the DLP in  $G = \langle P \rangle$  may be transferred using  $\Phi$  to  $J_C(\mathbb{F}_{2^{23}})$ , where sub-exponential algorithms using index-calculus techniques are available. It is estimated in [36, Sect. 2.2] that the DLP in  $J_C(\mathbb{F}_{2^{23}})$  can be solved using around  $2^{48}$  bit operations, involving a factor base of size  $2^{22}$ . Most of this work can be parallelised. In fact, individual discrete log problems in  $J_C(\mathbb{F}_{2^{23}})$  can be solved relatively easily once a pre-computation of this order of magnitude has been carried out: each new problem requires the generation of an extra relation for the index-calculus algorithm and a small amount of linear algebra. So, given that finding  $2^{22}$  relations takes about  $2^{48}$  bit operations, finding each additional relation should take about a further  $2^{26}$  bit operations.

Now from  $E$  we create a second elliptic curve  $E'$  having the same order as  $E$  by applying a “random walk of isogenies” to  $E$ . Explicit algorithms for calculating the defining equations of  $E'$  are given in [36]. The new curve  $E'$  is expected to be vulnerable to a Weil descent attack with very low probability, and indeed the best algorithm for solving the DLP in  $E'$  seems most likely to be the Pollard rho method, with running time  $O(2^{80})$  for this size of curve. It is possible to compute an explicit representation of the chain of the isogenies which maps  $E'$  back to  $E$ . Let  $\Psi$  denote the composition of the isogenies in this chain, and let  $P' \in E'$  denote the pre-image of  $P$  under this map. Then the DLP in  $G' = \langle P' \rangle$  of order  $r$  can be efficiently transferred into  $G$  using  $\Psi$ , and thence to a feasible DLP in  $J_C(\mathbb{F}_{2^{23}})$  using  $\Phi$ .

The construction of a trapdoor discrete log group in this setting is now straightforward: the trapdoor group is  $G'$  with generator  $P'$ . Without the trapdoor, this appears to be a subgroup of order  $r$  on a random curve over  $\mathbb{F}_{2^{161}}$  for which the most efficient algorithm for solving the DLP is the Pollard rho method. Detailed analysis supporting this statement can be found in [36, Sect. 5]. The trapdoor information is the pair of maps  $(\Psi, \Phi)$ . These can be used to efficiently map the DLP in  $G'$  into  $J_C(\mathbb{F}_{2^{23}})$ , where an index-calculus algorithm can be used to solve the DLP.

Applying our ID-NIKD and IBE constructions in this setting gives us schemes that are extremely efficient in every respect except for private key extraction, and which carry proofs of security based on the hardness of the CDH problem in  $G'$ , giving roughly 80 bits of security. To illustrate this efficiency claim further, consider the TDL-based IBE scheme of Sect. 4.2. Hashing onto elements  $(x, y) \in G'$  is straightforward. We may use an ordinary hash function to map bitstrings ID onto elements  $x \in \mathbb{F}_{2^{161}}$ . We then use an iterated approach to obtain a suitable point on  $E'$ : we repeatedly extend ID with prefixes and hash to produce an  $x$ -coordinate, until we can solve for  $y \in \mathbb{F}_{2^{161}}$  such that  $(x, y) \in E'$ . The success probability is roughly  $1/2$  on each iteration. Then we need to multiply by the co-factor  $|E'|/r$ , which can be arranged to be small, say 2 or 4 [36], to obtain an element of  $G'$ . Each private key is an element of  $\mathbb{Z}_r$  and can be represented using just 162 bits; public parameters are compact, containing a description of  $E'$ ,  $r$  and the generator  $P'$ . Encryption (resp. decryption) requires just 2 (resp. 1) point multiplications in  $E'$ , together with some hashing. Thus these operations are likely to be faster than in pairing-based schemes offering a similar security

level. Ciphertexts (in the IND-ID-CPA secure scheme) have lengths the same as plaintexts plus one group element, and so have an overhead of only 162 bits.

As we have indicated above, private key extraction, being based on an index-calculus style algorithm, requires a pre-computation step during which many relations are gathered and a large linear algebra computation to obtain a reduced matrix is carried out. After this step, extracting individual private keys is fairly easy because each key needs only one additional relation to be produced and then a small amount of linear algebra. For the specific parameters in [36], the pre-computation cost can be estimated at about  $2^{48}$  bit operations, and the cost of each private key extraction at about  $2^{26}$  bit operations. The pre-computation task can be parallelised. Notice that this deployment scenario is rather different from the one envisaged in [36]—there every individual generates his own curves  $E, E'$  and gives the trapdoor data  $(\Psi, \Phi)$  to an escrow agency, with the aim being to force the agency to expend considerable effort to break each key.

## 6 Conclusion and open problems

We have explored the relationships between ID-NIKD and IBE, showing how the IBE scheme of Boneh and Franklin [6] can be seen as arising from the ID-NIKD scheme of Sakai et al. [33]. We have given constructions for secure ID-NIKD and IBE schemes in TDL groups, and studied the instantiations of these schemes in two settings, the RSA-based setting of Maurer and Yacobi, and the disguised elliptic curve setting of Galbraith et al. [14] and Teske [36]. We obtained two provably secure IBE schemes with extremely attractive performance in every respect except private key generation.

Several open problems are raised by our work. From a theoretical perspective, it would be interesting to extend our constructions based on TDL groups to the hierarchical ID-based setting and to find constructions offering security in the standard model. It would also be interesting to explore if further known IBE schemes can be related to (presumably as yet unknown) ID-NIKD schemes. From a practical perspective, it would be nice to find an efficient and secure way of hashing onto  $\mathbb{Z}_N^*$  when  $N$  has more than two prime factors, since this would allow more flexibility in parameter choices in the RSA setting. In particular, this may allow us to make better use of index calculus algorithms in  $\mathbb{Z}_p$  for each factor  $p$ , shifting most of the cost of private key extraction to a pre-computation. One drawback of the disguised elliptic curve version of our TDL-based ID-NIKD and IBE schemes is that the scheme parameters do not seem to scale smoothly to higher security levels.

A closer examination of the techniques in [36] may reveal variations on this approach with more attractive scaling properties. Of course, the most important question of all is to find examples of TDL groups for which solving the DLP with the trapdoor and hashing onto the group are both easy. These would have many applications in cryptography.

**Acknowledgments** This research was sponsored in part by the US Army Research Laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon. The second author is supported by a Dorothy Hodgkin Postgraduate Award, funded by EPSRC and Vodafone and administered by Royal Holloway, University of London. The authors wish to thank Alex Dent, Andreas Enge, Steven Galbraith, Ueli Maurer, Juanma Gonzalez Nieto, Mike Scott and the anonymous referees for insightful comments during the development of this paper.

## References

1. Balfanz D., Durfee G., Shankar N., Smetters D., Staddon J., Wong H.-C.: Secret handshakes from pairing-based key agreements. In: Proceedings of the 2003 IEEE Symposium on Security and Privacy, pp. 180–196. IEEE Computer Society Press (2003).
2. Bellare M., Rogaway P.: Entity authentication and key distribution. In: Stinson D.R. (ed.) CRYPTO'93, LNCS 773, pp. 232–249. Springer-Verlag (1994).
3. Bellare M., Canetti R., Krawczyk H.: A modular approach to the design and analysis of authentication and key exchange protocols. In: 30th STOC, pp. 419–428. ACM Press (1998).
4. Bentahar K., Farshim P., Malone-Lee J., Smart N.P.: Generic constructions of identity-based and certificateless KEMs. *J. Cryptol.* **21**(2), 178–199 (2008).
5. Blake-Wilson S., Johnson D., Menezes A.: Key agreement protocols and their security analysis. In: Darnell M. (ed.) Cryptography and Coding, 6th IMA International Conference, LNCS 1355, pp. 30–45. Springer-Verlag (1997).
6. Boneh D., Franklin M.: Identity-based encryption from the Weil pairing. In: Kilian J. (ed.) CRYPTO 2001, LNCS 2139, pp. 213–229. Springer-Verlag (2001).
7. Boneh D., Gentry C., Hamburg M.: Space-efficient identity based encryption without pairings. In: 48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007), pp. 647–657. IEEE Computer Society (2007). Full version available at Cryptology ePrint Archive, Report 2007/177. <http://eprint.iacr.org/>.
8. Boyd C., Mao W., Paterson K.G.: Key agreement using statically keyed authenticators. In: Jakobsson M., et al. (eds.) ACNS 2004, LNCS 3089, pp. 248–262. Springer-Verlag (2004).
9. Canetti R., Krawczyk H.: Analysis of key-exchange protocols and their use for building secure channels. In: Pfitzmann B. (ed.) EUROCRYPT 2001, LNCS 2045, pp. 453–474. Springer-Verlag (2001).
10. Cavallar S., Dodson B., Lenstra A.K., Lioen W.M., Montgomery P.L., Murphy B., te Riele H., Aardal K., Gilchrist J., Guillerm G., Leyland P.C., Marchand J., Morain F., Muffett A., Putnam C., Putnam C., Zimmermann P.: Factorization of a 512-Bit RSA modulus. In: Peneel B. (ed.) EUROCRYPT 2000, LNCS 1807, pp. 1–18. Springer-Verlag (2000).
11. Cocks C.: An identity based encryption scheme based on quadratic residues. In: Honary B. (ed.) Cryptography and Coding, 8th IMA International Conference, LNCS 2260, pp. 360–363. Springer-Verlag (2001).
12. Dent A.W., Galbraith S.D.: Hidden pairings and trapdoor DDH groups. In: Hess F., Pauli S., Pohst M. (eds.) Algorithmic Number Theory: 7th International Symposium (ANTS VII), LNCS 4076, pp. 436–451. Springer-Verlag (2006).
13. Dupont R., Enge A.: Provably secure non-interactive key distribution based on pairings. *Discrete Appl. Math.* **154**(2), 270–276 (2006). See also cryptology ePrint archive, report 2002/136 (2002). <http://eprint.iacr.org/>.
14. Galbraith S., Hess F., Smart N.P.: Extending the GHS Weil descent attack. In: Knudsen L. (ed.) EUROCRYPT 2002, LNCS 2332, pp. 29–44. Springer-Verlag (2002).
15. Galbraith S.D., Paterson K.G., Smart N.P.: Pairings for cryptographers. *Discrete Appl. Math.* **156**, 3113–3121 (2008). Available from cryptology ePrint archive: report 2006/165 (2006). <http://eprint.iacr.org/>.
16. Gentry C., Peikert C., Vaikuntanathan V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner R.E., Dwork C. (eds.) Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 197–206, ACM (2008). Full version available from cryptology ePrint archive: report 2007/432 (2007). <http://eprint.iacr.org/>.
17. Gordon D.M.: Designing and detecting trapdoors for discrete log cryptosystems. In: Brickell E.F. (ed.) CRYPTO'92, LNCS 740, pp. 66–75. Springer-Verlag (1993).
18. Heng S.-H., Kurosawa K.:  $k$ -resilient identity-based encryption in the standard model. In: Okamoto T. (ed.) CT-RSA 2004, LNCS 2964, pp. 67–80. Springer-Verlag (2004).
19. Hühnelein D., Jacobson Jr. M.J., Weber D.: Towards practical non-interactive public-key cryptosystems using non-maximal imaginary quadratic orders. *Des. Codes Cryptogr.* **39**(3), 281–299 (2003).
20. Kitagawa T., Yang P., Hanaoka G., Zhang R., Watanabe H., Matsuura K., Imai H.: Generic transforms to acquire CCA-security for identity based encryption: the cases of FOPkc and REACT. In: Batten L.M., Safavi-Naini R. (eds.) ACISP 2006, LNCS 4058, pp. 348–359. Springer-Verlag (2006).
21. Kügler D., Maurer M.: A Note on the Weakness of the Maurer-Yacobi Squaring Method. Technical Report TI-15/99. Department of Computer Science, Technical University of Darmstadt, Darmstadt, Germany (1999).
22. Kunihiro N., Abe W., Ohta K.: Maurer-Yacobi ID-based key distribution revisited. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **89**(5), 1421–1424 (2006).

23. Lee W.-B., Liao K.-C.: Constructing identity-based cryptosystems for discrete logarithm based cryptosystems. *J. Netw. Comput. Appl.* **27**, 191–199 (2004).
24. Lim C.H., Lee P.J.: Modified Maurer-Yacobi's scheme and its applications. In: Seberry J., Zheng Y. (eds.) ASIACRYPT92, LNCS 718, pp. 308–323. Springer-Verlag (1992).
25. Maurer U., Yacobi Y.: Non-interactive public-key cryptography. In: Davies D.W. (ed.) EUROCRYPT91, LNCS 547, pp. 498–507. Springer-Verlag (1991).
26. Maurer U., Yacobi Y.: A remark on a non-interactive public-key distribution system. In: Rueppel R.A. (ed.) EUROCRYPT92, LNCS 658, pp. 458–460. Springer-Verlag (1993).
27. Maurer U.M., Yacobi Y.: A non-interactive public-key distribution system. *Des. Codes Cryptogr.* **9**(3), 305–316 (1996).
28. Menezes A.J., van Oorschot P.C., Vanstone S.A.: *Handbook of Applied Cryptography*. CRC Press (1997).
29. Murakami Y., Kasahara M.: Murakami-Kasahara ID-based key sharing scheme revisited—in comparison with Maurer-Yacobi schemes. *Cryptology ePrint archive*, report 2005/306 (2005). <http://eprint.iacr.org/>.
30. Okamoto T., Uchiyama S.: Security of an identity-based cryptosystem and the related reductions. In: Nyberg K. (ed.) EUROCRYPT98, LNCS 1403, pp. 546–560. Springer-Verlag (1998).
31. Paillier P.: Public-key cryptosystems based on composite-degree residuosity. In: Stern J. (ed.) EUROCRYPT99, LNCS 1592, pp. 223–238. Springer-Verlag (1999).
32. Rivest R.: Controlled algebras and GHs. Talk given at IPAM Workshop on “Securing Cyberspace: Applications and Foundations of Cryptography and Computer Security”, October (2006). Available from [http://www.ipam.ucla.edu/publications/scws1/scws1\\_6243.ppt](http://www.ipam.ucla.edu/publications/scws1/scws1_6243.ppt).
33. Sakai R., Ohgishi K., Kasahara M.: Cryptosystems based on pairing. In: *The 2000 Symposium on Cryptography and Information Security*, Okinawa, Japan, January, pp. 26–28 (2000).
34. Sakai R., Ohgishi K., Kasahara M.: Cryptosystems based on pairing over elliptic curve. In: *2001 Symposium on Cryptography and Information Security (SCIS2001)*, January (2001).
35. Shamir A.: Identity-based cryptosystems and signature schemes. In: Blakley G.R., Chaum D. (eds.) CRYPTO84, LNCS 196, pp. 47–53. Springer-Verlag (1985).
36. Teske E.: An elliptic curve trapdoor system. *J. Cryptol.* **19**(1), 115–133 (2006).
37. Tseng Y.-M., Jan J.-K.: ID-based cryptographic schemes using a non-interactive public-key distribution system. In: *ACSAC 1998*, pp. 237–243. IEEE Computer Society (1998).
38. Yang P., Kitagawa T., Hanaoka G., Zhang R., Matsuura K., Imai H.: Applying Fujisaki-Okamoto to identity-based encryption. In: Fossorier M., et al. (eds.) AAECC 2006, LNCS 3857, pp. 183–192. Springer-Verlag (2006).