48th Annual IEEE Symposium on Foundations of Computer Science

Space-Efficient Identity Based Encryption Without Pairings

(extended abstract)

Dan Boneh* Craig Gentry† Michael Hamburg
Stanford University Computer Science Department
{dabo, gentry, mhamburg}@cs.stanford.edu

Abstract

Identity Based Encryption (IBE) systems are often constructed using bilinear maps (a.k.a. pairings) on elliptic curves. One exception is an elegant system due to Cocks which builds an IBE based on the quadratic residuosity problem modulo an RSA composite N. The Cocks system, however, produces long ciphertexts. Since the introduction of the Cocks system in 2001 it has been an open problem to construct a space efficient IBE system without pairings. In this paper we present an IBE system in which ciphertext size is short: an encryption of an ℓ-bit message consists of a single element in $\mathbb{Z}/N\mathbb{Z}$ plus $\ell+1$ additional bits. Security, as in the Cocks system, relies on the quadratic residuosity problem. The system is based on the theory of ternary quadratic forms and as a result, encryption and decryption are slower than in the Cocks system.

1 Introduction

In an Identity Based Encryption (IBE) system any string can function as a public key [34]. IBE systems have found numerous applications in cryptography (see [11, 12, 6, 21, 39, 7, 4] to name a few) and computer security [2, 38, 28, 29, 35]. There are currently two approaches for constructing IBE systems. The first approach builds IBE systems using bilinear pairings [8, 5, 37, 33]. The resulting systems are efficient both in performance and ciphertext size. The rich structure of bilinear maps enables various extensions such as Hierarchical IBE [27, 24], anonymous IBE [7, 1, 10, 23], and many others.

The second approach, due to Cocks [15], builds an elegant IBE system based on the standard quadratic residuosity problem [30, p.99] modulo an RSA composite N (in the random oracle model). Ciphertexts in this system contain two elements in $\mathbb{Z}/N\mathbb{Z}$ for every bit of plaintext. Hence, the encryption of an ℓ-bit message key is of size $2\ell \cdot \log_2 N$. For example, when encrypting a 128-bit message key using 1024-bit modulus, one ends up with a ciphertext of size 32678 bytes. For comparison, pairing based methods produce a 36 byte ciphertext for comparable security.

A long standing open problem since [15] is the construction of a space efficient IBE system without pairings, namely a system with short ciphertexts. We construct such a system — an encryption of an ℓ bit message consists of a single element in $\mathbb{Z}/N\mathbb{Z}$ plus $(\ell+1)$ additional bits. Hence, ciphertext size is about $\ell + \log_2 N$. When encrypting a 128-bit message key the result is a ciphertext of size $1024 + 129 = 1153$ bits or 145 bytes. The system makes extensive use of the theory of quadratic forms [13]. In particular, encryption and decryption are based on an effective version of Legendre's famous three squares theorem.

Our main proposed IBE system is presented in Section 6. The system is related to the Cocks system and security is similarly based on the quadratic residuosity (QR) problem. As in the Cocks system, our IBE proof of security makes use of the random oracle model. However, the random oracle model is needed only for proving that the underlying Rabin signature scheme is existentially unforgeable. To make this precise we first prove security *in the standard model* under the Interactive QR assumption (IQR), namely under the assumption that the QR problem is difficult in the presence of a hash square root oracle. We then note that IQR is equivalent to QR in the random oracle model. We observe that the system is an anonymous IBE under the quadratic residuosity assumption. (The Cocks system is known to not be anonymous, although a variant of it is [19]).

Encryption time in our system is far from ideal. Encryption time in many practical public key systems such as RSA and existing IBE systems [8, 15] is cubic in the security parameter. Encryption time in our system is quartic in the security parameter per message bit. Decryption time, however, is cubic as in other systems. The bottleneck during encryption is the need to generate primes on the order of N. In Section 5.3 we show a time space tradeoff where the

*Supported by NSF and the Packard Foundation.
†Supported by the Herbert Kunzel Stanford Graduate Fellowship.
[1]The full version of this paper is available at
http://eprint.iacr.org/2007/177

0272-5428/07 $25.00 © 2007 IEEE
DOI 10.1109/FOCS.2007.50

number of primes to generate can be significantly reduced at the cost of a modest increase in the ciphertext size.

2 Definitions

Recall from [34, 8] that an Identity Based Encryption system (IBE) consists of four algorithms: *Setup*, *KeyGen*, *Encrypt*, *Decrypt*. The *Setup* algorithm generates system parameters, denoted by PP, and a master key MSK. The *KeyGen* algorithm uses the master key to generate the private key d_{ID} corresponding to a given identity ID. The *Encrypt* algorithm encrypts messages for a given identity (using the system parameters) and the *Decrypt* algorithm decrypts ciphertexts using the private key.

An IBE must remain secure against an attacker who can request private keys for identities of his choice. This is captured in the standard IBE security game [8], which also captures chosen ciphertext attacks. Beyond the basic semantic security requirements one can also require that the IBE be anonymous, namely that a ciphertext reveal no information about the identity used to create the ciphertext. Anonymous IBE is useful for a variety of applications such as searching on encrypted data [7, 1, 10, 35, 19]. Chosen ciphertext security, private key queries, and anonymity are captured in the following IBE security game:

Setup: The challenger runs *Setup*(λ) and gives the adversary $\mathcal{A}$ the resulting public parameters PP. It keeps MSK to itself. We set $\text{ID}_0^*,\ \text{ID}_1^* \leftarrow \perp$ and $C^* \leftarrow \perp$.

Queries: The adversary issues adaptive queries $q_1, q_2, \ldots$ where query q_i is one of:

- Private key query (ID_i) where $\text{ID}_i \neq \text{ID}_0^*,\ \text{ID}_1^*$. The challenger responds by running algorithm *KeyGen*$(\text{MSK}, \text{ID}_i)$ and sends the resulting private key d_i to $\mathcal{A}$.
- Decryption query (ID_i, C_i) where (ID_i, C_i) is neither (ID_0^*, C^*) nor (ID_1^*, C^*). The challenger responds by running algorithm *KeyGen*$(\text{MSK}, \text{ID}_i)$ to generate a private key d_i and then runs algorithm *Decrypt*(d_i, C_i). It sends the resulting plaintext to the adversary.
- A *single* encryption query

$$\big((\text{ID}_0, m_0),\ (\text{ID}_1, m_1)\big)$$

where ID_0, ID_1 are distinct from all previous private key queries and m_0, m_1 are two equal length plaintexts. The challenger picks a random bit $b \xleftarrow{\text{R}} \{0,1\}$ and sets

$$C^* \leftarrow \textit{Encrypt}(\text{PP}, \text{ID}_b, m_b),$$
$$\text{ID}_0^* \leftarrow \text{ID}_0, \qquad \text{ID}_1^* \leftarrow \text{ID}_1$$

It sends C^* to the adversary.

Guess: Eventually, the adversary outputs a guess $b' \in \{0,1\}$. The adversary wins if $b = b'$.

We define $\mathcal{A}$'s advantage in attacking the scheme $\mathcal{E}$ as

$$\text{IBEAdv}_{\mathcal{A},\mathcal{E}}(\lambda) = \left|\Pr[b = b'] - \frac{1}{2}\right|$$

The probability is over the random bits used by the challenger and the adversary. An adversary $\mathcal{A}$ in this game is called an ANON-IND-ID-CCA adversary. We will also consider three types of weaker adversaries:

- If $\mathcal{A}$ makes no decryption queries we say that $\mathcal{A}$ is an ANON-IND-ID-CPA adversary. This models an anonymous IBE under a chosen plaintext attack.
- If in the single encryption query used to create the challenge ciphertext C^*, the adversary uses $\text{ID}_0 = \text{ID}_1$ then we say that the adversary is an IND-ID-CCA adversary. This models a chosen ciphertext secure IBE that is not necessary anonymous [8].
- An adversary that makes no decryption queries and sets $\text{ID}_0 = \text{ID}_1$ is said to be an IND-ID-CPA adversary. This is the standard IBE security model under a chosen plaintext attack.

Definition 2.1. Let $\mathcal{S}$ be one of IND-ID-CPA, IND-ID-CCA or ANON-IND-ID-CPA. We say that an IBE system $\mathcal{E}$ is $\mathcal{S}$ secure if for all polynomial time $\mathcal{S}$ adversaries $\mathcal{A}$ we have that $\text{IBEAdv}_{\mathcal{A},\mathcal{E}}(\lambda)$ is a negligible function.

Notation: throughout the paper we let $\mathcal{ID}_\lambda$ denote the set of all identities ID. The size of the set grows with λ. As shorthand, we will often write $\mathcal{ID}$ instead of $\mathcal{ID}_\lambda$.

2.1 Jacobi symbols and the QR assumption

For a positive integer N, we use $J(N)$ to denote the set $\{x \in \mathbb{Z}/N\mathbb{Z} : \left(\frac{x}{N}\right) = 1\}$, where $\left(\frac{x}{N}\right)$ is the Jacobi symbol of x in $\mathbb{Z}/N\mathbb{Z}$. We use $\text{QR}(N)$ to denote the set of quadratic residues in $J(N)$. We base the security of our system on the following computational assumption.

Definition 2.2 (Quadratic Residuosity Assumption (QR))**.** Let $\text{RSAgen}(\lambda)$ be a PPT algorithm that generates two equal size primes p, q.

- Let $\mathcal{P}_{QR}(\lambda)$ be the distribution of (N, V) where:

$$(p,q) \xleftarrow{\text{R}} \text{RSAgen}(\lambda),\ \ N \leftarrow pq,\ \ V \xleftarrow{\text{R}} \text{QR}(N)$$

- Let $\mathcal{P}_{NQR}(\lambda)$ be the distribution of (N, V) where:

$$(p,q) \xleftarrow{\text{R}} \text{RSAgen}(\lambda),\ \ N \leftarrow pq,\ \ V \xleftarrow{\text{R}} J(N)\backslash\text{QR}(N)$$

Letting $D(\mathcal{A},\mathcal{P}) = \Pr[(N,V) \xleftarrow{R} \mathcal{P}(\lambda) \ : \ \mathcal{A}(N,V)=1]$, we say that the **QR assumption** holds for RSAgen if for all PPT algorithms $\mathcal{A}$, the function

$$\mathrm{QRAdv}_{\mathcal{A},\mathrm{RSAgen}}(\lambda) = \left|D(\mathcal{A},\mathcal{P}_{QR}(\lambda)) - D(\mathcal{A},\mathcal{P}_{NQR}(\lambda))\right|$$

is negligible. We call this quantity the QR advantage of $\mathcal{A}$ against RSAgen.

For completeness, we also prove our system secure without relying on random oracles. This, however, requires a stronger assumption we call the *interactive QR assumption*. Basically, the assumption says that the QR assumption holds relative to a square root oracle.

Definition 2.3 (Interactive Quadratic Residuosity Assumption (IQR))**.** Let H be a hash function such that for every integer N the function $H_N(\cdot)$ maps $\{0,1\}^*$ to $J(N)$. Let $\mathcal{O}$ be a square root oracle: for every N which is a product of two odd primes, $\mathcal{O}$ picks a quadratic non-residue $u_N \in J(N)$; it maps an input pair (N,x) to one of $H_N(x)^{1/2}$ or $(u_N\, H_N(x))^{1/2}$ in $\mathbb{Z}/N\mathbb{Z}$, depending on which value is a quadratic residue. The oracle $\mathcal{O}$ is chosen uniformly from the set of all such functions. We say that the **Interactive QR assumption** (IQR) holds for the pair (RSAgen, H) if for all PPT algorithms $\mathcal{A}$, the function

$$\left|D(\mathcal{A}^{\mathcal{O}},\mathcal{P}_{QR}(\lambda)) - D(\mathcal{A}^{\mathcal{O}},\mathcal{P}_{NQR}(\lambda))\right|$$

is negligible. We call this the IQR advantage $\mathrm{IQRAdv}_{\mathcal{A},(\mathrm{RSAgen},H)}(\lambda)$ of $\mathcal{A}$ against RSAgen and H. Note that for IQR to hold, it must be difficult to find collisions in H, since each collision allows the adversary to factor N with probability $1/2$.

Note that the oracle $\mathcal{O}$ is a Rabin signature oracle. For a messages $m \in \{0,1\}^*$ and public key (N, u_N), the value $\mathcal{O}(N,m)$ is the Rabin signature on m.

When H is a full-domain hash function modeled as a random oracle, the QR assumption implies the IQR assumption. Hence, our system will be secure in the standard model based on IQR and in the random oracle model based on the standard QR assumption.

Our IBE system also uses a pseudorandom function (PRF) as defined in [25].

3 An abstract IBE system with short ciphertexts

We begin by describing an abstract IBE system that produces short ciphertexts. The system uses a deterministic algorithm $\mathcal{Q}$ with the following properties.

Definition 3.1. Let $\mathcal{Q}$ be a deterministic algorithm that takes as input (N,R,S) where $N \in \mathbb{Z}^+$ and $R,S \in \mathbb{Z}/N\mathbb{Z}$. The algorithm outputs two polynomials $f,g \in \mathbb{Z}/N\mathbb{Z}[x]$. We say that $\mathcal{Q}$ is **IBE compatible** if the following two conditions hold:

- (Condition 1) If R and S are quadratic residues, then $f(r)g(s)$ is a quadratic residue for all square roots r of R and s of S.
- (Condition 2) If R is a quadratic residue, then $f(r)f(-r)S$ is a quadratic residue for all square roots r of R.

We construct an IBE compatible algorithm $\mathcal{Q}$ in Sections 4 and 5. Condition 1 implies that the Legendre symbol $(f(r)/N)$ is equal to $(g(s)/N)$, a fact that will be used during decryption. Condition 2 will be used to prove security.

A single bit system. As a warm up to our IBE construction, consider briefly the following simple IBE for one bit messages.

Setup(λ): generate $(p,q) \xleftarrow{R} \mathrm{RSAgen}(\lambda)$, $N \leftarrow pq$, and a random $u \xleftarrow{R} J(N) \setminus \mathrm{QR}(N)$. Output public parameters $\mathrm{PP} = (N,u,H)$ where H is a hash function $H : \mathcal{ID} \to J(N)$. The master key MSK is the factorization of N.

KeyGen$(\mathrm{MSK},\mathrm{ID})$: generate a private key by first setting $R \leftarrow H(\mathrm{ID})$. If $R \in \mathrm{QR}(N)$ set $r \leftarrow R^{1/2}$ and otherwise set $r \leftarrow (uR)^{1/2}$. Output r as the private key for ID. Note that the private key is essentially a Rabin signature on ID, as in the Cocks system.

Encrypt$(\mathrm{PP},\mathrm{ID},m)$: to encrypt $m \in \{\pm 1\}$ with public key ID pick a random $s \in \mathbb{Z}/N\mathbb{Z}$ and compute $S \leftarrow s^2$. Let $R \leftarrow H(\mathrm{ID})$. Run $\mathcal{Q}$ twice:

$$(f,g) \leftarrow \mathcal{Q}(N,\ R,\ S) \quad \text{and} \quad (\bar{f},\bar{g}) \leftarrow \mathcal{Q}(N,\ uR,\ S)$$

and encrypt m using the two Jacobi symbols:

$$c \leftarrow m \cdot \left(\frac{g(s)}{N}\right) \quad \text{and} \quad \bar{c} \leftarrow m \cdot \left(\frac{\bar{g}(s)}{N}\right)$$

Output the ciphertext $C \leftarrow (S,c,\bar{c})$.

Decrypt(C,r): decrypt $(S,c,\bar{c})$ using private key r. Let us first suppose that $R = H(\mathrm{ID})$ is in $\mathrm{QR}(N)$ so that $r^2 = R$. The decryptor runs $\mathcal{Q}(N,\ R,\ S)$ to obtain (f,g). By condition (1) of Definition 3.1 we know that

$$\left(\frac{g(s)}{N}\right) = \left(\frac{f(r)}{N}\right)$$

Hence the plaintext is obtained by setting $m \leftarrow c \cdot \left(\frac{f(r)}{N}\right)$. If R is a non-residue then uR is a quadratic residue and

$r^2 = uR$. We decrypt by running $\mathcal{Q}(N,\ uR,\ S)$ and recovering m from $\bar{c}$. Since $\mathcal{Q}$ is deterministic, both sender and receiver always obtain the same pairs (f, g) and $(\bar{f}, \bar{g})$.

This completes the description of the one bit abstract system. Condition (1) implies soundness. Condition (2) is used to prove semantic security under the QR assumption. We note that the Cocks system is not an instance of this system.

Remark: Throughout the paper we let S, s be values chosen by the Sender (encryptor). We let R, r be values chosen by the Receiver (decryptor).

3.1 The Multi-Bit Abstract IBE System

Observe that a single value S chosen by the encryptor can be used to encrypt multiple bits. To encrypt an ℓ-bit message we hash ID multiple times by computing $R_i \leftarrow H(\text{ID}, i)$ for $i = 1, \ldots, \ell$. Now each pair (S, R_i) can be used to encrypt one message bit. The ciphertext only grows by two bits $(c_i, \bar{c}_i)$ per message bit. Hence, when encrypting an ℓ bit message, the complete ciphertext is $C = (S, (c_1, \bar{c}_1), \ldots, (c_\ell, \bar{c}_\ell))$ whose length is $\lceil \log_2(N) \rceil + 2\ell$ bits. In Section 6 we show how to shrink the ciphertext length to $\lceil \log_2(N) \rceil + (\ell + 1)$ bits.

We describe the complete system, called "BasicIBE" in more detail.

Setup(λ): Generate $(p, q) \stackrel{\text{R}}{\leftarrow} \text{RSAgen}(\lambda)$, $N \leftarrow pq$, and a random $u \stackrel{\text{R}}{\leftarrow} J(N) \setminus \text{QR}(N)$. Output public parameters $\text{PP} = (N, u, H)$ where H is a hash function $H : \mathcal{ID} \times [1, \ell] \to J(N)$.
The master key MSK is the factorization of N and a random key K for a pseudorandom function $F_K : \mathcal{ID} \times [1, \ell] \to \{0, 1, 2, 3\}$.

KeyGen(MSK, ID, ℓ): Takes as input MSK, ID, and a message length parameter ℓ. It generates a private key for decrypting encryptions of ℓ-bit messages. For $j = 1, \ldots, \ell$ do:

$R_j \leftarrow H(\text{ID}, j) \in J(N)$
$w \leftarrow F_K(\text{ID}, j) \in \{0, 1, 2, 3\}$
let $a \in \{0, 1\}$ be such that $u^a R_j \in \text{QR}(N)$
let $\{z_0, z_1, z_2, z_3\}$ be the square roots of $u^a R_j$ in $\mathbb{Z}/N\mathbb{Z}$
Set $r_j \leftarrow z_w$

Output the decryption key $d_{\text{ID}} \leftarrow (\text{PP}, r_1, \ldots, r_\ell)$. The PRF F ensures that the key generator always outputs the same square roots for a given ID, but an adversary cannot tell ahead of time which of the four square roots will be output.

Encrypt(PP, ID, m): Takes as input PP, a user ID, and a message $m = m_1 \ldots m_\ell \in \{-1, 1\}^\ell$. It generates random $s \in \mathbb{Z}/N\mathbb{Z}$ and sets $S \leftarrow s^2$. For $j = 1, \ldots, \ell$ do:

1. $R_j \leftarrow H(\text{ID}, j)$, $(f, g) \leftarrow \mathcal{Q}(N, R_j, S)$, $(\bar{f}, \bar{g}) \leftarrow \mathcal{Q}(N, uR_j, S)$
2. $c_j \leftarrow m_j \cdot \left(\frac{g_j(s)}{N}\right)$ and $\bar{c}_j \leftarrow m_j \cdot \left(\frac{\bar{g}_j(s)}{N}\right)$

Set $c \leftarrow c_1 \cdots c_\ell$ and $\bar{c} \leftarrow \bar{c}_1 \cdots \bar{c}_\ell$ and output the ciphertext $C \leftarrow (S, c, \bar{c})$.

Decrypt(C, d_{ID}): Let $d_{\text{ID}} = (\text{PP}, r_1, \ldots, r_\ell)$.
For $j = 1, \ldots, \ell$ let $R_j \leftarrow H(\text{ID}, j)$ and do:

if $r_j^2 = R_j$ run $(f_j, g_j) \leftarrow \mathcal{Q}(N, R_j, S)$
and set $m_j \leftarrow c_j \cdot \left(\frac{f_j(r_j)}{N}\right)$
if $r_j^2 = uR_j$ run $(\bar{f}_j, \bar{g}_j) \leftarrow \mathcal{Q}(N, uR_j, S)$
and set $m_j \leftarrow \bar{c}_j \cdot \left(\frac{\bar{f}_j(r_j)}{N}\right)$

Output $m \leftarrow m_1 \ldots m_\ell$.

This completes the description of BasicIBE. Soundness of decryption follows from Condition 1.

Remark: The function H outputs elements in $J(N)$. This function can be easily implemented using a hash function H' that outputs elements in $\mathbb{Z}/N\mathbb{Z}$. Simply include in PP some element $z \in \mathbb{Z}/N\mathbb{Z}$ whose Jacobi symbol (z/N) is -1. Then, to compute $H(\text{ID}, j)$ first compute $x \leftarrow H'(\text{ID}, j)$. If $x \in J(N)$ output $H(\text{ID}, j) = x$, otherwise output $H(\text{ID}, j) = xz$. Either way, $H(\text{ID}, j) \in J(N)$ as required.

3.2 Security

We prove security of the IBE system BasicIBE in the random oracle model based on the QR assumption.

Theorem 3.2. *Suppose the QR assumption holds for RSAgen and F is a secure PRF. Then the IBE system* BasicIBE *is IND-ID-CPA secure when H is modeled as a random oracle. In particular, suppose $\mathcal{A}$ is an efficient IND-ID-CPA adversary. Then there exist efficient algorithms $\mathcal{B}_1, \mathcal{B}_2$ (whose running time is about the same as that of $\mathcal{A}$) such that*

$$\begin{aligned} \text{IBEAdv}_{\mathcal{A},\text{BasicIBE}}(\lambda) &\leq 2 \cdot \text{QRAdv}_{\mathcal{B}_1, RSAgen}(\lambda) \\ &+ \text{PRFAdv}_{\mathcal{B}_2, F}(\lambda). \end{aligned}$$

The proof is given in the full version of this paper [9]. We note that the bounds in Theorem 3.2 are tight, and do not depend on the number of private key queries from the adversary. Since the theorem is set in the random oracle model, we could avoid the additional PRF assumption by

implementing the PRF using the random oracle H. However, the proof is simpler and the result more concrete using a PRF.

The proof of Theorem 3.2 uses Condition (2) of Definition 3.1. Condition (2) is only used to satisfy the conditions of the following simple lemma, which is used to prove semantic security — it enables the challenger to create a challenge ciphertext that is well formed when S is in $\mathrm{QR}(N)$ but is random when S is not. This, in turn, lets us build a QR distinguisher from a semantic security attacker (see [9]).

Lemma 3.3. *Let $N = pq$ be an RSA modulus, $X \in \mathrm{QR}(N)$, and $S \in J(N)$. Let x be a random variable uniformly chosen from among the four square roots of X. Let f be a polynomial such that $f(x)f(-x)S$ is a quadratic residue for all four values of x. Then:*

- *when $S \notin \mathrm{QR}(N)$ the Jacobi symbol $(f(x)/N)$ is uniformly distributed in $\{\pm 1\}$;*
- *when $S \in \mathrm{QR}(N)$ then $(f(x)/N)$ is constant, namely the same for all four values of x.*

Proof. Let x, x' be two square roots of X such that $x' = x \bmod p$, but $x' = -x \bmod q$. Then the four square roots of X are $\{\pm x, \pm x'\}$. When $S \notin \mathrm{QR}(N)$ we have $\left(\frac{f(x)}{p}\right) = -\left(\frac{f(-x)}{p}\right)$, and the same on q. By the Chinese Remainder Theorem, $\left(\frac{f(x')}{p}\right) = \left(\frac{f(x)}{p}\right)$ but $\left(\frac{f(x')}{q}\right) = -1 \cdot \left(\frac{f(x)}{q}\right)$, so that $\left(\frac{f(x')}{N}\right) = -1 \cdot \left(\frac{f(x)}{N}\right)$. So of the four values $f(x), f(x'), f(-x), f(-x')$, the first two must have different Jacobi symbols, as must the last two. Hence, among the four symbols, two are $+1$ and two are -1.
When $S \in \mathrm{QR}(N)$ all four symbols are equal. □

Remark: The system BasicIBE is CPA secure, but is not anonymous — there are instances of $\mathcal{Q}$ for which anyone can test which identity created a given ciphertext. In Section 6 we make the system anonymous. Moreover, we shrink the ciphertext length to $\lceil \log_2(N) \rceil + (\ell + 1)$ bits. The resulting system can be proven secure in the standard model under the IQR assumption. We view the construction in Section 6 as our main result.

4 Concrete instantiation: an IBE compatible algorithm

To make our abstract system concrete, we need to give an IBE compatible algorithm $\mathcal{Q}$. Recall that $\mathcal{Q}$ must generate polynomials f and g that meet Conditions 1 and 2 of Definition 3.1. The algorithm works as follows.

Algorithm $\mathcal{Q}(N, R, S)$:

1. Construct a solution $(x, y) \in (\mathbb{Z}/N\mathbb{Z})^2$ to
$$Rx^2 + Sy^2 = 1 \qquad (*)$$
In Section 5 we describe algorithms for solving this equation. This is the main bottleneck in our system.
2. Output the polynomials
$$f(r) \leftarrow xr + 1 \quad \text{and} \quad g(s) \leftarrow 2ys + 2$$

We show that $\mathcal{Q}$ is IBE compatible. Let $R, S \in \mathbb{Z}/N\mathbb{Z}$. Let r be a square root of R and s a square root of S, if one exists. Condition 1 is met, since

$$\begin{aligned} f(r) \cdot g(s) &= 2xrys + 2xr + 2ys + 2 \\ &= 2xrys + 2xr + 2ys + 1 + Rx^2 + Sy^2 \\ &= (xr + ys + 1)^2 \pmod N . \end{aligned}$$

Condition 2 is met, since

$$f(r) \cdot f(-r) \cdot S = (1 - Rx^2)S = (Sy)^2 \pmod N$$

Hence we have a valid instantiation for $\mathcal{Q}$.

5 Algorithms and Optimizations

The instantiation of algorithm $\mathcal{Q}$ in the previous section needs an algorithm that given an integer N and $R, S \in \mathbb{Z}/N\mathbb{Z}$ outputs a pair $(x, y) \in \mathbb{Z}/N\mathbb{Z}^2$ such that

$$Rx^2 + Sy^2 = 1 \qquad (1)$$

Throughout the section we assume that $R \neq S$ and RS is in $(\mathbb{Z}/N\mathbb{Z})^*$. Recall that when encrypting an ℓ-bit message using the abstract system (Section 3), the encryptor must solve 2ℓ such equations. In particular, one needs pairs $(x_i, y_i),\ (\bar{x}_i, \bar{y}_i) \in \mathbb{Z}/N\mathbb{Z}^2$ such that

$$\begin{aligned} &R_i \cdot x_i^2 + S \cdot y_i^2 = 1 \quad \text{and} \qquad (2) \\ &(uR_i) \cdot \bar{x}_i^2 + S \cdot \bar{y}_i^2 = 1 \end{aligned}$$

for $i = 1, \ldots, \ell$. The decryptor needs a solution to ℓ of these equations.

5.1 A product formula

Although the encryptor needs solutions to the 2ℓ equations in (2), we show that solving only $\ell + 1$ equations is sufficient. We do so using a product formula that, given a solution to two equations, builds a solution to a third equation.

Lemma 5.1. *For $i = 1, 2$ suppose (x_i, y_i) is a solution to $A_i x^2 + By^2 = 1$. Then (x_3, y_3) is a solution to*

$$(A_1 A_2) \cdot x^2 + B \cdot y^2 = 1 \tag{3}$$

where $x_3 = (x_1 x_2)/(By_1 y_2 + 1)$
and $y_3 = (y_1 + y_2)/(By_1 y_2 + 1)$.

Proof. The proof is by direct substitution into (3). □

During encryption, the encryptor first finds solutions in $\mathbb{Z}/N\mathbb{Z}$ to the $\ell + 1$ equations

$$u \cdot x^2 + S \cdot y^2 = 1 \text{ and } R_i \cdot x^2 + S \cdot y^2 = 1 \tag{4}$$

for $i = 1, \ldots, \ell$, using the algorithms discussed in the next section. The encryptor can now use Lemma 5.1 to quickly find solutions to the remaining ℓ equations in (2). Simply apply Lemma 5.1 to the left equation in (4) and the ith right equation in (4) to obtain a solution to $(uR_i) \cdot x^2 + S \cdot y^2 = 1$, as required. The same applies during decryption.

Overall, during encryption and decryption we need only to construct solutions to $\ell + 1$ equations of the form (1). For convenience, from here on we set $R_0 = u$.

5.2 Constructing Solutions to $Rx^2 + Sy^2 = 1$ in $\mathbb{Z}/N\mathbb{Z}$

Now we turn to solving (1). Pollard and Schnorr[32] provided, as part of their cryptanalysis of the Ong-Schnorr-Shamir signature scheme[31], a beautiful polynomial time algorithm (assuming the GRH) for finding solutions to the equation $x^2 - Sy^2 = R$ in $\mathbb{Z}/N\mathbb{Z}$. Clearly, a solution to the Pollard-Schnorr equation (with $x \neq 0$) also gives a solution to (1). The algorithm, however, is relatively slow requiring one to generate $O(\log \log N)$ primes on the order of N.

We, instead, take a different approach. We lift (1) to the integers and consider the ternary quadratic form

$$\tilde{R}x^2 + \tilde{S}y^2 - z^2 = 0 \tag{5}$$

where $\tilde{R}, \tilde{S}, x, y, z \in \mathbb{Z}$ and $\tilde{R} = R,\ \tilde{S} = S \bmod N$. A solution to (5) in $\mathbb{Z}$ gives a solution to (1) in $\mathbb{Z}/N\mathbb{Z}$. Since N is odd, by adding multiples of N to $\tilde{R}$ and $\tilde{S}$ we can assume that $\gcd(\tilde{R}, \tilde{S}) = 1$, that $\tilde{R}$ is odd, and that $\tilde{S} = 1 \bmod 4$. Then, by quadratic reciprocity $\tilde{R}$ is a square mod $\tilde{S}$ if and only if $\tilde{S}$ is a square mod $\tilde{R}$. Let $S' \in [0, 4N]$ be an integer such that $S' = S \bmod N$ and $S' = 1 \bmod 4$. Then $\tilde{S} = S' \bmod 4N$.

A classic result of Legendre [13] says that, for such $\tilde{R}, \tilde{S}$, equation (5) has a solution (x, y, z) in $\mathbb{Z}$ whenever there exist integers r, s such that

$$r^2 = \tilde{R} \pmod{\tilde{S}} \quad \text{and} \quad s^2 = \tilde{S} \pmod{\tilde{R}} \tag{6}$$

Cremona and Rusin [18] present an effective version of Legendre's theorem. They present several algorithms that take as input integers $N, \tilde{R}, \tilde{S}, r, s$ satisfying (6) and output a solution to (5). At the heart of one of these algorithms is a lattice reduction in a simple 3-dimensional lattice of determinant $4\tilde{R}\tilde{S}$. For completeness, we present the algorithm in Appendix A.

Consequently, solving (1) reduces to finding integers r, s satisfying (6). To do so, one can look for the smallest (probable) prime $\tilde{R}$ along the arithmetic progression $\tilde{R} = R \bmod N$. Next one looks for the smallest (probable) prime $\tilde{S}$ along the arithmetic progression $\tilde{S} = S' \bmod 4N$ for which the Legendre symbol $(\tilde{S}/\tilde{R}) = 1$. Finally, one computes the appropriate modular square roots to find an r, s satisfying (6). Given r, s we obtain a solution to (5) which leads to a solution to (1) as required.

Fortunately, we do not need a full primality test – one can use a simple deterministic square root algorithm as a light weight primality test. If we obtain square roots r, s satisfying (6), we are done, even if $\tilde{R}, \tilde{S}$ are not primes. We give the details in the full version [9]. Similarly, the number of candidates $\tilde{R}, \tilde{S}$ considered can be greatly reduced by sieving to eliminate all candidates divisible by small primes. To summarize, we obtain the following result:

Lemma 5.2. *There is a deterministic algorithm $\mathcal{Q}$ that on input N and $R, S \in (\mathbb{Z}/N\mathbb{Z})^*$ finds a solution to (1). $\mathcal{Q}$'s running time is dominated by the time to generate two primes $\tilde{R} = R \bmod N$ and $\tilde{S} = S' \bmod 4N$ satisfying $(\tilde{S}/\tilde{R}) = 1$.*

We present the running times for various parts of the algorithm in Table 1. As expected, the bottleneck is prime number generation. In Section 5.5 we show that it suffices to generate primes on the order of $\sqrt{N}$ corresponding to the last row in the table. These performance numbers were obtained using the SAGE library.

1024-bit Jacobi symbol:	0.135 ms
full 1024-bit exponentiation:	5.0 ms
1536-bit 3x3 LLL lattice reduction:	6.8 ms
1024-bit prime generation:	123.6 ms
512-bit prime generation:	11.0 ms

Table 1. Running times on 2.015GHz AMD dual-core Athlon64

5.2.1 Solving the system of $\ell + 1$ equations

Recall that in our system the encryptor must solve the $\ell + 1$ equations in (4). Clearly one could apply Lemma 5.2 $(\ell+1)$ times. However, we would like to minimize the number of (expensive) prime generations needed by our system. In particular, we show that a single prime $\tilde{S}$ can be used to

solve all $\ell + 1$ equations. Moreover, we eliminate prime generation altogether during decryption.

As a preliminary optimization, instead of storing u in PP, we let $\tilde{u}$ be the smallest prime such that $\tilde{u} = u \bmod N$ and $\tilde{u} = 3 \bmod 4$, and we store $\tilde{u}$ in PP. Let $\tilde{R}_0 = \tilde{u}$. Below, we provide two methods for generating the other primes needed by the system.

Method 1: The encryptor finds the smallest (probable) prime $\tilde{S}$ such that $\tilde{S} = S' \bmod 4N$ and $(\tilde{S}/\tilde{u}) = 1$. Then for $i = 1, \ldots, \ell$ the encryptor finds the smallest (probable) prime $\tilde{R}_i \in \mathbb{Z}^+$ such that

$$\tilde{R}_i = R_i \bmod N \quad \text{and} \quad \left(\frac{\tilde{S}}{\tilde{R}_i}\right) = 1$$

Note that by quadratic reciprocity we also have $\left(\frac{\tilde{R}_i}{\tilde{S}}\right) = 1$ for $i = 0, \ldots, \ell$. Using the primes $(\tilde{S}, \tilde{R}_i)$ the encryptor can find a solution to $R_i x^2 + S y^2 = 1 \bmod N$ needed for encrypting the i'th bit of the message. Overall, the encryptor needs to generate $(\ell + 1)$ primes.

On the decryption side, for all $i = 1, \ldots, \ell$, the decryptor precomputes and stores the first few primes $\tilde{R}_{i,1}, \tilde{R}_{i,2}, \ldots$ that are congruent to R_i modulo N. Moreover, the encryptor can embed $\tilde{S}$ in the ciphertext and as a result, the decryptor can directly solve the equations $R_i x^2 + S y^2 = 1 \bmod N$ for $i = 0, \ldots, \ell$ without generating any primes. Consequently, decryption in this system is much faster than encryption. Note, however, that the decryptor must precompute and store about $\ell \log_2 \ell$ primes.

Method 2: We reduce the decryptor's storage requirement to ℓ primes, at the expense of slightly increasing the size of PP and the time needed to compute a solution to (5). In particular, we augment PP with values $p_1, \ldots, p_t \in \mathbb{Z}/N\mathbb{Z}$, where each $P_j \leftarrow p_j^2 \bmod N$ is a prime in $\mathbb{Z}$ congruent to 3 modulo 4. We can have $t = O(\log \ell)$. Let $\mathcal{P} = \{P_1, \ldots, P_t\}$ and notice that $\mathcal{P}$ can be generated without knowledge of N's factorization.

The encryptor generates $\tilde{S}$ as before, except that $\tilde{S} = 3 \bmod 4$. For $i = 1, \ldots, \ell$, the encryptor finds the smallest (probable) prime $\tilde{R}_i \in \mathbb{Z}^+$ such that $\tilde{R}_i = R_i \bmod N$ and $\tilde{R}_i = 3 \bmod 4$. For all $i = 0, \ldots, \ell$ there are two cases:

- If $(\tilde{S}/\tilde{R}_i) = -1$, then $(\tilde{R}_i/\tilde{S}) = 1$ by quadratic reciprocity. The encryptor and decryptor deterministically select a prime $P_j \in \mathcal{P}$ such that $(P_j/\tilde{R}_i) = -1$, and set $\tilde{S}' = \tilde{S} \cdot P_j$. In the unlikely event that no such P_j exists, the encryptor looks for the next suitable prime along the arithmetic progression $\tilde{S} = S \bmod N$. Note that $\tilde{S}'$ is a square modulo $\tilde{R}_i$, and vice versa by quadratic reciprocity. Moreover, $\tilde{S}' = 1 \bmod 4$. For these reasons, and since the factors of $\tilde{S}'$ and $\tilde{R}_i$ are known, the encryptor and decryptor can each find a solution to $\tilde{R}_i x^2 + \tilde{S}' y^2 - z^2 = 0$ in $\mathbb{Z}$. This yields a solution $(x/z,\ p_j \cdot y/z)$ to $R_i x^2 + S y^2 = 1$ in $\mathbb{Z}/N\mathbb{Z}$, as needed for encrypting the i'th bit of the message.
- If $(\tilde{S}/\tilde{R}_i) = 1$, then $(\tilde{R}_i/\tilde{S}) = -1$. Encryption and decryption proceed as in the first case, except that we multiply $\tilde{R}_j$ by $P_j \in \mathcal{P}$ to obtain $\tilde{R}' = \tilde{R}_j \cdot P_j$ where $(\tilde{R}'/\tilde{S}) = 1$.

Since $\tilde{S}'$ has about twice as many bits as $\tilde{S}$, the lattice reduction step may take a little longer. However, empirically, the lattice reduction time is dictated by the smaller of the two coefficients (e.g., $|\tilde{R}_i| < |\tilde{S}'|$). In practice, the effect on lattice reduction time is therefore minimal. Since it takes two full $(\log N)$-bit exponentiations to compute $\tilde{R}_i^{1/2} \bmod \tilde{S}'$, this method impacts decryption time somewhat. But it has little impact on encryption time, which is dominated by prime generation.

5.3 A time space tradeoff

So far the encryption bottleneck is in generating the primes $\tilde{S}, \tilde{R}_1, \tilde{R}_2, \ldots$. This work can be greatly reduced by splitting the ℓ-bit plaintext into $\sqrt{\ell}$ blocks each containing $\sqrt{\ell}$ bits. We encrypt each block separately. Write $t \leftarrow \lceil \sqrt{\ell} \rceil$.

For example, the encryptor may instantiate this idea with method 2, using the same values of $\tilde{R}_1, \ldots, \tilde{R}_t$ with each block. For the i'th block, the encryptor generates $\tilde{S}_i$ as in method 2, and encryption proceeds as usual. Overall, the encryptor generates only $2t = 2\lceil \sqrt{\ell} \rceil$ primes, as opposed to $\ell + 1$ primes as discussed in the previous section. The downside is that the ciphertext size increases from a single element in $\mathbb{Z}/N\mathbb{Z}$ to t elements $S_1, \ldots, S_t$ in $\mathbb{Z}/N\mathbb{Z}$.

For concrete parameters, say $\ell = 128$ bits, this approach reduces the number of primes to generate during encryption from $\ell + 1 = 129$ (as in method 2) to $2\lceil \sqrt{\ell} \rceil = 24$, about a factor of 5 improvement. The ciphertext grows from one element in $\mathbb{Z}/N\mathbb{Z}$ to 12 elements $(\tilde{S}_1, \ldots, \tilde{S}_{12})$.

5.4 Hashing IDs to small elements in $\mathbb{Z}/N\mathbb{Z}$

To further speed up prime generation, we can change slightly how the primes $\tilde{R}_1, \ldots, \tilde{R}_\ell$ are generated. Recall that $R_1, \ldots, R_\ell \in \mathbb{Z}/N\mathbb{Z}$ are generated by hashing (ID, i) into $\mathbb{Z}/N\mathbb{Z}$ for $i = 1, \ldots, \ell$. We needed a a uniform distribution over $\mathbb{Z}/N\mathbb{Z}$ for the random oracle simulation. In particular, we needed to generate random pairs $(z, z^2) \in \mathbb{Z}/N\mathbb{Z}$ and then program the oracle so that $H(\mathrm{ID}, i) = z^2$.

Gentry [22], building on work by Vallée [36] and Coron [16], shows how to generate pairs $(z, z^2) \in \mathbb{Z}/N\mathbb{Z}$ where z^2 is indistinguishable from uniform in $QR(N) \cap [1, 8N^{2/3}]$ and z is a uniformly random square root of z^2. Consequently, we can hash (ID, i) into $[1, 8N^{2/3}]$ without hurting the simulation. This can potentially speed up prime

number generation. We note that hashing into a shorter interval can become insecure due to the attacks of Desmedt and Odlyzko [20].

Let us assume that for $i = 1, \ldots, \ell$ the R_i are in the range $[1, 8N^{2/3}]$. We can no longer look for primes $\tilde{R}_i$ along the arithmetic progression $\tilde{R}_i = R_i \bmod N$ since that will negate the benefit of having small R_i. Instead, we generate the $\tilde{R}_i$ by hashing (ID, i, v) for $v = 1, 2, \ldots$. We define $\tilde{R}_i$ as the first prime along this sequence. Since the numbers are a little smaller than before (size $8N^{2/3}$ as opposed to N), finding such a prime is faster than looking along the arithmetic progression.

5.5 Hashing IDs to products of small primes

Another way to speed up prime generation is to compute $\tilde{R}_i$ as the product $\prod_{b=0}^{k-1} H(\mathrm{ID}, i, b)$, where H outputs $\lceil (\log N)/k \rceil$-bit primes, each with Jacobi symbol 1 modulo N. $\tilde{R}_0 = \tilde{u}$ can also be set in this way. Since prime generation takes quartic time (without sieving), generating $\tilde{R}_i$ in this way takes much less time for the sender – by a factor of $O(k^3)$ – than generating $\tilde{R}_i$ as in Section 5.2. The PKG computes the private key value r_i from $\tilde{R}_i \bmod N$ in the usual way.

When $k = 2$, this approach can be simulated very efficiently in the random oracle model. To define $H(\mathrm{ID}, i, b)$, the simulator generates random $q_i \xleftarrow{\mathrm{R}} \mathbb{Z}/N\mathbb{Z}$ and $a_i \xleftarrow{\mathrm{R}} \{0, 1\}$, sets $Q_i \leftarrow u^{a_i} q_i^2 \in \mathbb{Z}/N\mathbb{Z}$ and uses continued fractions to find small numbers $x, y \in \mathbb{Z}/N\mathbb{Z}$ such that $Q_i = x/y$. It repeatedly picks new random (q_i, a_i) until x and y are $\lceil (\log N)/2 \rceil$-bit primes coming from the appropriate distribution. Next, it sets $r_i \leftarrow q_i \cdot y$ and $\tilde{R}_i \leftarrow Q_i \cdot y^2 \ (= x \cdot y)$. When queried, it returns $H(\mathrm{ID}, i, 0) \leftarrow x$, $H(\mathrm{ID}, i, 1) \leftarrow y$, and r_i as the private key component for $\tilde{R}_i$.

Moreover, when $k = 2$, we can construct a variant of our system that uses the faster approach to prime generation, without making the other steps (notably, the lattice reduction) much slower. In this variant, the PKG augments PP as in Method 3 in Section 5.2.1, except that each $P_j \in \mathcal{P}$ is a product of two $\lceil (\log N)/2 \rceil$-bit primes, each congruent to 3 modulo 4; let $P_{j,0}$ and $P_{j,1}$ denote the two factors of P_j. To instantiate H efficiently, $H(\mathrm{ID}, i, 0)$ is computed (using sieving) as the first prime following $H'(\mathrm{ID}, i, 0)$ that is congruent to 1 modulo 4, where H' outputs $\lceil (\log N)/2 \rceil$-bit integers; $H(\mathrm{ID}, i, 1)$ is computed identically, except it equals 3 modulo 4. To compute $\tilde{R}_i$, the encryptor sets $\tilde{R}_{i,0} \leftarrow H(\mathrm{ID}, i, 0)$, $\tilde{R}_{i,1} \leftarrow H(\mathrm{ID}, i, 1)$, and $\tilde{R}_i \leftarrow \tilde{R}_{i,0} \cdot \tilde{R}_{i,1}$. The encryptor generates $\tilde{S}$ in the usual way. (This single large prime generation seems unavoidable.) For $i = 0, \ldots, \ell$, if $(\tilde{R}_{i,0}/\tilde{S}) = (\tilde{R}_{i,1}/\tilde{S}) = 1$, the encryptor and decryptor compute a solution to $\tilde{R}_i x^2 + \tilde{S} y^2 - z^2 = 0$ in the usual way. If $(\tilde{R}_{i,0}/\tilde{S}) = -1$ and $(\tilde{R}_{i,1}/\tilde{S}) = 1$, they select $P_j, P_k \in \mathcal{P}$ such that $\tilde{R}_{i,0} \cdot P_{j,0}$, $\tilde{R}_{i,1}$ and $P_{j,1}$ are all squares modulo $\tilde{S}' \leftarrow \tilde{S} \cdot P_k$, and vice versa. Then, they solve the three equations $Tx^2 + \tilde{S}' y^2 - z^2 = 0$, where T is one of $\tilde{R}_{i,0} \cdot P_{j,0}$, $\tilde{R}_{i,1}$, or $P_{j,1}$. Afterwards, these three solutions can be combined to obtain a solution to $\tilde{R}'_i x^2 + \tilde{S}' y^2 = 1 \bmod N$ for $\tilde{R}'_i \leftarrow \tilde{R}_i \cdot P_j$ by using the product formula (Lemma 5.1) twice. Multiplying x by p_j and y by p_k gives a solution to $R_i x^2 + S y^2 = 1 \bmod N$, after which encryption and decryption proceed in the usual way. The remaining cases for $(\tilde{R}_{i,0}/\tilde{S})$ and $(\tilde{R}_{i,1}/\tilde{S})$ are similar.

Empirically, the time needed to reduce the lattice associated to the equation $Tx^2 + \tilde{S}' y^2 - z^2 = 0$ is dictated by the smaller of the two values T and $\tilde{S}'$. Since lattice reduction takes cubic time, and since T is $(\log N)$ bits once, and $(\log N)/2$ bits twice, in the three equations above, the three lattice reductions take only about 1.25 times as long as one "normal" lattice reduction in our system. So, aside from the generation of $\tilde{S}$, this approach allows significantly (8 times) faster prime generation almost for free.

Setting $k > 2$ is an interesting possibility, but it is unclear how to make the simulation efficient. Of course, the simulator could generate $\tilde{R}_i$ as follows: generate random $r_i \xleftarrow{\mathrm{R}} \mathbb{Z}/N\mathbb{Z}$ and $a_i \xleftarrow{\mathrm{R}} \{0, 1\}$, set $\tilde{R}_i \leftarrow u^{a_i} r_i^2$, and try to factor $\tilde{R}_i$, hoping that it is (roughly speaking) $N^{1/k}$-smooth. But this leads to an extremely loose reduction.

6 Anonymous IBE and security without random oracles

The product formula (Lemma 5.1) allows us to quickly compute $\mathcal{Q}(N, uR_j, S)$ from $\mathcal{Q}(N, R_j, S)$. We show that for these derived solutions the decryptor can deduce the ciphertext bit $\bar{c}_j$ from the bit c_j. As a result, the bits $(\bar{c}_1, \ldots, \bar{c}_\ell)$ need not be included in the ciphertext. By eliminating the extra ciphertext bits we obtain an anonymous IBE and we can prove its security based on the IQR assumption *without* the random oracle model.

As in Section 3 we begin by stating the abstract properties needed for this construction. We then give an example instantiation and describe the resulting IBE system.

Definition 6.1. Let $\mathcal{Q}'$ be a deterministic algorithm that takes as input (N, u, R, S) where $N \in \mathbb{Z}^+$ and $u, R, S \in \mathbb{Z}/N\mathbb{Z}$. The algorithm outputs polynomials $f, \bar{f}, g, \tau \in \mathbb{Z}/N\mathbb{Z}[x]$. We say that $\mathcal{Q}'$ is **Enhanced IBE Compatible** if the following conditions hold:

- (Condition 1a) If R and S are quadratic residues, then $f(r)g(s)$ is also a quadratic residue for all square roots r of R and s of S.

- (Condition 1b) If uR and S are quadratic residues, then $\bar{f}(\bar{r})g(s)\tau(s)$ is also a quadratic residue for all square roots $\bar{r}$ of uR and s of S.
- (Condition 2a) If R is a quadratic residue, then $f(r)f(-r)S$ is also a quadratic residue for all square roots r of R.
- (Condition 2b) If uR is a quadratic residue, then $\bar{f}(\bar{r})\bar{f}(-\bar{r})S$ is also a quadratic residue for all square roots $\bar{r}$ of uR.
- (Condition 2c) If S is a quadratic residue, then $\tau(s)\tau(-s)u$ is also a quadratic residue for all square roots s of S.
- (Condition 3) τ is independent of R, that is, $\mathcal{Q}'(N, u, R_1, S)$ and $\mathcal{Q}'(N, u, R_2, S)$ produce the same τ for all N, u, R_1, R_2, S.

6.1 An enhanced IBE compatible example

As an instance of such a $\mathcal{Q}'$, we proceed from Section 5.1 as follows. Algorithm $\mathcal{Q}'(N, u, R, S)$:

1. Construct a solution $(x, y) \in (\mathbb{Z}/N\mathbb{Z})^2$ to $Rx^2 + Sy^2 = 1$
2. Construct a solution $(\alpha, \beta) \in (\mathbb{Z}/N\mathbb{Z})^2$ t $u \cdot \alpha^2 + S \cdot \beta^2 = 1$
3. Output the polynomials:
   $f(r) \leftarrow xr + 1$ $\quad \bar{f}(\bar{r}) \leftarrow 1 + Sy\beta + \alpha x\bar{r}$
   $g(s) \leftarrow 2ys + 2$ $\quad \tau(s) \leftarrow 1 + \beta s.$

This satisfies Conditions 1a, 2a and 2c as in Section 4, and trivially satisfies Condition 3. As for Conditions 1b and 2b, we note that, per Lemma 5.1, $\bar{x} \leftarrow \alpha x/(Sy\beta + 1)$ and $\bar{y} \leftarrow (y + \beta)/(Sy\beta + 1)$ satisfy $uR \cdot \bar{x}^2 + S \cdot \bar{y}^2 = 1$, so that when uR and S are quadratic residues:

$$\begin{aligned}
&\bar{f}(\bar{r})g(s)\tau(s) \\
&= (1 + Sy\beta + \alpha x\bar{r}) \cdot (2 + 2ys) \cdot (1 + \beta s) \\
&= 2 \cdot (1 + Sy\beta + \alpha x\bar{r}) \cdot (1 + Sy\beta + ys + \beta s) \\
&= 2 \cdot \left(1 + \bar{r}\tfrac{\alpha x}{Sy\beta+1}\right) \cdot \left(1 + s\tfrac{y+\beta}{Sy\beta+1}\right) \cdot (Sy\beta + 1)^2 \\
&= 2 \cdot (1 + \bar{r}\bar{x}) \cdot (1 + s\bar{y}) \cdot (Sy\beta + 1)^2
\end{aligned}$$

This is a quadratic residue: $2\cdot(1+\bar{r}\bar{x})\cdot(1+s\bar{y})$ is a quadratic residue as in Section 4, and so is $(Sy\beta + 1)^2$ by definition. Similarly, if uR is a quadratic residue, then

$$\bar{f}(\bar{r})\bar{f}(-\bar{r})S = (Sy\beta + 1)^2(1 + \bar{r}\bar{x})(1 - \bar{r}\bar{x})S$$

is a quadratic residue as in Section 4.

6.2 The modified IBE system

With an Enhanced IBE Compatible $\mathcal{Q}'$ we construct an IBE, which we call AnonIBE. Its *Setup* and *KeyGen* algorithms are the same as in the simpler system of Section 3.1.

Encrypt(PP, ID, m): To encrypt a message $m = m_1, m_2, \ldots, m_\ell \in \{\pm 1\}^\ell$ to some ID, pick a random $s \in \mathbb{Z}/N\mathbb{Z}$ and compute $S \leftarrow s^2 \bmod N$. Run $\mathcal{Q}'(N, u, 1, S)$ to obtain the polynomial τ, and compute $k \leftarrow \left(\frac{\tau(s)}{N}\right)$. Then for $j \in [1, \ell]$, set $R_j \leftarrow H(\text{ID}, j)$, run $\mathcal{Q}'(N, u, R_j, S)$ to obtain g_j, and compute $c_j \leftarrow m_j \cdot \left(\frac{g_j(s)}{N}\right)$. Set $c = c_1, \ldots, c_\ell$; the ciphertext is then (S, k, c). The ciphertext size is now $(\log_2 N + \ell + 1)$ bits which is $\ell - 1$ bits shorter than before.

Decrypt(C, d_{ID}): To decrypt a ciphertext $C = (S, k, c)$ with a private key $d_{\text{ID}} = (r_1, \ldots, r_\ell)$, for $j = 1, \ldots, \ell$ the receiver sets $R_j \leftarrow H(\text{ID}, j)$ and runs $\mathcal{Q}'(N, u, R_j, S)$ to obtain f_j and $\bar{f}_j$.

$$\text{if} \quad r_j^2 = R_j \quad \text{set} \; m_j \leftarrow c_j \cdot \left(\frac{f_j(r_j)}{N}\right)$$
$$\text{if} \quad r_j^2 = uR_j \quad \text{set} \; m_j \leftarrow c_j \cdot k \cdot \left(\frac{\bar{f}_j(r_j)}{N}\right)$$

Output $m \leftarrow m_1 \ldots m_\ell$. Conditions 1a, 1b and 3 above guarantee that this correctly recovers the message.

Security We prove that the system is ANON-IND-ID-CPA based on the IQR assumption.

Theorem 6.2. *Let $\mathcal{A}$ be an efficient ANON-IND-ID-CPA adversary against AnonIBE. Then there exist efficient algorithms $\mathcal{B}_1, \mathcal{B}_2$, running in about the same time as $\mathcal{A}$, such that*

$$\begin{aligned}
\text{IBEAdv}_{\mathcal{A},\text{AnonIBE}}(\lambda) &\leq \text{PRFAdv}_{\mathcal{B}_1,F}(\lambda) \\
&+ \text{IQRAdv}_{\mathcal{B}_2,(RSAgen,H)}(\lambda)
\end{aligned}$$

The proof is given in the full version [9]. Note that the IQR assumption follows from the QR assumption where H (i.e., H_N) is a full-domain hash function modeled as a random oracle. The challenger picks some $v \in J(n)$, and can replace calls to $H_N(X)$ by choosing $x \xleftarrow{R} \mathbb{Z}/N\mathbb{Z}, a \xleftarrow{R} \{0, 1\}$, and returning $H_N(X) = v^{-a}x^2$. When queried for $\mathcal{O}(N, X)$, it returns x. If v is not a quadratic residue, this gives a properly random distribution of H_N. If v is a quadratic residue, the results are still indistinguishable by the QR assumption. In summary, AnonIBE is secure in the standard model under the IQR assumption, and in the random oracle model under the QR assumption.

7 Extensions and Open Problems

7.1 Chosen ciphertext security

BasicIBE can be converted into an IND-ID-CCA system FullIBE in the random oracle model using generic techniques, such as Construction 3 in [3].

More importantly, in the full version [9] we build an ANON-IND-ID-CCA IBE system in the standard model, under the IQR assumption. First we describe a hash proof system for quadratic residuosity (as defined in [17]) that employs ideas described earlier in this paper. We then construct an ANON-IND-ID-CCA IBE system based on IQR. We also briefly describe an anonymous PKE system based on our hash proof system that is anonymous in the multi-user setting and IND-CCA secure in the standard model under the (*non-interactive*) QR assumption.

7.2 Multivariate quadratics and higher residue symbols

A possible way to extend our IBE system is to use a higher-order character in place of the Jacobi symbol. This may improve efficiency by allowing the sender and receiver to derive more than one bit of shared secret data with each computation. Another possible extension is to use multivariate polynomials f and g instead of univariate ones. This may improve efficiency if it allows us to replace $\mathcal{Q}$ by a faster algorithm.

8 Conclusions

We described a space-efficient anonymous IBE system without pairings based on the quadratic residuosity assumption in the random oracle model. In the standard model, the system is secure under the interactive quadratic residuosity assumption. Our system is more space efficient than Cocks' IBE, but is less time efficient. Since the bottleneck in our system is due to algorithm $\mathcal{Q}$ it is natural to look for other instantiations of our abstract system that are more efficient. Another natural important problem is to design an IBE system which is secure based on quadratic residuosity without random oracles. Currently, both our system and Cocks' need the random oracle model to argue that the underlying Rabin signature system is existentially unforgeable.

Acknowledgments

We thank Peter Montgomery for his comments on extensions to Section 4.

References

[1] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. In *CRYPTO*, pages 205–222, 2005.

[2] D. Balfanz, G. Durfee, N. Shankar, D. Smetters, J. Staddon, and H.-C. Wong. Secret handshakes from pairing-based key agreements. In *Proceedings of the IEEE Symposium and Security and Privacy*, pages 180–196, 2003.

[3] K. Bentahar, P. Farshim, J. Malone-Lee, and N. Smart. Generic constructions of identity-based and certificateless kems. `http://eprint.iacr.org/2005/058`.

[4] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of 2007 IEEE Symposium on Security and Privacy*, 2007.

[5] D. Boneh and X. Boyen. Efficient selective-ID identity based encryption without random oracles. In *Proceedings of Eurocrypt 2004*, LNCS, pages 223–238. Springer-Verlag, 2004.

[6] D. Boneh, R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. of Computing (SICOMP)*, 36(5):915–942, 2006.

[7] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *Proceedings of Eurocrypt '04*, 2004.

[8] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM Journal of Computing*, 32(3):586–615, 2003. extended abstract in Crypto '01.

[9] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. ePrint report 2007/177, 2007. `http://eprint.iacr.org/2007/177`.

[10] X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Crypto '06*, 2006.

[11] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In *Proceedings of Eurocrypt 2003*, volume 2656 of *LNCS*. Springer, 2003.

[12] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Proceedings of Eurocrypt 2004*, LNCS, 2004.

[13] J. W. S. Cassels. *Rational quadratic forms*, volume 13 of *London Mathematical Society Monographs*. Academic Press, 1978.

[14] T. Cochrane and P. Woods. Small solutions of the legendre equation. *Journal of Number Theory*, 70(1):62–66, 1998.

[15] C. Cocks. An identity based encryption scheme based on quadratic residues. In *Proceedings of the 8th IMA International Conference on Cryptography and Coding*, pages 26–8, 2001.

[16] J.-S. Coron. Security proof for partial-domain hash signature schemes. In *Proceedings of Crypto '02*, LNCS, pages 613–626. Springer, 2002.

[17] R. Cramer and V. Shoup. Universal hash proofs and a paradigm for chosen ciphertext secure public key encryption. In *Proceedings of Eurocrypt 2002*, 2002.

[18] J. Cremona and D. Rusin. Efficient solution of rational conics. *Math. of computation*, 72(243):1417–1441, 2003.

[19] G. D. Crescenzo and V. Saraswat. Searchable public-key encryption based on jacobi symbols. manuscript, Feb. 2007.

[20] Y. Desmedt and A. Oldyzko. A chosen text attack on the rsa cryptosystem and some discrete logarithm schemes. In *Proceedings of Crypto '85*, volume 218 of *LNCS*, pages 516–521. Springer-Verlag, 1985.

[21] Y. Dodis and N. Fazio. Public key broadcast encryption for stateless receivers. In *Proceedings of the Digital Rights Management Workshop 2002*, volume 2696 of *LNCS*, pages 61–80. Springer, 2002.

[22] C. Gentry. How to compress rabin ciphertexts and signatures (and more). In *Proceedings of Crypto '04*, pages 179–200, 2004.

[23] C. Gentry. Practical identity-based encryption without random oracles. In *Proceedings of Eurocrypt '06*, volume 4004 of *LNCS*, pages 445–464, 2006.

[24] C. Gentry and A. Silverberg. Hierarchical ID-based cryptography. In *Proceedings of Asiacrypt 2002*, volume 2501 of *LNCS*, pages 548–66, 2002.

[25] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*. Springer, 1999.

[26] L. L. H. Lensta, L. Lensta. Factoring polynomials with rational coefficients. *Math. Ann.*, 261:515–534, 1982.

[27] J. Horwitz and B. Lynn. Towards hierarchical identity-based encryption. In *Proceedings of Eurocrypt 2002*, volume 2332 of *LNCS*, pages 466–81. Springer, 2002.

[28] N. Li, W. Du, and D. Boneh. Oblivious signature-based envelope. In *In proceedings of PODC '03*, pages 182–189, 2003.

[29] K. Mccusker, N.O'Connor, and D. Diamond. Low-energy finite field arithmetic primitives for implementing security in wireless sensor networks. In *Proceedings of 2006 conference on Communications, Circuits and Systems*, 2006.

[30] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1997.

[31] H. Ong, C. P. Schnorr, and A. Shamir. An efficient signature scheme based on quadratic equations. In *Proceedings of STOC '84*, pages 208–216, 1984.

[32] J. M. Pollard and C. P. Schnorr. An efficient solution of the congruence $x^2 + ky^2 = m(modn)$. *IEEE Transactions on Information Theory*, 33(5):702–709, 1987.

[33] R. Sakai and M. Kasahara. ID based cryptosystems with pairing over elliptic curve. `http://eprint.iacr.org/2003/054`, 2003.

[34] A. Shamir. Identity-based cryptosystems and signature schemes. In *Advances in Cryptology—CRYPTO 1984*, volume 196 of *LNCS*, pages 47–53. Springer-Verlag, 1984.

[35] E. Shi, J. Bethencourt, T.-H. H. Chan, D. Song, and A. Perrig. Multi-dimensional range query over encrypted data. In *Proceedings of the IEEE Symposium and Security and Privacy*, 2007.

[36] B. Vallee. Generation of elements with small modular squares and provably fast integer factoring algorithms. *Mathematics of Computation*, 56(194):823–849, 1991.

[37] B. Waters. Efficient identity-based encryption without random oracles. In *Proceedings of Eurocrypt 2005*, LNCS, 2005.

[38] B. Waters, D. Balfanz, G. Durfee, and D. Smetters. Building an encrypted and searchable audit log. In *Proceedings of Network and Distributed System Security Symposium (NDSS 2004)*, 2004.

[39] D. Yao, N. Fazio, Y. Dodis, and A. Lysyanskaya. ID-based encryption for complex hierarchies with applications to forward security and broadcast encryption. In *Proceedings of the ACM Conference on Computer and Communications Security 2004*, pages 354–63, 2004.

A Solving $R \cdot x^2 + S \cdot y^2 = z^2$ in $\mathbb{Z}$

In section 5.2, we needed an algorithm to solve $R \cdot x^2 + S \cdot y^2 = z^2$. Here we describe such an algorithm. We are given integers R, S, r, s where

- $\gcd(R, S) = 1$, R is odd, and $S = 1 \bmod 4$,
- $r^2 = R \bmod S$ and $s^2 = S \bmod R$.

We show how to construct a solution $(x, y, z) \in \mathbb{Z}^3$ to

$$R \cdot x^2 + S \cdot y^2 - z^2 = 0 \tag{7}$$

using an algorithm described in [18]. The algorithm presented here is based on the proof of Legendre's theorem due to Mordell. Consider the three dimensional lattice L in $\mathbb{Z}^3$ containing all triples (x, y, z) satisfying

$$\begin{aligned} s \cdot y &= z \pmod R \\ r \cdot x &= z \pmod S \\ x &= 0 \pmod 2 \\ y &= z \pmod 2 \end{aligned}$$

It is easy to see that any point (x, y, z) in L satisfies

$$Rx^2 + Sy^2 - z^2 = 0 \pmod{4RS}$$

We define the norm $q(x, y, z) = |R|x^2 + |S|y^2 + z^2$ and look for the shortest vector $v \in L$ under the norm q. A simple determinant calculation (and an application Minkowski's theorem) shows that a vector $v = (x_0, y_0, z_0) \in L$ exists whose q-norm is less than $4RS$. Then $|Rx_0^2 + Sy_0^2 - z_0^2| < |4RS|$. But since $Rx_0^2 + Sy_0^2 - z_0^2 = 0 \bmod 4RS$, it follows that $Rx_0^2 + Sy_0^2 - z_0^2 = 0$ holds over the integers, as required.

To find the shortest vector $v \in L$ Cremona and Rusin [18] use the LLL algorithm [26]. They show that in three dimensions, one can easily derive the shortest vector from an LLL reduced basis. Cremona and Rusin present other algorithms for solving (7) that do not use LLL and are claimed to be twice as fast as the algorithm above. We also note that Cochrane and Mitchell [14] show that the lattice L without the 2-adic components already contains a solution to (7).