Revocation Systems with Very Small Private Keys

Allison B. Lewko [*]
abishop@math.utexas.edu

Amit Sahai[†]
sahai@cs.ucla.edu

Brent Waters [‡]
bwaters@cs.utexas.edu

Abstract

In this work, we design a method for creating public key broadcast encryption systems. Our main technical innovation is based on a new "two equation" technique for revoking users. This technique results in two key contributions:

First, our new scheme has ciphertext size overhead $O(r)$, where r is the number of revoked users, and the size of public and private keys is only a *constant* number of group elements from an elliptic-curve group of prime order. In addition, the public key allows us to encrypt to an unbounded number of users. Our system is the first to achieve such parameters. We give two versions of our scheme: a simpler version which we prove to be secure in the standard model under a new, but non-interactive assumption, and another version that employs the new dual system encryption technique of Waters to obtain security under the d-BDH and decisional Linear assumptions.

Second, we show that our techniques can be used to realize Attribute-Based Encryption (ABE) systems with non-monotonic access formulas, where our key storage is significantly more efficient than previous solutions. This result is also proven in the standard model under our new non-interactive assumption.

We believe that our new technique will be of use elsewhere as well.

1 Introduction

In a broadcast encryption system [20], a broadcaster encrypts a message such that a particular set S of devices can decrypt the message sent over a broadcast channel. Broadcast systems have a wide range of applications including file systems, group communication, DVD content distribution, and satellite subscription services. In many of these applications, the notion of revocation is important. For example, if a DVD-player's key material is leaked on the Internet, one might want to revoke it from decrypting future disks. In another example, consider a group of nodes communicating sensitive control and sensor information over a wireless network; if any of these nodes becomes compromised, we'd like to revoke them from all future broadcasts.

In this work, we design new broadcast encryption schemes, and we focus on two important contributions.

Revocation Systems with Small Key Sizes. We create public key revocation encryption systems with small cryptographic private and public keys. Our systems have two important features relating respectively to public and private key size.

[*]Supported by the National Defense Science and Engineering Graduate Fellowship.

[†]This research was supported in part by an Alfred P. Sloan Foundation Research Fellowship, an Intel equipment grant, a Cyber-TA Army grant, and NSF ITR/Cybertrust grants 0205594, 0456717 and 0627781.

[‡]Supported by NSF CNS-0749931, CNS-0524252, CNS-0716199; the U.S. Army Research Office under the CyberTA Grant No. W911NF-06-1-0316; and the U.S. Department of Homeland Security under Grant Award Number 2006-CS-001-000001.

First, public keys in our two systems are short (just 5 group elements and 12 group elements respectively) and enable a user to create a ciphertext that revokes an *unbounded* number of users. This is in contrast to other systems [10, 33, 18] where the public parameters bound the number of users in the system and must be updated to allow more users.

Second, the cryptographic key material that must be stored securely on the receiving devices is small. Keeping the size of private key storage as low as possible is important as cryptographic keys will often be stored in tamper-resistant memory, which is more costly. This can be especially critical in small devices such as sensor nodes, where maintaining low device cost is particularly crucial. Device keys in our systems are only a small *constant* number of group elements (in fact, just 3 group elements and 5 group elements respectively) from an elliptic-curve group of prime order. Furthermore, our schemes are public-key stateless broadcast encryption schemes[1], and we work with stateless receivers.

We achieve this small device key size without compromising on other critical parameters such as ciphertext length – our ciphertexts will consist of just $O(r)$ group elements, where r is the number of revoked users. This is the same behavior as the previously best-known schemes for revocation. We also do not compromise on security: we obtain our results in the standard model under the well-established d-BDH and decisional Linear assumptions.

Attribute-Based Encryption with Non-Monotonic Formulas. Our second key contribution is that we show how our techniques can be applied to achieving efficient Attribute-Based Encryption (ABE) [36] schemes with *non-monotonic* access formulas. Ostrovsky, Sahai, and Waters [34] showed a connection between revocation schemes and achieving non-monotonic access formulas in ABE; to negate an attribute in an access formula one applies a revocation scheme using the attribute as an identity to be revoked. Ostrovsky, Sahai, and Waters give a particular instance by adapting the revocation scheme of Naor and Pinkas [33] to the ABE scheme of Goyal et. al [24]. The primary drawback of their scheme is that the private key size of their scheme blows up by a *multiplicative* factor of $\log n$, where n is the maximum number of attributes. More precisely, once the DeMorgan's law transformation is made, each negated attribute in the private key will have $O(\log n)$ group elements. By adapting our new revocation techniques to the Goyal et. al ABE scheme, we get that each negated attribute will only take two group elements. In practice, for many applications the private key storage will decrease by an order of magnitude.

Our Techniques. The primary challenge in constructing broadcast encryption schemes is to achieve full collusion resilience – to make sure that if all the revoked users combine their key material, they still cannot decrypt ciphertexts.

In order to understand our techniques it is useful to review the Naor-Pinkas [33] revocation scheme. In their system in order to revoke r users[2] a degree r polynomial $q(x)$ is chosen and $O(r)$ group elements are published allowing anyone to compute $g^{q(x)}$ for generator g in group $\mathbb{G}$ of order p. A private key for user i consists of $q(i)$. To encrypt, a user selects a revoked set of users S and a secret exponent $s \in \mathbb{Z}_P$. The ciphertext consists of g^s along with $g^{sq(j)}$ for each revoked user j in the set S. If an attacker consists of just users from the set S, he will be unable to produce any new points of the polynomial $s \cdot q(x)$. From a high level view, this system revokes by giving revoked users *redundant* information. The system provides collusion resistance by defining a "global" polynomial across the whole system. Unfortunately,

[1]And in fact, our schemes are identity-based: each device's private key can be based on the device's natural "identity," which could be an arbitrary string like a serial number or even an email address. In most previous schemes, every device had to be assigned a specific number between 1 and n.

[2]To revoke less than r users, they simply revoke some "dummy" users.

this structure inherently locks the system to a predetermined maximum number of revoked users and a long public key.

In order to avoid these limitations, we propose a new methodology for building revocation systems. Like the Naor-Pinkas system, we use the idea of revocation by redundant equations. However, instead of using a system that defines a global polynomial, we let the encryption algorithm define several "local" revocation equations. Our techniques have two major components:

First, we use a "two equation" method for decryption. A ciphertext will be encrypted such that a certain set $S = \{\mathrm{ID}_1, \ldots, \mathrm{ID}_r\}$ will be revoked from decrypting it. Since the ciphertext consists of $O(r)$ group elements, there will be a ciphertext component for each ID_i. Intuitively, when decrypting, a user ID will apply his secret key to each component. If $\mathrm{ID} \neq \mathrm{ID}_i$, he will get two *independent* equations and be able to extract the ith decryption share. However, if $\mathrm{ID} = \mathrm{ID}_i$ (*i.e.* he is revoked), then he will only get two *dependent* equations of a two variable formula and thus be unable to extract the decryption share. Alternatively, we can view each ciphertext component as *locally* defining a different degree one polynomial. For component i, a user ID will get two points on a fresh degree one polynomial $q_i(x)$ iff $\mathrm{ID} \neq \mathrm{ID}_i$ (and otherwise the user will essentially only get one point on the polynomial, which is not enough to solve). We can view this as a local revocation of each user to a component of the ciphertext.

One large challenge of our "local" revocation approach is that we need to make sure that multiple users cannot collude to decrypt the message. For example, if there is a ciphertext that revokes $S = \{\mathrm{ID}_1, \mathrm{ID}_2\}$, these users might try to decrypt by letting user ID_2 get the first share and user ID_1 obtain the second share. To prevent this attack, our key shares are randomized or "personalized" to each user to prevent combination of decryption shares. To achieve this, we devise a new technique for achieving collusion resilience using novel cancellation techniques based on the power of a bilinear map.

Our first (simpler) system clearly demonstrates our techniques and is shown to be secure under a new non-interactive assumption that we call the decisional q-Multi Exponent Bilinear Diffie-Hellman (q-MEBDH) assumption. We show the assumption to hold in the generic bilinear group model in Appendix A.1[3]. We prove security in the standard model, showing that a ciphertext that revokes up to r users is secure if the decisional r-MEBDH assumption holds.

Our second system combines the techniques of our first system with the recent dual system encryption technique of Waters [44]. This technique was used to give a fully secure IBE system under the d-BDH and decisional Linear Assumptions which we will adapt to form our revocation system. In a dual system, keys and ciphertext can take on two forms: they can either be normal (as used in the real system) or semi-functional. (Semi-functional keys and ciphertexts are not used in the real system, they are only used in proving its security.) When a normal key is used to decrypt a semi-functional ciphertext or a semi-functional key is used to decrypt a normal ciphertext, decryption will still work. When a semi-functional key is used to decrypt a semi-functional ciphertext, decryption will fail. Security for dual systems is proved using a sequence of indistinguishable games. In the first game, all keys and ciphertexts are normal as in the real system. Next, the ciphertext is changed to be semi-functional. Then, the keys are changed to be semi-functional one by one. Once all the keys given to the attacker are semi-functional, none are useful for decrypting the challenge ciphertext, so proving security becomes much easier.

In the intermediate games where the keys switch to semi-functional, the simulator is prepared to create a semi-functional key for any identity and a challenge ciphertext for any allowed subset of revoked identities. This may seem problematic, since the simulator might try to test semi-functionality of the key in question for itself by creating a semi-functional challenge ciphertext

[3]One might wonder if the security proof of our assumption in the generic group model suggests the need for much larger security parameters, thereby negating the efficiency advantages claimed here; indeed we show that this is *not* the case. See Section A.3 and Appendix A.1 for more details.

where that user is not revoked. We will avoid this issue by making sure the simulator can only form the semi-functional ciphertext properly when the key in question is a revoked user. This is similar to the technique used in the Broadcast Encryption scheme in [44], but this system had key sizes which were linear in the number of users while our system achieves constant key sizes.

We prove our system to be secure in the standard model under the well-established d-BDH and decisional Linear assumptions. The clear advantage of this system is its reliance on simpler, more standard assumptions. Its only (relatively) disadvantage is that the constant public and private key sizes are slightly higher than in our first system.

We believe that our technique will be of use in other cryptographic applications, as well. Recently, Waters [44] applied the revocation techniques of a prior version of this paper to construct new fully secure HIBE schemes based on simple assumptions, and fully secure IBE schemes with very short public parameters.

1.1 Related Work

Fiat and Naor [20] first introduced the problem of broadcast encryption. In their system they proposed a scheme that is secure against a collusion of t users, where the ciphertext size was $O(t \log^2 t \log n)$. This system and other following work [40, 41, 42, 29, 21, 22], used a combinatorial approach. For this type of approach, there is an inherent tradeoff between the efficiency of the system and the number, t, of colluders that the system is resistant to. An attacker in the system that compromises more than t users can compromise the security of the scheme.

For systems without a bound on the number of revoked users at setup, there have been two general classes of revocation broadcast schemes. The first stateless tree-based revocation schemes were proposed by Naor, Naor and Lopspeich [32] where they introduced the "subset cover" framework. In their framework users were assigned to leaves in a tree and belonged to different subsets. An encryptor encrypts to the minimum number of subsets that covers all the non-revoked users and none of the revoked ones. The primary challenge is to structure the subsets so that they are expressive enough to allow for small ciphertext overhead, yet don't impose large private key overhead on the user. The NNL paper proposed two systems with ciphertext sizes of $O(r \lg n)$ and $O(2r)$ and private key sizes of $O(\lg n)$ and $O(\lg^2 n)$ respectively. These methods were subsequently improved upon in future works by Halvey and Shamir [26] and by Goodrich, Sun, and Tamassia [23], where the GST system gives $O(r)$ size ciphertexts and $O(\lg n)$ size private keys. Dodis and Fazio [19] show how to make the the NNL and Halevy and Shamir systems public key by employing hierarchical identity-based encryption methods. It is unknown how to realize the more efficient GST scheme in the public key setting.

The second class of methods is based on polynomial interpolation in the exponents of group elements and was given by Kurosawa and Desmedt [30] and Naor and Pinkas [33]. In these systems the setup algorithm picks a polynomial of degree d, where d is the maximum number of users that can be revoked. Both the public key and ciphertexts are of size d. Yoo et. al. [46] observe that $\lg(n)$ parallel systems can be used to handle n users with $O(r)$ size private keys, $O(n)$ size public keys and $O(r)$ size ciphertexts.

We note that there are a class of *stateful* encryption schemes known as logical-tree-hierarchy schemes independently discovered by Wallner et al. [43] and Wong [45], which are improved in further work [13, 16, 38]. The drawback of stateful schemes is that if a receiver misses an update it won't be able to decrypt future messages (or this must be corrected somehow). Even so, our stateless solution actually provides a more efficient way to revoke users in the stateful setting than previous schemes.

We remark that two equation techniques are somewhat reminiscent of of those used for knowledge extraction in discrete log proof of knowledge settings [37]. In addition, different types of two equation techniques have been applied in ecash applications (see e.g., [12] and the references therein).

We also note that [10] proposed the first non-trivial *fully collusion resistant* broadcast encryption scheme; broadcasts to a set of uncompromised users remain secure no matter how many other keys the adversary obtained. (In contrast, our approach and those referenced above would lead to very long ciphertexts if the number of revoked users were very large.) Their scheme allows for broadcasts to an arbitrary set of users where the ciphertexts and private key material are both a constant number of group elements, however, the public key material is linear in the number of users in the system and, moreover, the public key must be accessible by any decryptor in the system. This makes their solution unusable for small devices that cannot store the public key. In comparison, our solution is appropriate for applications, like group encryption, where we expect relatively few devices will be compromised and revoked from the encryption and where we need very small storage.

Finally, Delerablée, Paillier and Pointcheval [18] use a type of inversion technique to achieve a system with small private keys, but public parameters still require a linear number of group elements in the number of users.[4] Unlike our system, the published public parameters will establish an upper bound on the number of users that may be encrypted to (without "appending" to the public key), although private keys need not be modified.

Attribute-Based Encryption was introduced by Sahai and Waters [36]; subsequent works [24, 6, 17, 34, 25] have proposed ABE systems with different properties. Different authors [39, 31, 3, 11, 1, 4] have considered similar problems without considering collusion resistance.

Key Sizes. We stress that, as summarized above, all previous public key and identity-based revocation schemes required[5] either (1) larger private key size by at least a factor of $\log n$, where n is the number of users, or (2) much larger public parameter size, by a factor of n.

1.2 Organization

The rest of the paper is organized as follows. In Section 2 we provide the relevant definitions for revocation systems and background information on groups with efficiently computable bilinear maps. We then give the construction of our simple revocation system in Section 3 and our second system in Section 4. We prove security of our system in Section 5. Finally, we show how to realize a non-monotonic Attribute-Based Encryption system with small private key sizes in Section 6.

2 Background

We begin by providing a security definition for a revocation system, in the identity-based framework. We use definitions that are similar, for example, to the definitions for broadcast encryption used by Boneh, Gentry, and Waters [10]; however we adapt our definition to the Identity-Based setting. Later, we state our complexity assumptions.

[4]The authors additionally describe a secret key version of the scheme where the broadcaster is the same as the authority. In this case the trusted broadcaster can contain a short secret for encryption (e.g., the seed used for system setup).

[5]We also stress that this is an "apples to apples" comparison, since in all these *public-key* schemes, the underlying group size would be comparable for a given security level (or favor our setting of elliptic curve groups).

2.1 Identity-Based Revocation Systems

An encryption system is made up of three randomized algorithms: For simplicity of notation, we assume an implicit security parameter of λ.

Setup. An authority will run the setup algorithm. The algorithm outputs a public key PK and master secret key MSK.

***KeyGen*(MSK, ID).** The key generation algorithm takes in the master secret key MSK and an identity, ID. It generates a private key $\mathrm{SK_{ID}}$ for the identity.

***Encrypt*(S, PK, M).** The encryption algorithm takes as input a revocation set S of identities along with the public key and a message M to encrypt. It outputs a ciphertext CT such that any user with a key for an identity ID $\notin S$ can decrypt.

***Decrypt*(S, CT, ID, $D_{\mathbf{ID}}$)** The decryption algorithm takes as input a ciphertext CT that was generated for the revocation set S, as well as an identity ID and a private key for it. If ID $\notin S$ the algorithm will be able to decrypt and recover the message M encrypted in the ciphertext.

We now define (chosen plaintext) security of an ID-based revocation encryption system against a static adversary. Security is defined using the following "Revocation Game" between an attack algorithm $\mathcal{A}$ and a challenger, for a revocation set S of identities.

Setup. The challenger runs *Setup* to obtain a public key PK and master secret key MSK. It gives $\mathcal{A}$ the public key PK. In addition, it gives $\mathcal{A}$ the decryption keys d_{ID} for all $ID \in S$.

Challenge. The attacker gives the challenger two messages M_0, M_1. Next, the challenger picks a random $b \in \{0, 1\}$. The challenger runs algorithm *Encrypt* to obtain CT $\overset{\text{R}}{\leftarrow}$ *Encrypt*(S, PK, M_b). It then gives CT to algorithm $\mathcal{A}$.

Guess. Algorithm $\mathcal{A}$ outputs its guess $b' \in \{0, 1\}$ for b and wins the game if $b = b'$.

Definition 1 *We say that a revocation system is (chosen-plaintext) secure if, for all revocations sets S of size polynomial in the security parameter, no polynomial-time adversary can win the "Revocation Game" (defined above) with non-negligible advantage over* $1/2$.

Our attack models the game where all users in the revoked set S get together and collude (this is because the adversary gets all private keys from the revoked set).

Chosen-Ciphertext Security. We will also consider chosen-ciphertext (CCA) security, where the adversary can also issue decryption queries for ciphertexts that it constructs (as long as the challenge ciphertexts are not equal to the challenge ciphertext). The game is identical to the game above, except decryption queries (for arbitrary revocation sets) are allowed. Our main construction will be chosen-plaintext secure; however it can be made CCA-secure using the techniques of Cannetti, Halevi, and Katz [15].

2.2 Bilinear Maps

We briefly review the necessary facts about bilinear maps and bilinear map groups. We use the following standard notation [27, 28, 7]:

1. $\mathbb{G}$ and $\mathbb{G}_T$ are two (multiplicative) cyclic groups of prime order p;

2. g is a generator of $\mathbb{G}$.
3. $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear map.

Let $\mathbb{G}$ and $\mathbb{G}_T$ be two groups as above. A bilinear map is a map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1. Bilinear: for all $u, v \in \mathbb{G}$ and $a, b \in \mathbb{Z}$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degenerate: $e(g, g) \neq 1$.

We say that $\mathbb{G}$ is a bilinear group if the group action in $\mathbb{G}$ can be computed efficiently and there exists a group $\mathbb{G}_T$ and an efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ as above. Note that $e(,)$ is symmetric since $e(g^a, g^b) = e(g, g)^{ab} = e(g^b, g^a)$.

2.3 Complexity Assumptions

Decisional Bilinear Diffie-Hellman Assumption The decisional Bilinear Diffie-Hellman problem is defined as follows. We choose a group $\mathbb{G}$ of prime order p. We choose a random generator g of $\mathbb{G}$ and random exponents $c_1, c_2, c_3 \in \mathbb{Z}_p$. If the attacker is given

$$\vec{y} = \{g, g^{c_1}, g^{c_2}, g^{c_3}\},$$

it must remain hard to distinguish $e(g, g)^{c_1 c_2 c_3} \in \mathbb{G}_T$ from a random element of $\mathbb{G}_T$.

An algorithm $\mathcal{B}$ that outputs $z \in \{0, 1\}$ has advantage ϵ in solving decisional BDH in $\mathbb{G}$ if

$$\left| \Pr\left[\mathcal{B}\big(\vec{y}, T = e(g, g)^{c_1 c_2 c_3}\big) = 0\right] - \Pr\left[\mathcal{B}\big(\vec{y}, T = R\big) = 0\right] \right| \geq \epsilon$$

Definition 2 *We say the decisional BDH assumption holds if no poly-time algorithm has a non-negligible advantage in solving the decisional BDH problem.*

Decisional Linear Assumption The decisional Linear problem is defined as follows. We choose a group $\mathbb{G}$ of prime order p. We choose random generators g, f, ν of $\mathbb{G}$ and random exponents $c_1, c_2 \in \mathbb{Z}_p$. If the attacker is given

$$\vec{y} = g, f, \nu, g^{c_1}, f^{c_2},$$

it must remain hard to distinguish $\nu^{c_1 + c_2}$ from a random element of $\mathbb{G}$.

An algorithm $\mathcal{B}$ that outputs $z \in \{0, 1\}$ has advantage ϵ in solving the decisional Linear problem in $\mathbb{G}$ if

$$\left| \Pr\left[\mathcal{B}\big(\vec{y}, T = \nu^{c_1} + c_2\big) = 0\right] - \Pr\left[\mathcal{B}\big(\vec{y}, T = R\big) = 0\right] \right| \geq \epsilon.$$

Definition 3 *We say the decisional Linear assumption holds if no poly-time algorithm has a non-negligible advantage in solving the decisional Linear problem.*

***q*-Decisional Multi-Exponent Bilinear Diffie-Hellman Assumption** To prove the security of our simple system we use a new assumption that we call the q-decisional Multi-Exponent Bilinear Diffie-Hellman assumption. Our assumption falls within a class of assumptions shown to be secure in the generic group model by Boneh, Boyen, and Goh [9]. While our assumption is non-standard, we emphasize that it is non-interactive and thus falsifiable.

Let $\mathbb{G}$ be a bilinear group of prime order p. The q-MEBDH problem in $\mathbb{G}$ is stated as follows:

A challenger picks a generator $g \in \mathbb{G}$ and random exponents $s, \alpha, a_1, \ldots, a_q$. The attacker is then given $\vec{y}$=

$$\begin{array}{lcccc}
 & & g, g^s, e(g,g)^\alpha & & \\
\forall_{1 \le i,j \le q} & g^{a_i} & g^{a_i s} & g^{a_i a_j} & g^{\alpha / a_i^2} \\
\forall_{1 \le i,j,k \le q, i \ne j} & g^{a_i a_j s} & g^{\alpha a_j / a_i^2} & g^{\alpha a_i a_j / a_k^2} & g^{\alpha a_i^2 / a_j^2},
\end{array}$$

it must remain hard to distinguish $e(g,g)^{\alpha \cdot s} \in \mathbb{G}_T$ from a random element in $\mathbb{G}_T$.

An algorithm $\mathcal{B}$ that outputs $z \in \{0,1\}$ has advantage ϵ in solving decisional q-parallel BDHE in $\mathbb{G}$ if

$$\left| \Pr\left[\mathcal{B}\big(\vec{y}, T = e(g,g)^{\alpha s}\big) = 0\right] - \Pr\left[\mathcal{B}\big(\vec{y}, T = R\big) = 0\right] \right| \ge \epsilon.$$

Definition 4 *We say that the q-decisional Multi-Exponent Bilinear Diffie-Hellman assumption holds if no poly-time algorithm has non-negligible advantage in solving the q-MEBDH problem.*

Remark. It is tempting to try to simplify our assumption using previous techniques. For example, we might consider letting choosing a single variable a and substituting all a_j with a^j. Unfortunately, this substitution gives rise to an problem that is insecure.

3 Our Simple Revocation System

We now present our simpler revocation system. Our system has the following features: both public and private keys are of size independent of the number of users (*i.e.* only a constant number of group elements[6]); the ciphertext only contains $O(r)$ group elements, where r is the number of revoked users.

Intuition Our construction uses a novel application of a secret sharing in the exponent. Suppose an encryption algorithm needs to create an encryption with a revocation set $S = \mathrm{ID}_1, \ldots, \mathrm{ID}_r$ of r identities. The algorithm will create an exponent $s \in \mathbb{Z}_p$ and split it into r random shares $s_1, \ldots, s_r$ such that $\sum s_i = s$. It will then create a ciphertext such that any user key with $\mathrm{ID} = \mathrm{ID}_i$ will not be able to incorporate the $i - th$ share and thus not decrypt the message.

Our approach presents us with two challenges. First, we need to make sure that a user with revoked identity $\mathrm{ID} = \mathrm{ID}_i$ cannot do anything useful with share i. Second, we need to worry about collusion attacks between multiple revoked users. Suppose a user with $\mathrm{ID} = \mathrm{ID}_i$ and a user with $\mathrm{ID} = \mathrm{ID}_j$ collude to attack a ciphertext. The attack we need to worry about is where user j processes ciphertext share i, while user i processes share j, and then they combine their results.

The first problem is addressed by the method of decryption. For each share, the ciphertext will have two components. A user with $\mathrm{ID} \ne \mathrm{ID}_i$ can use these two components to obtain two linearly independent equations (in the exponent) involving the share s_i (and another variable), which he will use to solve for the share s_i. However, if $\mathrm{ID} = \mathrm{ID}_i$ he will get two linearly dependent equations and not be able to solve the system. We remark that these techniques are somewhat reminiscent of of those used for knowledge extraction in discrete log proof of knowledge settings [37]. In addition, different types of two equation techniques have been applied in ecash applications (see e.g., [12] and the references therein).

[6]Indeed, since we are using elliptic curves of prime order, these elements can be quite short.

To address the second challenge, we randomize each user's private key by an exponent t such that in decryption each user recovers shares $t \cdot s_i$ in the exponent. Thus, we disallow useful collusions in a similar manner to some Identity-Based [14, 8] and Attribute-Based [36, 24, 6] encryption systems. Our construction follows.

3.1 Simple Construction

In the description of our construction we will use a bilinear group $\mathbb{G}$ of prime order p. We will assume that identities are taken from the set $\mathbb{Z}_p$; in practice, of course, we can perform a collision resistant hash from identity strings to $\mathbb{Z}_p$. We now give our construction as a set of four algorithms.

Setup The setup algorithm chooses a group $\mathbb{G}$ of prime order p. It then picks random generators $g, h \in \mathbb{G}$ and picks random exponents $\alpha, b \in \mathbb{Z}_p$. The public key is published as:

$$\mathrm{PK} = (g, g^b, g^{b^2}, h^b, e(g,g)^\alpha).$$

The authority keeps α, b as secrets.

Key Gen(MSK, ID) The key generation algorithm first chooses a random $t \in \mathbb{Z}_p$ and publishes the private key as:

$$D_0 = g^\alpha g^{b^2 t}, D_1 = (g^{b \cdot \mathrm{ID}} h)^t, D_2 = g^{-t}.$$

Encrypt(PK, M, S) The encryption algorithm first picks a random $s \in \mathbb{Z}_p$. Then it lets $r = |S|$ and chooses random $s_1, \ldots, s_r$ such that $s = s_1 + \ldots + s_r$. We let ID_i denote the i-th identity in S. It then creates the ciphertext CT as:

$$C' = e(g,g)^{\alpha s} M, C_0 = g^s$$

together with, for each $i = 1, 2, \ldots, r$:

$$\left(C_{i,1} = g^{b \cdot s_i}, C_{i,2} = \left(g^{b^2 \cdot \mathrm{ID}_i} h^b \right)^{s_i} \right)$$

Decrypt(S, CT, ID, $D_{\mathbf{ID}}$) If there exists $\mathrm{ID}' \in S$ such that $\mathrm{ID} = \mathrm{ID}'$ then the algorithm aborts; otherwise, the decryption algorithm computes:

$$\frac{e(C_0, D_0)}{e\left(D_1, \prod_{i=1}^r C_{i,1}^{1/(\mathrm{ID}-\mathrm{ID}_i)}\right) \cdot e\left(D_2, \prod_{i=1}^r C_{i,2}^{1/(\mathrm{ID}-\mathrm{ID}_i)}\right)}$$

which gives us $e(g,g)^{\alpha s}$; this can immediately be used to recover the message M from C'. Note that this computation is only defined if $\forall i \quad \mathrm{ID} \neq \mathrm{ID}_i$.

We can verify the correctness of the decryption computation.

$$\begin{aligned}
& e(C_0, D_0) / \left(e\left(D_1, \prod_{i=1}^{r} C_{i,1}^{1/(\mathrm{ID}-\mathrm{ID}_i)}\right) \cdot e\left(D_2, \prod_{i=1}^{r} C_{i,2}^{1/(\mathrm{ID}-\mathrm{ID}_i)}\right) \right) \\
= \; & e(C_0, D_0) / \left(\prod_{i=1}^{r} \left(e\left(D_1, C_{i,1}\right) \cdot e\left(D_2, C_{i,2}\right)\right)^{\mathrm{ID}-\mathrm{ID}_i} \right) \\
= \; & e(g^s, g^\alpha g^{b^2 t}) / \left(\prod_{i=1}^{r} \left(e\left((g^{b\mathrm{ID}} h)^t, g^{bs_i}\right) \cdot e\left(g^{-t}, (g^{b^2 \mathrm{ID}_i} h^b)^{s_i}\right)\right)^{\mathrm{ID}-\mathrm{ID}_i} \right) \\
= \; & e(g,g)^{s\alpha} e(g,g)^{sb^2 t}) / \left(\prod_{i=1}^{r} e(g,g)^{s_i b^2 t}) \right) \\
= \; & e(g,g)^{s\alpha}
\end{aligned}$$

We obtain the following theorem. (The proof appears in Appendix A.)

Theorem 5 *Suppose the decisional q-MEBDH assumption holds. Then no poly-time adversary can selectively break our system with a ciphertext encrypted to $r^* \leq q$ revoked users.*

4 Our Second Revocation System

This system retains the desirable properties of our simpler system: public and private keys still require only a constant number of group elements, and the ciphertext requires $O(r)$ group elements, where r is the number of revoked users. The primary advantage of this system is that we obtain security from simple assumptions, namely the decisional Linear assumption and $d - BDH$.

Intuition We combine the techniques of our simple construction with the dual system encryption technique of Waters [44]. Essentially, we append a version of our simple construction onto the core IBE construction of Waters.

4.1 Construction

We will again use a bilinear group G of order p and assume that identities are taken from $\mathbb{Z}_p$.

Setup The setup algorithm chooses a bilinear group G of prime order p. It then chooses random generators $g, v, v_1, v_2, w, h \in G$ and random exponents $a_1, a_2, b, \alpha \in \mathbb{Z}_p$. It lets $\tau_1 = vv_1^{a_1}, \tau_2 = vv_2^{a_2}$. The public key is published as:

$$PK = (g^b, g^{a_1}, g^{a_2}, g^{ba_1}, g^{ba_2}, \tau_1, \tau_2, \tau_1^b, \tau_2^b, w, h, e(g,g)^{\alpha a_1 b}).$$

The master secret key is:

$$MSK = (g, g^\alpha, g^{\alpha a_1}, v, v_1, v_2, PK).$$

KeyGen(MSK, ID) The key generation algorithm chooses random exponents $d_1, d_2, z_1, z_2 \in \mathbb{Z}_p$ and sets $d = d_1 + d_2$. The private key D_{ID} is:

$$D_1 = g^{\alpha a_1} v^d, D_2 = g^{-\alpha} v_1^d g^{z_1}, D_3 = (g^b)^{-z_1}, D_4 = v_2^d g^{z_2}, D_5 = (g^b)^{-z_2}, D_6 = g^{d_2 b},$$

$$D_7 = g^{d_1}, K = (w^{ID} h)^{d_1}.$$

Encrypt(PK, M, S) The encryption algorithm chooses random exponents $s_1, s_2, t_1, \ldots, t_r$ and sets $s = s_1 + s_2$, $t = t_1 + \cdots + t_r$ (where $r = |S|$, the number of revoked users). We let ID_i denote the i-th identity in S. The ciphertext CT is constructed as:

$$C_0 = M\left(e(g,g)^{\alpha a_1 b}\right)^{s_2}, C_1 = (g^b)^s, C_2 = (g^{ba_1})^{s_1}, C_3 = (g^{a_1})^{s_1},$$

$$C_4 = (g^{ba_2})^{s_2}, C_5 = (g^{a_2})^{s_2}, C_6 = \tau_1^{s_1}\tau_2^{s_2}, C_7 = (\tau_1^b)^{s_1}(\tau_2^b)^{s_2}w^{-t},$$

along with, for each $i = 1, 2, \ldots, r$:

$$C_{i,1} = g^{t_i}, C_{i,2} = (w^{ID_i}h)^{t_i}.$$

Decrypt(S, CT, ID, D_{ID}) If $ID = ID_i$ for some $ID_i \in S$, then the algorithm aborts. Otherwise, the decryption algorithm begins by computing:

$$\begin{aligned} A_1 &= e(C_1, D_1)e(C_2, D_2)e(C_3, D_3)e(C_4, D_4)e(C_5, D_5) \\ &= e(g,g)^{\alpha a_1 b s_2} e(v,g)^{bsd} e(v_1,g)^{a_1 b s_1 d} e(v_2,g)^{a_2 b s_2 d}. \end{aligned}$$

Next, the algorithm computes:

$$\begin{aligned} A_2 &= e(C_6, D_6)e(C_7, D_7) \\ &= e(v,g)^{bsd} e(v_1,g)^{a_1 b s_1 d} e(v_2,g)^{a_2 b s_2 d} e(g,w)^{-d_1 t}. \end{aligned}$$

Now,

$$A_3 = A_1/A_2 = e(g,g)^{\alpha a_1 b s_2} e(g,w)^{d_1 t},$$

so if we separately compute $e(g,w)^{d_1 t}$, we can cancel this term and compute the blinding factor and hence recover the message. We compute $e(g,w)^{d_1 t}$ as follows:

$$\begin{aligned} A_4 &= \prod_{i=1}^{r}\left(\frac{e(C_{i,1}, K)}{e(C_{i,2}, D_7)}\right)^{\frac{1}{ID - ID_i}} \\ &= \prod_{i=1}^{r}\left(e(g,w)^{d_1 t_i (ID - ID_i)}\right)^{\frac{1}{ID - ID_i}} \\ &= \prod_{i=1}^{r} e(g,w)^{d_1 t_i} = e(g,w)^{d_1 t}. \end{aligned}$$

Thus, the message can be computed as:

$$C_0/(A_3/A_4) = M.$$

5 Security

We will prove the following theorem.

Theorem 6 *If the decisional Linear and decisional BDH assumptions hold, then our revocation system above is secure.*

To prove this, we first define semi-functional keys and ciphertexts. These are not used in the real system, but they will be used in our proof of security. These objects have the following functionality: a semi-functional key can decrypt a normal ciphertext and a normal key can decrypt a semi-functional ciphertext. However, a semi-functional key cannot decrypt a semi-functional ciphertext. We define these as in the Waters IBE system:

Semi-Functional Ciphertexts We generate a semi-functional ciphertext by first running the encryption algorithm to produce a normal ciphertext for message M and set S:

$$C'_0, C'_1, C'_2, C'_3, C'_4, C'_5, C'_6, C'_7, C'_{i,1}, C'_{i,2} \forall i \in S.$$

Then we set $C_1 = C'_1, C_2 = C'_2, C_3 = C'_3, C_{i,1} = C'_{i,1}, C_{i,2} = C'_{i,2} \forall i \in S$ (these values are left unchanged). We choose a random $x \in \mathbb{Z}_p$, and set the rest of the ciphertext as:

$$C_4 = C'_4 \cdot g^{ba_2 x}, C_5 = C'_5 \cdot g^{a_2 x}, C_6 = C'_6 \cdot v_2^{a_2 x}, C_7 = C'_7 \cdot v_2^{a_2 bx}.$$

Semi-Functional Keys We generate a semi-functional key by first running the key generation algorithm to produce a normal private key for identity ID:

$$D'_1, D'_2, D'_3, D'_4, D'_5, D'_6, D'_7, K'.$$

Then we set $D_3 = D'_3, D_5 = D'_5, D_6 = D'_6, D_7 = D'_7, K = K'$ (these values are left unchanged). We choose a random $\gamma \in \mathbb{Z}_p$. We set the rest of the key as:

$$D_1 = D'_1 \cdot g^{-a_1 a_2 \gamma}, D_2 = D'_2 \cdot g^{a_2 \gamma}, D_4 = D'_4 \cdot g^{a_1 \gamma}.$$

We will prove selective security of our system under the decisional Linear and d-BDH assumptions through a hybrid argument. We use the following sequence of games which we will show are indistinguishable.

Game$_{Real}$: This denotes the real security game. We let Game$_{Real}Adv_{\mathcal{A}}$ denote the advantage of an algorithm $\mathcal{A}$ in the real security game.

Game$_0$: This is the same as Game$_{Real}$, except that the ciphertext given to the attacker is semi-functional.

Game$_k$: In this game, the ciphertext is semi-functional, and the keys given out for the first k users in the revoked set S are semi-functional, while the rest of the keys are normal. For an adversary that submits a revocation set S of size r, we will let k range from 0 to r. Note that in Game$_r$, the ciphertext and all the keys are semi-functional.

Game$_{Final}$: This is the same as Game$_r$, except that the ciphertext is a semi-functional encryption of a random message instead of M_b.

Lemma 7 *Suppose there exists an algorithm $\mathcal{A}$ such that $Game_{Real}Adv_{\mathcal{A}} - Game_0 Adv_{\mathcal{A}} = \epsilon$. Then we can build an algorithm $\mathcal{B}$ with advantage ϵ in the decision Linear game.*

Proof. (This proof is essentially the same as the proof of Lemma 1 in [44], but we include it for completeness.) $\mathcal{B}$ first receives an instance of the decisional Linear problem: $(G, g, f, \nu, g^{c_1}, f^{c_2}, T)$. $\mathcal{B}$ must decide whether $T = \nu^{c_1+c_2}$ or is random. To accomplish this, $\mathcal{B}$ will call on $\mathcal{A}$ by simulating either Game$_{Real}$ or Game$_0$. $\mathcal{A}$ first sends a set $S = \{ID_1, \ldots, ID_r\}$ to $\mathcal{B}$.

Setup $\mathcal{B}$ chooses random exponents $b, \alpha, y_v, y_{v_1}, y_{v_2} \in \mathbb{Z}_p$ and random group elements $w, h \in G$. It then sets $g = g, g^{a_1} = f, g^{a_2} = \nu, w = w, h = h$. Note that $\mathcal{B}$ does not know the values a_1, a_2. It also sets:

$$g^b, g^{ba_1} = f^b, g^{ba_2} = \nu^b, v = g^{y_v}, v_1 = g^{y_{v_1}}, v_2 = g^{y_{v_2}}.$$

$\mathcal{B}$ also computes $\tau_1, \tau_2, \tau_1^b, \tau_2^b, e(g,g)^{\alpha a_1 b} = e(g,f)^{\alpha b}$. Note that τ_1 (for example) can be computed as $\tau_1 = v v_1^{a_1} = v f^{y_{v_1}}$. $\mathcal{B}$ sends the public parameters to $\mathcal{A}$.

Key Generation $\mathcal{B}$ only needs to produce normal keys for ID_i for all $ID_i \in S$. It can produce these through the usual key generation algorithm since it knows $MSK = \{g, g^{a_1}, \alpha, v, v_1, v_2\}$.

Challenge Ciphertext Once $\mathcal{B}$ has given $\mathcal{A}$ the public parameters and the keys for all elements of $S = \{ID_1, \dots, ID_r\}$, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0, M_1. $\mathcal{B}$ chooses a random value $\beta \in \{0,1\}$ and will create a semi-functional ciphertext for M_β, S as follows. First, $\mathcal{B}$ chooses random exponents, $s_1', s_2', t_1, \dots, t_r$, and uses the normal encryption algorithm to produce $C_0', C_1', \dots, C_7', C_{1,1}', C_{1,2}', \dots, C_{r,1}', C_{r,2}'$. It leaves the terms $C_{i,1} = C_{i,1}', C_{i,2} = C_{i,2}'$ unchanged for i from 1 to r. The rest of the terms are set as:

$$C_0 = C_0' \left(e(g^{c_1}, f)e(g, f^{c_2})\right)^{b\alpha}, C_1 = C_1'(g^{c_1})^b, C_2 = C_2'(f^{c_2})^{-b}, C_3 = C_3'(f^{c_2})^{-1}, C_4 = C_4'(T)^b,$$

$$C_5 = C_5'T, C_6 = C_6'(g^{c_1})^{y_v}(f^{c_2})^{-y_{v_1}}T^{y_{v_2}}, C_7 = C_7'\left((g^{c_1})^{y_v}(f^{c_2})^{-y_{v_1}}T^{y_{v_2}}\right)^b.$$

If $T = \nu^{c_1+c_2}$, this will be a normal ciphertext with $s_1 = -c_2 + s_1'$, $s_2 = c_1 + c_2 + s_2'$, and $s = s_1 + s_2 = c_1 + s_1' + s_2'$. If T is random, this will be a properly distributed semi-functional ciphertext. Thus, $\mathcal{B}$ can use $\mathcal{A}$'s output to obtain the same advantage in distinguishing $T = \nu^{c_1+c_2}$ from random that $\mathcal{A}$ has in distinguishing Game$_{Real}$ from Game$_0$. $\square$

Lemma 8 *Suppose there exists an algorithm $\mathcal{A}$ that submits a revoked set of r users and $Game_{k-1}Adv_{\mathcal{A}} - Game_kAdv_{\mathcal{A}} = \epsilon$ for some k with $1 \leq k \leq r$. Then we can build an algorithm B with advantage ϵ in the decision Linear game.*

Proof. $\mathcal{B}$ first receives an instance of the decisional Linear problem: $(G, g, f, \nu, g^{c_1}, f^{c_2}, T)$. $\mathcal{B}$ must decide whether $T = \nu^{c_1+c_2}$ or is random. To accomplish this, $\mathcal{B}$ will call on $\mathcal{A}$ by simulating either Game$_k$ or Game$_{k-1}$. $\mathcal{A}$ first sends a set $S = \{ID_1, \dots, ID_r\}$ to $\mathcal{B}$.

Setup $\mathcal{B}$ chooses random exponents $\alpha, a_1, a_2, y_{v_1}, y_{v_2}, y_w, y_h \in \mathbb{Z}_p$ and sets the public parameters by computing:

$$g^b = f, g^{a_1}, g^{a_2}, g^{ba_1} = f^{a_1}, g^{ba_2} = f^{a_2}, v = \nu^{-a_1a_2}, v_1 = \nu^{a_2}g^{y_{v_1}}, v_2 = \nu^{a_1}g^{y_{v_2}},$$

$$e(g,g)^{\alpha a_1 b} = e(f,g)^{\alpha a_1}, \tau_1 = vv_1^{a_1}, \tau_2 = vv_2^{a_2}, \tau_1^b = f^{y_{v_1}a_1}, \tau_2^b = f^{y_{v_2}a_2}, w = fg^{y_w}, h = w^{-ID_k g^{y_h}}.$$

Key Generation To generate a normal key for ID_j when $j > k$, the simulator B can run the usual key generation algorithm, since it knows the MSK. To generate a semi-functional key for ID_j when $j < k$, the simulator can run the semi-functional key generation algorithm described above because it knows the exponents a_1 and a_2. For ID_k, the simulator will create a key that is normal if $T = \nu^{c_1+c_2}$ and is semi-functional if T is random.

To generate the key for ID_k, $\mathcal{B}$ starts by running the usual key generation algorithm to produce a normal key SK_{ID_k}: $D_1', D_2', \dots, D_7', K'$. We let d_1', d_2', z_1', z_2' denote the random exponents that were chosen. We then set:

$$D_1 = D_1'T^{-a_1a_2}, D_2 = D_2'T^{a_2}(g^{c_1})^{y_{v_1}}, D_3 = D_3'(f^{c_2})^{y_{v_1}}, D_4 = D_4'T^{a_1}(g^{c_1})^{y_{v_2}},$$

$$D_5 = D_5'(f^{c_2})^{y_{v_2}}, D_6 = D_6'f^{c_2}, D_7 = D_7'(g^{c_1}), K = K'(g^{c_1})^{y_h}.$$

We note that we have implicitly set $z_1 = z_1' - y_{v_1}c_2$ and $z_2' - y_{v_2}c_2$. If $T = \nu^{c_1+c_2}$, then this is a normal key with $d_1 = d_1' + c_1$ and $d_2 = d_2' + c_2$. We can compute K because the w^{ID_k} terms cancel: $K = (w^{ID_k}w^{-ID_k}g^{y_h})^{d_1'+c_1}$. If T is random, we can write T as $T = \nu^{c_1+c_2}g^\gamma$ and we obtain a semi-functional key with γ playing the same role as in the semi-functional key definition above.

Challenge Ciphertext Once $\mathcal{B}$ has given $\mathcal{A}$ the public parameters and the keys for all elements of $S = \{ID_1, \ldots, ID_r\}$, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0, M_1. $\mathcal{B}$ chooses a random value $\beta \in \{0,1\}$ and will create a semi-functional ciphertext for M_β, S as follows. First, $\mathcal{B}$ uses the normal encryption algorithm with randomly chosen exponents s_1', s_2', t' to create $C_0', C_1', \ldots, C_7'$. Then $C_0 = C_0', C_1 = C_1', C_2 = C_2', C_3 = C_3'$ are left unchanged. To add semi-functionality, $\mathcal{B}$ chooses a random exponent $x \in \mathbb{Z}_p$ and sets:

$$C_4 = C_4' f^{a_2 x}, C_5 = C_5' g^{a_2 x}, C_6 = C_6' v_2^{a_2 x}, C_7 = C_7' f^{a_2 y_{v_2} x} \nu^{-a_1 x y_w a_2}.$$

To create C_7, we have implicitly set $g^t = g^{t'} \nu^{a_1 x a_2}$. We let y_ν denote the unknown discrete log of ν in base g. Then, we have set $t = t' + y_\nu a_1 a_2 x$, so t is not known to $\mathcal{B}$, but t' is. For $i \neq k$, $1 \leq i \leq r$, $\mathcal{B}$ sets t_i to be a randomly chosen value. We let t'' denote the sum of these values. Then t_k is defined to be $t' - t'' + y_\nu a_1 a_2 x$. For $i \neq k$, the simulator $\mathcal{B}$ knows the value of t_i, and so can compute:

$$C_{i,1} = g^{t_i}, C_{i,2} = (w^{ID_i} h)^{t_i}.$$

For $i = k$, $\mathcal{B}$ computes:

$$C_{k,1} = g^{t_k} = g^{t'-t''} \nu^{a_1 a_2 x},$$

$$C_{k,2} = (w^{ID_k} w^{-ID_k} g^{y_h})^{y_\nu a_1 a_2 x + t' - t''} = \nu^{y_h a_1 a_2 x} g^{y_h (t' - t'')}.$$

We note that the we could only form the semi-functional ciphertext because $ID_k \in S$: otherwise we would not have been able to use the cancelation of w^{ID_k} to compute the ciphertext term corresponding to the unknown share. This is an essential feature of our argument: the simulator must not be able to test semi-functionality of key k for itself by doing a test decryption on the semi-functional ciphertext it can create. In this case, such a test will fail because the created key k must always be for a revoked user who cannot decrypt, otherwise the semi-functional challenge ciphertext cannot be created.

In summary, when $T = \nu^{c_1+c_2}$, $\mathcal{B}$ has properly simulated Game$_{k-1}$. When T is random, $\mathcal{B}$ has properly simulated Game$_k$. Thus, $\mathcal{B}$ can use $\mathcal{A}$'s output to obtain the same advantage in distinguishing $T = \nu^{c_1+c_2}$ from random that $\mathcal{A}$ has in distinguishing Game$_{k-1}$ from Game$_k$. □

Lemma 9 *Suppose there exists an algorithm $\mathcal{A}$ that submits a revoked set of r users and $Game_r Adv_{\mathcal{A}} - Game_{Final} Adv_{\mathcal{A}} = \epsilon$. Then we can build an algorithm B with advantage ϵ in the decision BDH game.*

Proof. (This proof is essentially the same as the proof of Lemma 3 in [44], but we include it for completeness.) $\mathcal{B}$ first receives an instance of the d-BDH problem: $(g, g^{c_1}, g^{c_2}, g^{c_3}, T)$. $\mathcal{B}$ must decide whether $T = e(g,g)^{c_1 c_2 c_3}$ or is random. To accomplish this, $\mathcal{B}$ will call on $\mathcal{A}$ by simulating either Game$_r$ or Game$_{Final}$. $\mathcal{A}$ first sends a set $S = \{ID_1, \ldots, ID_r\}$ to $\mathcal{B}$.

Setup $\mathcal{B}$ chooses random exponents $a_1, b, y_v, y_{v_1}, y_{v_2}, y_w, y_h \in \mathbb{Z}_p$. It sets:

$$g = g, g^b, g^{a_1}, g^{a_2} = g^{c_2}, g^{ba_1}, g^{ba_2} = (g^{c_2})^b, v = g^{y_v}, v_1 = g^{y_{v_1}},$$

$$v_2 = g^{y_{v_2}}, w = g^{y_w}, h = g^{y_h}, e(g,g)^{a_1 \alpha b} = e(g^{c_1}, g^{c_2})^{a_1 b}.$$

Note that this implicitly sets a_2 to the unknown value c_2 and α to the unknown value $c_1 c_2$. $\mathcal{B}$ also computes $\tau_1 = v v_1^{a_1}, \tau_1^b, \tau_2 = v(g^{c_2})^{y_{v_2}}, \tau_2^b$ and sends the public parameters to $\mathcal{A}$.

Key Generation $\mathcal{B}$ must now generate semi-functional keys for $ID_1, \ldots, ID_r$. For each ID_i, $\mathcal{B}$ chooses random exponents $d_1, d_2, z_1, z_2, \gamma' \in \mathbb{Z}_p$ and sets $d = d_1 + d_2$. The key elements are computed as:

$$D_1 = (g^{c_2})^{-\gamma' a_1} v^d, D_2 = (g^{c_2})^{\gamma'} v_1^d g^{z_1}, D_3 = (g^b)^{-z_1}, D_4 = (g^{c_1})^{a_1} g^{a_1 \gamma'} v_2^d g^{z_2},$$

$$D_5 = g^{-bz_2}, D_6 = g^{d_2 b}, D_7 = g^{d_1}, K = (w^{ID_i} h)^{d_1}.$$

Challenge Ciphertext Once $\mathcal{B}$ has given $\mathcal{A}$ the public parameters and the keys for all elements of $S = \{ID_1, \ldots, ID_r\}$, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0, M_1. $\mathcal{B}$ chooses a random value $\beta \in \{0, 1\}$ and will create either a semi-functional ciphertext for M_β or a semi-functional encryption of a random message.

$\mathcal{B}$ chooses random exponents $s_1, x', t_1, \ldots, t_r$ and sets $t = t_1 + \cdots + t_r$. It forms the ciphertext as:

$$C_0 = M_\beta T^{a_1 b}, C_1 = g^{s_1 b} (g^{c_3})^b, C_2 = g^{b a_1 s_1}, C_3 = g^{a_1 s_1}, C_4 = (g^{c_2})^{x' b}, C_5 = (g^{c_2})^{x'},$$

$$C_6 = \tau_1^{s_1} (g^{c_3})^{y_v} (g^{c_2})^{y_{v_2} x'}, C_7 = (\tau_1^b)^{s_1} (g^{c_3})^{y_v b} (g^{c_2})^{y_{v_2} x' b} w^{-t},$$

$$C_{1,1} = g^{t_1}, C_{1,2} = (w^{ID_1} h)^{t_1}, \ldots, C_{r,1} = g^{t_r}, C_{r,2} = (w^{ID_r} h)^{t_r}.$$

These assignments implicitly set $s_2 = c_3$ and $x = -c_3 + x'$.

If $T = e(g, g)^{c_1 c_2 c_3}$, then this is a properly distributed semi-functional encryption of M_β. If T is random, then this is a properly distributed semi-functional encryption of a random message. Thus, $\mathcal{B}$ can use $\mathcal{A}$'s output to distinguish $T = e(g, g)^{c_1 c_2 c_3}$ from random with the same advantage that $\mathcal{A}$ has in distinguishing Game$_r$ from Game$_{Final}$. $\square$

6 Attribute-Based Encryption

Our simple revocation scheme also gives rise to a new efficient Attribute-Based Encryption (ABE) scheme that allows access policies to be expressed in terms of *any* access formula over attributes. Until the recent work of Ostrovsky, Sahai, and Waters [34], all previous ABE schemes were limited to expressing only monotonic access structures. Our new ABE scheme, however, achieves significantly superior parameters in terms of key size. In the random oracle model, our new scheme will have the following key sizes: public parameters will be only $O(1)$ group elements, and private keys for access structures involving t leaf attributes will be of size $O(t)$. This is a significant improvement over previous work, which needed public parameters consisting of $O(n)$ group elements, and private keys consisting of $O(t \log(n))$ group elements, where n is a bound on the maximum number of attributes that any ciphertext could have. In our scheme, we do not need any such bound.

For brevity, we only describe at a high level what makes our revocation scheme so amenable to incorporation into ABE schemes. The essential property of our revocation scheme is that successful decryption (if a non-revoked user tries to decrypt) allows the user to recover $e(g, g)^{\alpha s}$, where α is a system parameter, while s is a random choice made at the time of encryption. This idea can be applied with α replaced by a linear secret share of α that corresponds to a negated leaf node in an access formula. By the properties of linear secret sharing schemes, and the randomization provided by s, this allows for a secure ABE system to be built using our revocation scheme as a building block.

Taken altogether, our revocation scheme gives a new and much more efficient instiantion of the OSW framework for non-monotonic ABE. We now describe our construction. We refer the reader to [34] for definitions. Our proofs appear in Appendix B.

6.1 Description of ABE construction

We follow the notation of [34] here, and describe our construction in the random oracle model to highlight the most efficient form of our construction.

Setup. The setup algorithm chooses generators g, h and picks random exponents $\alpha', \alpha'', b \in \mathbb{Z}_p$. We define $\alpha = \alpha' \cdot \alpha''$, $g_1 = g^{\alpha'}$ and $g_2 = g^{\alpha''}$.) The public parameters are published as the following, where H is a random oracle that outputs elements of the elliptic curve group:

$$\mathrm{PK} = (g, g^b, g^{b^2}, h^b, e(g,g)^\alpha, H(\cdot)).$$

The authority keeps (α', α'', b) as the master key MK.

Encryption (M, γ, PK). To encrypt a message $M \in \mathbb{G}_T$ under a set of d attributes $\gamma \subset \mathbb{Z}_p^*$, choose a random value $s \in \mathbb{Z}_p$, and choose a random set of d values $\{s_x\}_{x\in\gamma}$ such that $s = \sum_{x\in\gamma} s_x$. Output the ciphertext as

$$E = (\gamma, E^{(1)} = Me(g,g)^{\alpha\cdot s}, E^{(2)} = g^s, \{E_x^{(3)} = H(x)^s\}_{x\in\gamma},$$
$$\{E_x^{(4)} = g^{b\cdot s_x}\}_{x\in\gamma}, \{E_x^{(5)} = g^{b^2\cdot s_x x} h^{b\cdot s_x}\}_{x\in\gamma})$$

Key Generation $(\tilde{\mathbb{A}}, \mathrm{MK}, \mathrm{PK})$. This algorithm outputs a key that enables the user to decrypt an encrypted message *only* if the attributes of that ciphertext satisfy the access structure $\tilde{\mathbb{A}}$. We require that the access structure $\tilde{\mathbb{A}}$ is $NM(\mathbb{A})$ for some monotonic access structure $\mathbb{A}$, (see [34] for a definition of the $NM(\cdot)$ operator) over a set $\mathcal{P}$ of attributes, associated with a linear secret-sharing scheme Π. First, we apply the linear secret-sharing mechanism Π to obtain shares $\{\lambda_i\}$ of the secret α'. We denote the party corresponding to the share λ_i as $\breve{x}_i \in \mathcal{P}$, where x_i is the attribute underlying $\breve{x}_i$. Note that $\breve{x}_i$ can be primed (negated) or unprimed (non negated). For each i, we also choose a random value $r_i \in \mathbb{Z}_p$.

The private key D will consist of the following group elements: For every i such that $\breve{x}_i$ is *not* primed (i.e., is a non-negated attribute), we have

$$D_i = (D_i^{(1)} = g_2^{\lambda_i} \cdot H(x_i)^{r_i}, D_i^{(2)} = g^{r_i})$$

For every i such that $\breve{x}_i$ is primed (i.e., is a negated attribute), we have

$$D_i = (D_i^{(3)} = g_2^{\lambda_i} g^{b^2 r_i}, D_i^{(4)} = g^{r_i b x_i} h^{r_i}, D_i^{(5)} = g^{-r_i})$$

The key D consists of D_i for all shares i.

Decryption (E, D). Given a ciphertext E and a decryption key D, the following procedure is executed: (All notation here is taken from the above descriptions of E and D, unless the notation is introduced below.) First, the key holder checks if $\gamma \in \tilde{\mathbb{A}}$ (we assume that this can be checked efficiently). If not, the output is $\perp$. If $\gamma \in \tilde{\mathbb{A}}$, then we recall that $\tilde{\mathbb{A}} = NM(\mathbb{A})$, where $\mathbb{A}$ is an access structure, over a set of parties $\mathcal{P}$, for a linear secret sharing-scheme Π. Denote $\gamma' = N(\gamma) \in \mathbb{A}$, and let $I = \{i : \breve{x}_i \in \gamma'\}$. Since γ' is authorized, an efficient procedure associated with the linear secret-sharing scheme yields a set of coefficients $\Omega = \{\omega_i\}_{i\in I}$ such that $\sum_{i\in I} \omega_i \lambda_i = \alpha$. (Note, however, that these λ_i are not known to the decryption procedure, so neither is α.)

For every positive (non negated) attribute $\breve{x}_i \in \gamma'$ (so $x_i \in \gamma$), the decryption procedure computes the following:

$$\begin{aligned} Z_i &= e\left(D_i^{(1)}, E^{(2)}\right) / e\left(D_i^{(2)}, E_i^{(3)}\right) \\ &= e\left(g_2^{\lambda_i} \cdot H(x_i)^{r_i}, g^s\right) / e\left(g^{r_i}, H(x)^s\right) \\ &= e\left(g, g_2\right)^{s\lambda_i} \end{aligned}$$

For every negated attribute $\breve{x}_i \in \gamma'$ (so $x_i \notin \gamma$), the decryption procedure computes the following, following a simple analogy to the basic revocation scheme:

$$\begin{aligned} Z_i &= \frac{e\left(D_i^{(3)}, E^{(2)}\right)}{e\left(D_i^{(4)}, \prod_{x \in \gamma}\left(E_x^{(4)}\right)^{1/(x_i - x)}\right) \cdot e\left(D_i^{(5)}, \prod_{x \in \gamma}\left(E_x^{(5)}\right)^{1/(x_i - x)}\right)} \\ &= e\left(g, g_2\right)^{s\lambda_i} \end{aligned}$$

Finally, the decryption is obtained by computing

$$\frac{E^{(1)}}{\prod_{i \in I} Z_i^{\omega_i}} = \frac{Me(g,g)^{s\alpha}}{e(g,g_2)^{s\alpha'}} = M$$

Note on Efficiency and Use of Random Oracle Model. We note that encryption requires only a single pairing, which may be pre-computed, regardless of the number of attributes associated with a ciphertext. We also note that decryption requires two or three pairings per share utilized in decryption, depending on whether the share corresponds to a non-negated attribute or a negated attribute, respectively.

We also note that we use a random oracle for description simiplicity and efficiency of the system. We can, alternatively, realize our hash function concretely as in other previous ABE systems [36, 24, 34].

References

[1] Sattam S. Al-Riyami, John Malone-Lee, and Nigel P. Smart. Escrow-free encryption supporting cryptographic workflow. *Int. J. Inf. Sec.*, 5(4):217–229, 2006.

[2] H. Anton and C. Rorres. *Elementary Linear Algebra, 9th Edition.* 2005.

[3] Walid Bagga, Refik Molva, and Stefano Crosta. Policy-based encryption schemes from bilinear pairings. In *ASIACCS*, page 368, 2006.

[4] Manuel Barbosa and Pooya Farshim. Secure cryptographic workflow in the standard model. In *INDOCRYPT*, pages 379–393, 2006.

[5] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution.* PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[6] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2007.

[7] D. Boneh and M. Franklin. Identity Based Encryption from the Weil Pairing. In *Advances in Cryptology – CRYPTO*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.

[8] Dan Boneh and Xavier Boyen. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles. In *Advances in Cryptology – Eurocrypt*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.

[9] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.

[10] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO*, pages 258–275, 2005.

[11] Robert W. Bradshaw, Jason E. Holt, and Kent E. Seamons. Concealing complex policies with hidden credentials. In *ACM Conference on Computer and Communications Security*, pages 146–157, 2004.

[12] Jan Camenisch, Susan Hohenberger, and Anna Lysyanskaya. Compact e-cash. In *EUROCRYPT*, pages 302–321, 2005.

[13] R. Canetti, J. Garay, G. Itkis, D. Micciancio, M. Naor, and B. Pinkas. Multicast security: A taxonomy and some efficient constructions. In *Proc. IEEE INFOCOM 1999*, volume 2, pages 708–716. IEEE, 1999.

[14] R. Canetti, S. Halevi, and J. Katz. A Forward-Secure Public-Key Encryption Scheme. In *Advances in Cryptology – Eurocrypt*, volume 2656 of *LNCS*. Springer, 2003.

[15] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *Proc. of Eurocrypt 2004*, volume 3027 of *LNCS*, pages 207–222. Springer-Verlag, 2004.

[16] R. Canetti, T. Malkin, and K. Nissim. Efficient communication-storage tradeoffs for multicast encryption. In *Proc. of Eurocrypt 1999*, pages 459–474. Springer-Verlag, 1999.

[17] Melissa Chase. Multi-authority attribute-based encryption. In *The Fourth Theory of Cryptography Conference (TCC 2007)*, 2007.

[18] Cécile Delerablée, Pascal Paillier, and David Pointcheval. Fully collusion secure dynamic broadcast encryption with constant-size ciphertexts or decryption keys. In *Pairing*, pages 39–59, 2007.

[19] Yevgeniy Dodis and Nelly Fazio. Public key broadcast encryption for stateless receivers. In *Digital Rights Management Workshop*, pages 61–80, 2002.

[20] A. Fiat and M. Naor. Broadcast encryption. In *Proc. of Crypto 1993*, volume 773 of *LNCS*, pages 480–491. Springer-Verlag, 1993.

[21] E. Gafni, J. Staddon, and Y.L. Yin. Efficient methods for integrating traceability and broadcast encryption. In *Proc. of Crypto 1999*, volume 1666 of *LNCS*, pages 372–387. Springer-Verlag, 1999.

[22] J. Garay, J. Staddon, and A. Wool. Long-lived broadcast encryption. In *Proc. of Crypto 2000*, volume 1880 of *LNCS*, pages 333–352. Springer-Verlag, 2000.

[23] M.T. Goodrich, J.Z. Sun, and R. Tamassia. Efficient tree-based revocation in groups of low-state devices. In *Proc. of Crypto 2004*, volume 3152 of *LNCS*, pages 511–527. Springer-Verlag, 2004.

[24] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute Based Encryption for Fine-Grained Access Conrol of Encrypted Data. In *ACM conference on Computer and Communications Security (ACM CCS)*, 2006.

[25] Vipul Goyal, Abishek Jain, Omkant Pandey, and Amit Sahai. Bounded ciphertext policy attribute-based encryption. In *ICALP*, 2008.

[26] D. Halevy and A. Shamir. The LSD Broadcast Encryption Scheme. In *Advances in Cryptology – CRYPTO*, volume 2442 of *LNCS*, pages 47–60. Springer, 2002.

[27] A. Joux. A one round protocol for tripartite Diffie-Hellman. In *Proc. of ANTS IV*, volume 1838 of *LNCS*, pages 385–94. Springer-Verlag, 2000.

[28] A. Joux and K. Nguyen. Separating decision Diffie-Hellman from Diffie-Hellman in cryptographic groups. *J. of Cryptology*, 16(4):239–247, 2003. Early version in Cryptology ePrint Archive, Report 2001/003.

[29] Ravi Kumar, Sridhar Rajagopalan, and Amit Sahai. Coding constructions for blacklisting problems without computational assumptions. In *CRYPTO*, pages 609–623, 1999.

[30] Kaoru Kurosawa and Yvo Desmedt. Optimum traitor tracing and asymmetric schemes. In *EUROCRYPT*, pages 145–157, 1998.

[31] Gerome Miklau and Dan Suciu. Controlling access to published data using cryptography. In *VLDB*, pages 898–909, 2003.

[32] D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Proc. of Crypto 2001*, volume 2139 of *LNCS*, pages 41–62. Springer-Verlag, 2001.

[33] M. Naor and B. Pinkas. Efficient trace and revoke schemes. In *Proc. of Financial cryptography 2000*, volume 1962 of *LNCS*, pages 1–20. Springer-Verlag, 2000.

[34] Rafail Ostrovksy, Amit Sahai, and Brent Waters. Attribute Based Encryption with Non-Monotonic Access Structures. In *ACM conference on Computer and Communications Security (ACM CCS)*, 2007.

[35] V.V. Prasolov. *Problems and Theorems in Linear Algebra.* American Mathematical Society, 1994.

[36] A. Sahai and B. Waters. Fuzzy Identity Based Encryption. In *Advances in Cryptology – Eurocrypt*, volume 3494 of *LNCS*, pages 457–473. Springer, 2005.

[37] Claus-Peter Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3), 1991.

[38] A.T. Sherman and D.A. McGrew. Key establishment in large dynamic groups using one-way function trees. *IEEE Trans. Softw. Eng.*, 29(5):444–458, 2003.

[39] Nigel P. Smart. Access control using pairing based cryptography. In *CT-RSA*, pages 111–121, 2003.

[40] D.R. Stinson. On some methods for unconditionally secure key distribution and broadcast encryption. *Des. Codes Cryptography*, 12(3):215–243, 1997.

[41] D.R. Stinson and T.V. Trung. Some new results on key distribution patterns and broadcast encryption. *Des. Codes Cryptography*, 14(3):261–279, 1998.

[42] D.R. Stinson and R. Wei. Combinatorial properties and constructions of traceability schemes and frameproof codes. *SIAM J. Discret. Math.*, 11(1):41–53, 1998.

[43] D.M. Wallner, E.J. Harder, and R.C. Agee. Key management for multicast: Issues and architectures. IETF draft wallner-key, 1997.

[44] B. Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In *Proc. of Crypto 2009*, volume 5677 of *LNCS*, pages 619–636. Springer-Verlag, 2009.

[45] C.K. Wong, M. Gouda, and S. Lam. Secure group communications using key graphs. In *Proc. of SIGCOMM 1998*, 1998.

[46] E. Yoo, N. Jho, J. Cheon, and M. Kim. Efficient broadcast encryption using multiple interpolation methods. In *Proc. of ICISC 2004*, volume 3506 of *LNCS*, pages 87–103. Springer-Verlag, 2005.

A Security of our Simple Revocation System

A.1 Generic Security of Multi-Exponent BDH

We briefly show that are decisional MEBDH assumption is generically secure. We use the generic proof template of Boneh, Boyen, and Goh [9].

Using the terminology from BBG we need to show that $f = \alpha s$ in independent of the polynomials P and Q. We have that $Q = \{1, \alpha\}$ In addition, we have

$$\begin{aligned} P &= \{1, s,\ \forall_{i,j \in [1,q]} \quad a_i,\ a_i s,\ a_i a_j,\ \alpha/(a_i)^2\} \\ &\cup \{\forall_{i,j,k \in [1,q], i \neq j} \quad a_i a_j s,\ \alpha a_j/a_i^2, \alpha a_i a_j/a_k^2, \alpha a_i^2/a_j^2\} \end{aligned}$$

We first note that this case at first might appear to be outside the BBG framework, since the polynomials are rational function (due to the terms with inverses. However, by a simple renaming of terms we can see this is equivalent to an assumption where we use a generator u and let $g = g^{\prod_{j \in [1,q]} a_j^2}$. Applying this substitution we get a a set of polynomials where maximum degree of any polynomial in the set P is $2q + 3$.

We need to also check that f is symbolically independent of the of any two polynomials in P, Q. To realize f from P, Q we would need to have a term of the form αs. We note that no such terms can be realized from the product of two polynomials $p, p' \in P$. If we use the polynomial s as p then no other potential p' has α. If we use $a_i \cdot s$ as p then no other potential p' has α/a_i. Finally, if we use $a_i a_j s$ with $i \neq j$ for p then no other potential p' is of the form $\alpha/(a_i a_j)$ for $i \neq j$. Any dependence on f must have an a term of s in it, but we just eliminated all possibilities.

It follows from the BBG framework that the assumption is then generically secure. In particular, for an attacker that makes at most n queries to the group oracle we have that its advantage is bounded by

$$\frac{(n + 2(q^3 + 4q^2 + 3q) + 2)^2 \cdot (4q + 6)}{2p}$$

In the general case where $n > q^3$ we have that the advantage is $O(n^2 \cdot q/p)$.

A.2 Proof of Security for Simple Revocation System

We now prove the following theorem.

Theorem 10 *Suppose the decisional q-MEBDH assumption holds. Then no poly-time adversary can selectively break our simple revocation system with a ciphertext encrypted to $r^* \leq q$ revoked users.*

Suppose we have an adversary $\mathcal{A}$ with non-negligible advantage $\epsilon = \mathsf{Adv}_{\mathcal{A}}$ in the selective security game against our construction. Moreover, suppose attacks our system with a ciphertext of at most q revoked users. We show how to build a simulator, $\mathcal{B}$, that plays the decisional q-MEBDH problem.

The simulator begins by receiving a q-MEDDH challenge $\vec{X}, T$. The simulator then proceeds in the game as follows.

Init The adversary $\mathcal{A}$ declares a revocation set $S^* = \mathrm{ID}_1, \ldots, \mathrm{ID}_{r^*}$ of size $r^* \leq q$ that he gives to the simulator. (If $r < q$ the simulator will just ignore some of the terms given in $\vec{X}$).

Setup The simulator now creates the public key PK and gives $\mathcal{A}$ the private keys for all identities in S^*. Conceptually, it will set b as $a_1 + a_2 + \cdots a_r$. The simulator first chooses a random $y \in \mathbb{Z}_p$.

The public key PK is published as:

$$\big(g,\ g^b = \prod_{1 \leq i \leq r^*} g^{a_i}, \quad g^{b^2} = \prod_{1 \leq i,j \leq r} (g^{a_i \cdot a_j}), \quad h = \prod_{1 \leq i \leq r^*} (g^{a_i})^{-\mathrm{ID}_i} g^y, \quad e(g,g)^\alpha\big)$$

We observe that the public parameters are distributed identically to the real system and that the revocation set S^* is reflected in the simulation's construction of the parameter h.

Now the simulator must construct all private keys in the revocation set S. For each identity ID_i the simulator will choose a random $z_i \in \mathbb{Z}_p$ and will (implicitly) set the randomness t_i of the ith identity as $t_i = -\alpha/a_i^2 + z_i$.

Setting t_i allows us to generate the private key components for two reasons. First, in the D_0 component we need to cancel out the g^α term that we do not know. Since g^{b^2} contains a term of $g^{a_i^2}$ raising it to the $-\alpha/a_i^2$ will cancel this term. Second, we need to make sure that we can still realize the D_2 component. To generate this we will have several terms of the form $g^{\alpha a_j / a_i^2}$, which we have for $i \neq j$. Yet, if $i = j$ this generates a term g^{α/a_i} that we do not have. However, by our setting of the h parameter a term like this will never appear.

The private key for ID_i is generated as follows:

$$D_0 = \left(\prod_{\substack{1 \leq j,k \leq n \\ \text{s.t. if } j=k \text{ then } j,k \neq i}} (g^{-\alpha a_j a_k / a_i^2}) \right) \prod_{1 \leq j,k \leq n} (g^{a_j a_k})^{z_i}$$

$$D_1 = \left(\prod_{\substack{1 \leq j \leq n \\ j \neq i}} (g^{-\alpha \cdot a_j / a_i^2})^{(\mathrm{ID}_i - \mathrm{ID}_j)} (g^{(\mathrm{ID}_i - \mathrm{ID}_j) \cdot a_j})^{z_i} \right) (g^{-\alpha/a_i^2})^y g^{y z_i}$$

$$D_2 = g^{\alpha/a_i^2} g^{-z_i}$$

Remark. Note that in the above construction, for any fixed coefficient μ, by changing $t_i = -\mu\alpha/a_i^2 + z_i$, and appropriately raising the relevant parts of the construction above to a μ factor, one can create $D_0 = g^{\mu\alpha + b^2 t_i}$, while keeping $D_1 = (g^{b\mathrm{ID}_i} h)^{t_i}$, and $D_2 = g^{-t_i}$. This observation is not relevant to this proof, but will be useful in the proof of our related ABE scheme.

Challenge The simulator receives M_0, M_1 and chooses random $\beta \in \{0,1\}$. The simulator then chooses random $s', s'_1, \ldots, s'_{r^*} \in \mathbb{Z}_p$ such that $s' = \sum_i s'_i$. For notational convenience let $u_i = g^{b^2\mathrm{ID}_i} h^b$, note this is computable from the public parameters, which were already set.

Conceptually, the ciphertext will be encrypted under randomness $\tilde{s} = s + s'$ and be broken into shares $\tilde{s}_i = a_i s/b + s'_i$. Recall, that $b = \sum_j a_j$; therefore, $\sum \tilde{s}_i = \tilde{s}$.

Our methodology is to split s into pieces such that we can simulate all ciphertext components. Conceptually, we will look for a "hole" in each term. We will use the fact that from the simulator's view the function $g^{b\mathrm{ID}_i} h$ has no term of g^{a_i} by cancellation. Therefore, if we raise this to $s \cdot a_i$ the simulator will have all the necessary terms. In this manner we "spread" the different shares of s as $s \cdot a_i/b$, each into its own "slot".

Our proof technique has two important points. First, in simulating the $C_{i,1}$ and $C_{i,2}$ components the b^{-1} term from the shares will cancel out. Second, in generating the $C_{i,2}$ components we will need elements of the form $g^{s a_i a_j}$ that we have for $i \neq j$. Yet, if $i = j$ this creates an element that we do not have. Again, by our setting of h we do not run into this case.

The challenge CT is created as

$$C' = Te(g,g)^{\alpha s'} \cdot M_\beta \quad C_0 = g^s g^{s'} \quad C_{i,1} = g^{s a_i} (\prod_j g^{a_j})^{s'_i} \quad C_{i,2} = \left(\prod_{\substack{1 \le j \le r^* \\ i \neq j}} (g^{s a_i a_j})^{\mathrm{ID}_i - \mathrm{ID}_j} \right) (g^{a_i s})^y u_i^{s'_i}$$

The $C_{i,2}$ equation can be understood by recalling that $C_{i,2} = (g^{b\mathrm{ID}_i} h)^{b\tilde{s}_i}$ and then noting that $b\tilde{s}_i = s a_i + s'_i$.

Guess The adversary will eventually output a guess β' of β. The simulator then outputs 0 to guesses that $T = e(g,g)^{\alpha s}$ if $\beta = \beta'$; otherwise, it and outputs 1 to indicate that it believes T is a random group element in $\mathbb{G}_T$.

When T is a tuple the simulator $\mathcal{B}$ gives a perfect simulation so we have that

$$\Pr\left[\mathcal{B}\left(\vec{X}, T = e(g,g)^{\alpha s}\right) = 0\right] = \frac{1}{2} + \mathsf{Adv}_{\mathcal{A}}.$$

When T is a random group element the message M_β is completely hidden from the adversary and we have $\Pr\left[\mathcal{B}\left(\vec{X}, T = R\right) = 0\right] = \frac{1}{2}$. Therefore, $\mathcal{B}$ can play the decisional q-MEBDH game with non-negligible advantage.

A.3 Remark on Security Parameters

Our system is shown to be secure under a new non-interactive assumption. Our proof, in the standard model, shows that a ciphertext that revokes up to r users is secure if the decisional r-MEBDH assumption holds. We remark that generically, an adversary that makes n queries to a group oracle will have advantage $O(n^2 r/p)$ (see Appendix A.1 for a group of prime order p. Equivalent *generic* security to decisional Bilinear Diffie-Hellman can then be realized by increasing the size of p by just an *additive* factor of $\lg(r)$ bits. We recognize, of course, that in general for concrete groups a simpler assumption is desirable, and leave achieving comparable efficiency under simpler assumptions as an important open problem.

B Proof of Security for ABE scheme

We prove that the security of our main construction in the attribute-based selective-set model reduces to the hardness of the q-MEBDH assumption.

Theorem 11 *If an adversary can break our ABE scheme with advantage ϵ in the attribute-based selective-set model of security, then a simulator can be constructed to play the q-MEBDH game with advantage $\epsilon/2$.*

Proof. Our proof will follow the outline of, and include much of the text from, the proofs of previous ABE schemes [36, 24, 34], but will incorporate the ideas from our new revocation scheme. We note that our revocation scheme, which we will use to realize "negated" attributes in our ABE scheme, is based on the q-MEDDH assumption. The technique we use to deal with ordinary, non-negated attributes, is the same as [24], which was based on the BDDH assumption. To adapt that part to the q-MEDDH assumption, we note that the BDDH assumption is embedded (in many different ways) in the q-MEDDH assumption that we use. In the BDDH assumption, we are given $A = g^{\tilde{a}}, B = g^{\tilde{b}}, g^s$ and must distinguish $e(g,g)^{\tilde{a}\tilde{b}s}$ from a random element. We will implicitly set $\tilde{a} = \alpha/a_1^2$, and $\tilde{b} = a_1^2$. Note that in the q-MEDDH assumption, we are given $A = g^{\tilde{a}}$ and $B = g^{\tilde{b}}$ for these settings of $\tilde{a}$ and $\tilde{b}$. Below we will use A and B to mean these values.

Suppose there exists a polynomial-time adversary $\mathcal{A}$ that can attack our scheme in the selective-set model with advantage ϵ. We build a simulator $\mathcal{B}$ that can play the q-MEDDH game with advantage $\epsilon/2$. The simulation proceeds as follows:

The simulator begins by receiving a q-MEDDH challenge $\vec{X}, Z$. Note that with probability $1/2$, $Z = e(g,g)^{\alpha s}$. We will denote this event as $\Xi = 0$. With probability $1/2$, however, $Z = e(g,g)^z$ where z is a random element of $\mathbb{Z}_p$. We will denote this event as $\Xi = 1$.

Init The simulator $\mathcal{B}$ runs $\mathcal{A}$. $\mathcal{A}$ chooses the challenge set, γ, a set of d members of $\mathbb{Z}_p^*$.

Setup The simulator assigns the public parameters $g_1 = A$ and $g_2 = B$, thereby implicitly setting $\alpha' = \alpha/a_1^2$ and $\alpha'' = a_1^2$.

The simulator will also program the random oracle $H(x)$ as follows. Suppose the adversary queries the oracle on x. If the simulator already answered such a query, it simply returns the same answer. Otherwise, it picks a random $f_x \in \mathbb{Z}_p$ and responds as follows:

$$H(x) = \begin{cases} g^{f_x} & \text{if } x \in \gamma \\ g_2 g^{f_x} & \text{if } x \notin \gamma \end{cases}$$

The simulator sets up the remainder of the public key exactly as in the proof of the revocation scheme, where the revocation set $S^* = \gamma$.

Phase 1 $\mathcal{A}$ adaptively makes requests for several access structures such that γ passes through none of them. Suppose $\mathcal{A}$ makes a request for the secret key for an access structure $\tilde{\mathbb{A}}$ where $\tilde{\mathbb{A}}(\gamma) = 0$. Note that by assumption, $\tilde{\mathbb{A}}$ is given as $NM(\mathbb{A})$ for some monotonic access structure $\mathbb{A}$, over a set $\mathcal{P}$ of parties (whose names will be attributes), associated with a linear secret-sharing scheme Π.

Let M be the share-generating matrix for Π: Recall, M is a matrix over $\mathbb{Z}_p$ with ℓ rows and $n+1$ columns. For all $i = 1, \ldots, \ell$, the i'th row of M is labeled with a party named $\breve{x}_i \in \mathcal{P}$, where x_i is the attribute underlying $\breve{x}_i$. Note that $\breve{x}_i$ can be primed (negated) or unprimed (non-negated). When we consider the column vector $v = (s, r_1, r_2, \ldots, r_n)$, where s is the secret to be shared, and $r_1, \ldots, r_n \in \mathbb{Z}_p$ are randomly chosen, then Mv is the vector of ℓ shares of the secret s according to Π.

We make use of the following well-known observation about linear secret-sharing schemes (see, e.g. [5][7]): If $S \subset \mathcal{P}$ is a set of parties, then these parties can reconstruct the secret iff the column vector $(1, 0, 0, \ldots, 0)$ is in the span of the rows of M_S, where M_S is the submatrix of M containing only those rows that are labeled by a party in S. Note that since $\tilde{\mathbb{A}}(\gamma) = 0$, we know that $\mathbb{A}(\gamma') = 0$, where $\gamma' = N(\gamma)$. Thus, we know that $(1, 0, \ldots, 0)$ is linearly independent of the rows of $M_{\gamma'}$.

During key generation, a secret sharing of the secret $\alpha' = \tilde{a}$ is supposed to be selected. In this simulation, however, we will choose this sharing (implicitly) in a slightly different manner, as we describe now: First, we pick a uniformly random vector $v = (v_1, \ldots, v_{n+1}) \in \mathbb{Z}_p^{n+1}$. Now, we make use of the following simple proposition [2, 35] from linear algebra:

Proposition 12 *A vector π is linearly independent of a set of vectors represented by a matrix N if and only if there exists a vector w such that $Nw = \vec{0}$ while $\pi \cdot w = 1$.*

Since $(1, 0, \ldots, 0)$ is independent of $M_{\gamma'}$, there exists a vector $w = (w_1, \ldots, w_{n+1})$ such that $M_{\gamma'} w = \vec{0}$ and $(1, 0, \ldots, 0) \cdot w = w_1 = 1$. Such a vector can be efficiently computed [2, 35]. Now we define the vector $u = v + (\tilde{a} - v_1)w$. (Note that u is distributed uniformly subject to the constraint that $u_1 = \tilde{a}$.) We will implicitly use the shares $\vec{\lambda} = Mu$. This has the property that for any λ_i such that $\breve{x}_i \in \gamma'$, we have that $\lambda_i = M_i u = M_i v$ has no dependence on $\tilde{a}$.

Now that we have established how to distribute shares to "parties", which map to negated or non negated attributes, we need to show how to generate the key material.

We first describe how to generate decryption key material corresponding to negated parties $\breve{x}_i = x'_i$. Note that by definition, $\breve{x}_i \in \gamma'$ if and only if $x_i \notin \gamma$.

- If $x_i \in \gamma$, then since $\breve{x}_i \notin \gamma'$, we have that λ_i may depend linearly on $\tilde{a}$, and in general $\lambda_i = \mu\tilde{a} + \theta$, for some known constants μ and θ. However, by the simulator's choices at setup, we can invoke the proof of the revocation scheme to generate the appropriate key material. Note that in our setting, the randomness r_i is the name of the randomness t_i from the revocation scheme, and x_i is the name of the identity ID_i. Furthermore, note that with our parameters, we have that $D_i^{(3)} = g_2^{\lambda_i} g^{b^2 r_i} = g^{\mu\alpha} g^{b^2 r_i} \cdot g^{\theta\alpha''}$. Note that $g^{\theta\alpha''}$ can be generated immediately from $g^{\alpha''} = g^{a_1^2}$ which is given as part of the q-MEDDH assumption. The remainder of the key material is generated exactly as specified in the proof of the revocation scheme (see also the remark following the key generation part of the proof).

- If $x_i \notin \gamma$, then since $\breve{x}_i \in \gamma'$, we have that λ_i is independent of any secrets and is completely known to the simulator. In this case, the simulator chooses $r_i \in \mathbb{Z}_p$ at random, and outputs the following:
$$D_i = (D_i^{(3)} = g_2^{\lambda_i + b^2 r_i}, D_i^{(4)} = g^{r_i b x_i} h^{r_i}, D_i^{(5)} = g^{-r_i})$$
Note that the simulator can compute all these elements using elements already computed as part of the computation of the public key (g^{b^2}, g^b, h).

We now describe how to give key material corresponding to non negated parties $\breve{x}_i = x_i$. The simulated key construction techniques for non negated parties is similar to previous work [24, 36].

[7]Here, we are essentially exploiting the equivalence between linear secret-sharing schemes and monotone span programs, as proven in [5]. The proof in [5] is for a slightly different formulation, but applies here as well.

- If $x_i \in \gamma$, then since λ_i has no dependence on any unknown secrets, we simply choose $r_i \in \mathbb{Z}_p$, and output $D_i = (D_i^{(1)} = g_2^{\lambda_i} \cdot H(x_i)^{r_i}, D_i^{(2)} = g^{r_i})$.
- If $x_i \notin \gamma$, then we work as follows: Let $g_3 = g^{\lambda_i}$. Note that the simulator can compute g_3 using A and g. Choose $r_i' \in \mathbb{Z}_p$ at random, and output the components of D_i as follows:

$$\begin{aligned} D_i^{(1)} &= g_3^{-f_{x_i}}(g_2 g^{f_{x_i}})^{r_i'} \\ D_i^{(2)} &= g_3^{-1} g^{r_i'} \end{aligned}$$

Claim 13 *The simulation above produces valid decryption keys, that are furthermore distributed identically to the decryption keys that would have been produced by the ABE scheme for the same public parameters.*

Proof.

We will establish this claim by a case analysis. For key material corresponding to negated parties $\breve{x}_i$, this has already been verified in the proof of the revocation scheme.

For key material corresponding to non negated parties $\breve{x}_i$:

- If $x_i \in \gamma$, then the simulation produces key material using the same procedure as the ABE scheme.
- If $x_i \notin \gamma$, then to see why the simulated key material is good, note that by our programming of the hash function $H(x)$ has a g_2 component for all $x_i \notin \gamma$. Now let $r_i = r_i' - \lambda_i$. Note that r_i is distributed uniformly over $\mathbb{Z}_p$ and is independent of all other variables except r_i'. Then,

$$\begin{aligned} D_i^{(1)} &= g_3^{-f_{x_i}}(g_2 g^{f_{x_i}})^{r_i'} \\ &= g^{-\lambda_i f_{x_i}}(g_2 g^{f_{x_i}})^{r_i'} \\ &= g_2^{\lambda_i}(g_2 g^{f_{x_i}})^{-\lambda_i}(g_2 g^{f_{x_i}})^{r_i'} \\ &= g_2^{\lambda_i}(g_2 g^{f_{x_i}})^{r_i'-\lambda_i} \\ &= g_2^{\lambda_i} H(x_i)^{r_i} \end{aligned}$$

  and

$$D_i^{(2)} = g_3^{-1} g^{r_i'} = g^{r_i'-\lambda_i} = g^{r_i}$$

□

Challenge The adversary $\mathcal{A}$, will submit two challenge messages M_0 and M_1 to the simulator. Let C denote $g^s g^{s'}$, where s' is chosen at random, and g^s is as provided by the q-MEDDH assumption. The simulator flips a fair binary coin ν, and returns an encryption of M_ν. The ciphertext is output as

$$E = \left(\gamma, E^{(1)} = M_\nu Z, E^{(2)} = C, \{E_x^{(3)} = C^{f(x)}\}_{x\in\gamma}, \{E_x^{(4)}\}, \{E_x^{(5)}\}\right)$$

where $\{E_x^{(4)}\}, \{E_x^{(5)}\}$ are constructed exactly as $C_{i,1}$ and $C_{i,2}$, respectively, in the proof of the revocation scheme.

If $\Xi = 0$ then $Z = e(g,g)^{\alpha s}$. Then by inspection, the ciphertext is a valid ciphertext for the message M_ν under the set γ.

Otherwise, if $\Xi = 1$, then $Z = e(g,g)^z$. We then have $E^{(1)} = M_\nu e(g,g)^z$. Since z is random, $E^{(1)}$ will be a random element of $\mathbb{G}_T$ from the adversary's viewpoint and the message contains no information about M_ν.

Phase 2 The simulator acts exactly as it did in Phase 1.

Guess $\mathcal{A}$ will submit a guess ν' of ν. If $\nu' = \nu$ the simulator will output $\Xi' = 0$ to indicate that it was given a valid q-MEDDH tuple; otherwise, it will output $\Xi' = 1$ to indicate it was given a random target element Z.

As shown above, the simulator's generation of public parameters and private keys is identical to that of the actual scheme.

In the case where $\Xi = 1$ the adversary gains no information about ν. Therefore, we have $\Pr[\nu \neq \nu' | \Xi = 1] = \frac{1}{2}$. Since the simulator guesses $\Xi' = 1$ when $\nu \neq \nu'$, we have $\Pr[\Xi' = \Xi | \Xi = 1] = \frac{1}{2}$.

If $\Xi = 0$ then the adversary sees an encryption of M_ν. The adversary's advantage in this situation is ϵ by assumption. Therefore, we have $\Pr[\nu = \nu' | \Xi = 0] = \frac{1}{2} + \epsilon$. Since the simulator guesses $\Xi' = 0$ when $\nu = \nu'$, we have $\Pr[\Xi' = \Xi | \Xi = 0] = \frac{1}{2} + \epsilon$.

The overall advantage of the simulator in the q-MEDDH game is $\frac{1}{2}\Pr[\Xi' = \Xi | \Xi = 0] + \frac{1}{2}\Pr[\Xi' = \Xi | \Xi = 1] - \frac{1}{2} = \frac{1}{2}(\frac{1}{2} + \epsilon) + \frac{1}{2}\frac{1}{2} - \frac{1}{2} = \frac{1}{2}\epsilon$. $\square$