Hierarchical Identity Based Encryption with Polynomially Many Levels

Craig Gentry (Stanford & IBM) and Shai Halevi* (IBM)

Abstract. We present the first hierarchical identity based encryption (HIBE) system that has full security for more than a constant number of levels. In all prior HIBE systems in the literature, the security reductions suffered from exponential degradation in the depth of the hierarchy, so these systems were only proven fully secure for identity hierarchies of constant depth. (For deep hierarchies, previous work could only prove the weaker notion of selective-ID security.) In contrast, we offer a tight proof of security, regardless of the number of levels; hence our system is secure for polynomially many levels.
Our result can very roughly be viewed as an application of Boyen's framework for constructing HIBE systems from exponent-inversion IBE systems to a (dramatically souped-up) version of Gentry's IBE system, which has a tight reduction. In more detail, we first describe a generic transformation from "identity based broadcast encryption with key randomization" (KR-IBBE) to a HIBE, and then construct KR-IBBE by modifying a recent construction of IBBE of Gentry and Waters, which is itself an extension of Gentry's IBE system. Our hardness assumption is similar to that underlying Gentry's IBE system.

* Research was sponsored by US Army Research laboratory and the UK Ministry of Defense and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

Table of Contents

1 Introduction

Identity-Based Encryption (IBE) is a public-key encryption scheme where one's public key can be freely set to any value (such as one's identity): An authority that holds a master secret key can take any arbitrary identifier and extract a secret key corresponding to this identifier. Anyone can then encrypt messages using the identifier as a public encryption key, and only the holder of the corresponding secret key can decrypt these messages. This concept was introduced by Shamir [18], a partial solution was proposed by Maurer and Yacobi [17], and the first fully functional IBE systems were described by Boneh and Franklin [5] and Cocks [11].

IBE systems can greatly simplify the public-key infrastructure for encryption solutions, but they are still not as general as one would like. Many organizations have an hierarchical structure, perhaps with one central authority, several sub-authorities and sub-sub-authorities and many individual users, each belonging to a small part of the organization tree. We would like to have a solution where each authority can delegate keys to its sub-authorities, who in turn can keep delegating keys further down the hierarchy to the users. The depth of the hierarchy can range from two or three in small organizations, up to ten or more in large ones. An IBE system that allows delegation as above is called Hierarchical Identity-Based Encryption (HIBE). In HIBE, messages are encrypted for *identity-vectors*, representing nodes in the identity hierarchy. This concept was introduced by Horwitz and Lynn [16], who also described a partial solution to it, and the first fully functional HIBE system was described by Gentry and Silverberg [14].

The security model for IBE and HIBE systems postulates an attacker that can adaptively make "key-reveal" queries, thereby revealing the decryption keys of identities of its choice. The required security property asserts that such an attacker still cannot break the encryption at any identity other than those for which it issued key-reveal queries. (Or in the case of HIBE, other than those for which it issued key-reveal queries or their descendants.)

For the first IBE and HIBE systems, the only known proofs of security are carried out in the random-oracle model. Canetti et al. [9] introduced a weaker notion of security called *selective-ID*, where the attacker must choose the identity to attack before the system parameters are chosen (but can still make adaptive key-reveal queries afterward). They proved that a variant of the Gentry-Silverberg system is secure in this model even without random oracles. Boneh and Boyen described a more efficient selective-ID HIBE [1], and later described a fully secure IBE system without a random oracle [2]. Waters [21] described what is currently the most practical adaptively-secure HIBE system without random oracles.

All currently known fully-secure HIBE systems, however, suffer from loose security reductions (whether they use random oracles or not). Specifically, they lose a multiplicative factor of $\Omega(q/\ell)^\ell$ in the success probability, where q is the number of key-reveal queries and ℓ is the depth of the identity hierarchy. This means that asymptotically these reductions can only be used for hierarchies of constant depth. When considering concrete parameters, these reductions are only meaningful for hierarchies of depth two or three.

Gentry [13] proposed the first adaptively-secure IBE system without random oracles that has a tight reduction to its underlying hard problem. Recently Gentry and Waters extended Gentry's IBE to construct an adaptively-secure identity based broadcast encryption (IBBE) system without random oracles [15], whose security is tightly based on a related hard problem. Our HIBE system builds on the Gentry-Waters system.

Boyen [8] proposed a framework for constructing HIBE systems from exponent-inversion IBE systems. Specifically, Boyen described some properties of pairing-based IBE systems (called parallel IBE and linear IBE), and proved that an IBE system with these properties can be transformed to HIBE with comparable security. Boyen noted that Gentry's IBE does not quite fit within this template, and left it as an open problem to construct a HIBE system from Gentry's IBE system. Our system, which solves this problem, does not quite fit within Boyen's framework, yet our approach owes much to Boyen's idea.

We construct the first fully-secure HIBE with a tight proof of security. Namely, ours is the first HIBE system that can be proven fully secure for more than a small constant number of levels. As for the systems of Gentry [13] and Gentry-Waters [15], we exhibit a tight reduction, albeit to a problem whose instances are of size linear in $q + \ell$. This solves an open problem posed in [14, 1, 2, 21, 3, 13, 8].

1.1 Loose and Tight Reductions

On a high level, the reason that most IBE systems have loose reductions is that those reductions involve the following trade-off: For each identity ID, either the simulator knows a decryption key for ID, or it doesn't. If it knows a key for ID then it does not learn anything new if the adversary chooses ID as the target identity to attack, since it could have used the decryption key to learn the same information. And if the simulator does not know a decryption key for ID then it must abort if the adversary makes a key-reveal query for this identity.

The crucial difference in the security proof of Gentry's IBE [13] is that there are many different decryption keys for each identity, and the simulator knows a *small subset* of these keys. Thus, the simulator can answer every key-reveal query without aborting, but still learn something when the adversary choses that identity for the challenge ciphertext. In this sense, Gentry's IBE system follows the universal hash proof paradigm of Cramer and Shoup [12]: Given a well-formed ciphertext, all the decryption keys recover the same message, but they recover different messages when the ciphertext is mal-formed (in a certain sense). The adversary is assumed to have a non-negligible advantage when the challenge ciphertext is well-formed, but has essentially no advantage (statistically) when it is mal-formed; the adversary's different behavior in these cases allows the simulator to solve the underlying decision problem. Gentry's reduction uses an underlying hard problem that has a large problem instance (size $\theta(q)$), to ensure that the adversary cannot use its q key-reveal queries to determine what keys the simulator possesses for the target identity. In this work we extend Gentry's IBE system and proof to the case of a HIBE.

1.2 Constructing HIBE, Step 1: From IBBE to HIBE

In our quest to construct HIBE, we use as an intermediate step a specific type of identity-based broadcast encryption (IBBE). An IBBE system can be seen as somewhere in between regular IBE and HIBE: It allows a sender to encrypt a message to a set identities, and each member of this set can use its own key to decrypt the message. This is somewhat similar to HIBE, in that encryption is targeted at a group of identities (similarly to the identity vector in HIBE).[1] However, IBBE is simpler than HIBE since decryption keys correspond only to single identities (see Section 2.2).

As a first step in constructing HIBE systems, we provide a generic transformation from IBBE to HIBE. This transformation, however, requires an "augmented IBBE system" that also has decryption keys corresponding to sets of identities (for decrypting ciphertexts that were encrypted for these sets). Specifically, we require a *key-randomizable identity based broadcast encryption* (KR-IBBE), where it is possible to generate a *uniformly random* decryption key K_S for a set of identities S from any decryption key $K_{S'}$ for $S' \subsetneq S$ (see Section 2.3). KR-IBBE is rather close to HIBE, but a major difference is that security is defined with respect to an adversary that can only ask for decryption keys corresponding to single identities, not for sets of identities. Hence it is still simpler to design KR-IBBE and use our transformation than to design a HIBE "from scratch."

1.3 Constructing HIBE, Step 2: Constructing KR-IBBE

Even with the simplification of KR-IBBE, our construction and its proof are still rather complex. Part of the reason for the complexity of our system and proof stems from the inherent tension between the key-randomization requirement and the Cramer-Shoup proof paradigm: On one hand, key-randomization implies in particular that one can generate a *random* decryption key for an identity set S from any fixed valid encryption key for the same set. On the other hand, the Cramer-Shoup paradigm require that the simulator be able to generate only a small subset of the decryption keys for the target identity set.

Our proof resolves this tension by going through an intermediate step in which we replace the full-randomization requirement with "pseudo-randomization": Namely, from each fixed valid encryption key we can only generate a small subset of the decryption keys, but this small subset still looks random. In our case, the difference between "fully-random" and "pseudo-random" keys is that "fully random" keys are taken from some linear space and "pseudo-random" keys are taken from a proper subspace of this linear space. Being linear spaces of group elements, these are indistinguishable under the Decision Linear Assumption [4].

We prove the security of the "pseudo-random" system using techniques and hard problems analogous to those used by Gentry and Waters in [15], but we we need to make rather substantial modifications to the system given in [15]. (Most notably we cannot have scalars in the decryption key, so we must convert all the parameters, decryption keys, and ciphertexts into vectors of group elements.)

1.4 Organization

We begin in Section 2 below by introducing the formal definitions of HIBE, IBBE, and KR-IBBE, and then in Section 3 we describe and prove our transformation from KR-IBBE to HIBE. We then move to constructing KR-IBBE, first presenting in Section 4 the notations, tools, and assumptions on which we base out construction, and then describing the construction itself in Section 5. The proof of security is presented in Section 6.

[1] Since we are interested in IBBE as a tool for constructing HIBE, we use here a variant where there are only few recipients and they must be enumerated explicitly by the encryption procedure, as opposed to the more common variant where one needs to enumerate the "revoked" recipients. Arguably, this variant should be called *multicast* encryption rather than broadcast encryption.

2 HIBE and IBBE: Definitions

For simplicity, we define our encryption systems as key encapsulation mechanism (KEM). The standard transformation from KEM to encryption is ignored here.

2.1 Hierarchical Identity-Based Encryption

A HIBE system consists of the following five procedures:

$\mathsf{Setup}(\lambda, \ell)$ Takes as input a security parameter λ and the hierarchy depth ℓ. It outputs a public key PK and a master secret key SK. The public key implies also a key space $\mathcal{K}(PK)$ and an identity space $\mathcal{ID}(PK)$, and hierarchical identities are (ordered) tuples in $\mathcal{ID}(PK)^{\leq \ell}$.

$\mathsf{KeyGen}(PK, SK, \mathbf{ID})$ Takes as input the public key PK and master secret key SK, and an identity vector $\mathbf{ID} = [\mathsf{ID}_1, \ldots, \mathsf{ID}_t] \in \mathcal{ID}(PK)^{\leq \ell}$. It outputs a decryption key $K_{\mathbf{ID}}$ for $\mathbf{ID}$.

$\mathsf{KeyDerive}(PK, \mathbf{ID}, K_{\mathbf{ID}}, \mathbf{ID'})$ Takes as input the public key PK, the identity vector $\mathbf{ID}$ and corresponding decryption key $K_{\mathbf{ID}}$, and another vector $\mathbf{ID'}$ such that $\mathbf{ID}$ is a prefix of $\mathbf{ID'}$. It outputs a decryption key $K_{\mathbf{ID'}}$ for $\mathbf{ID'}$.

$\mathsf{KEM}(PK, \mathbf{ID})$ Takes as input the public key PK and identity vector $\mathbf{ID}$. It outputs a pair (K, C), where K is the KEM key (from the key space $\mathcal{K}(PK)$) and C is the ciphertext.

$\mathsf{Decrypt}(PK, C, \mathbf{ID}, K_{\mathbf{ID}})$ Takes as input the public key PK, ciphertext C, identity vector $\mathbf{ID}$ and corresponding decryption key $K_{\mathbf{ID}}$. It outputs the corresponding KEM key K (or an error message $\perp$).

We require the usual "completeness", namely that decryption with the correct decryption key always recovers the correct KEM key. In particular, setting $(PK, SK) \leftarrow \mathsf{Setup}(\lambda, \ell)$ and fixing any chain of identity vectors $\mathbf{ID}_1$,$\mathbf{ID}_2$,...,$\mathbf{ID}_t$ with each $\mathbf{ID}_i$ a prefix of $\mathbf{ID}_{i+1}$, if we set $K_{\mathbf{ID}_1} \leftarrow \mathsf{KeyGen}(PK, SK, \mathbf{ID}_1)$ and then $K_{\mathbf{ID}_i} \leftarrow \mathsf{KeyDerive}(PK, \mathbf{ID}_i, K_{\mathbf{ID}_{i-1}}, \mathbf{ID}_i)$ for $i = 2, \ldots, t$ and $(K, C) \leftarrow \mathsf{KEM}(PK, \mathbf{ID}_t)$, then we have $\mathsf{Decrypt}(PK, C, \mathbf{ID}_t, K_{\mathbf{ID}_t}) = K$ (with probability one).

Security. [2] Chosen-plaintext security for a HIBE system $\mathcal{E}$ against an adversary A is defined by the following game between A and a "challenger": Both A and the challenger are given the parameters λ, ℓ as inputs.

Setup: The challenger chooses a "challenge bit" $\sigma \in_R \{0, 1\}$, runs $(PK, SK) \leftarrow \mathcal{E}.\mathsf{Setup}(\lambda, \ell)$, and gives PK to A.

Key-Reveal: The adversary A makes adaptive key-reveal queries to the challenger, each consisting of an identity vector $\mathbf{ID} = [\mathsf{ID}_1, \ldots, \mathsf{ID}_t] \in \mathcal{ID}(PK)^{\leq \ell}$. If the adversary already made the challenge query and $\mathbf{ID}$ is a prefix of the target identity $\mathbf{ID}^*$ then the challenger ignores this query, and otherwise it returns to the adversary the decryption key $K_{\mathbf{ID}} \leftarrow \mathcal{E}.\mathsf{KeyGen}(PK, SK, \mathbf{ID})$.

Challenge: The adversary queries the challenger with the target identity vector $\mathbf{ID}^* = [\mathsf{ID}^*_1, \ldots, \mathsf{ID}^*_t] \in \mathcal{ID}(PK)^{\leq \ell}$. If the adversary already made a challenge query before, or if it made a key-reveal query for any prefix of the target identity $\mathbf{ID}^*$ then the challenger ignores this query. Otherwise the challenger sets $(K_1, C) \leftarrow \mathcal{E}.\mathsf{KEM}(PK, \mathbf{ID}^*)$, chooses another key at random from the key space $K_0 \in_R \mathcal{K}(PK)$, and returns (K_σ, C) to the adversary (where σ is the challenge bit chosen during Setup).

[2] Our security definition ignores the delegation issue that was raised by Shi and Waters [19], since it does not apply to our system; see discussions at the end of Sections 2.3 and 3.

The adversary can make many Key-Reveal queries and one Challenge query, in whatever order. Then it halts, outputting a guess σ' for the bit σ. The HIBE advantage of A is

$$\mathsf{AdvHIBE}_A^{\mathcal{E}}(\lambda, \ell) \;=\; \Pr[A \Rightarrow 1 | \sigma = 1] \;-\; \Pr[A \Rightarrow 1 | \sigma = 0]$$

Definition 1 (CPA-secure HIBE). *The system $\mathcal{E}$ is CPA-secure if for any efficient adversary $\mathcal{A}$ and any $\ell = poly(\lambda)$ it holds that $\mathsf{AdvHIBE}_A^{\mathcal{E}}(\lambda, \ell(\lambda))$ is negligible in λ.*

CCA-security is defined similarly, where the adversary can also make decryption queries (except for decrypting the target ciphertext by the target identity vector).

2.2 Identity-Based Broadcast Encryption

Identity-Based Broadcast Encryption (IBBE) is somewhat similar to HIBE, in that encryption is targeted at a group of identities (similarly to the identity vector in HIBE). However, IBBE is simpler than HIBE, since decryption keys correspond to a single identity. Formally, an IBBE system consists of the following procedures:

$\mathsf{Setup}(\lambda, \ell)$ Takes as input a security parameter λ and an upper bound on the group size ℓ. It outputs a public key PK (that implies a keys space $\mathcal{K}(PK)$ and an identity space $\mathcal{ID}(PK)$) and a master secret key SK.

$\mathsf{KeyGen}(PK, SK, \mathbf{ID})$ Takes as input the public key PK, master secret key SK, and an identity $\mathsf{ID} \in \mathcal{ID}(PK)$. It outputs a decryption key K_{ID} for ID.

$\mathsf{KEM}(PK, S)$ Takes as input the public key PK and a set of identities $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_t\}$ (with $t \leq \ell$). It outputs a pair (K, C), where K is the KEM key (from key space $\mathcal{K}(PK)$) and C is the ciphertext.

$\mathsf{Decrypt}(PK, C, S, \mathsf{ID}, K_{\mathsf{ID}})$ Takes as input the public key PK, ciphertext C, identity set $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_t\}$ (with $t \leq \ell$) and the decryption key K_{ID} for some $\mathsf{ID} \in S$. It outputs the corresponding KEM key K (or an error message $\perp$).

We require that setting $(PK, SK) \leftarrow \mathsf{Setup}(\lambda, \ell)$ and fixing any set of identities $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_t\} \subset \mathcal{ID}(PK)^{\leq \ell}$, if we set $K_{ID_1} \leftarrow \mathsf{KeyGen}(PK, SK, ID_1)$ and $(K, C) \leftarrow \mathsf{KEM}(PK, S)$, then we have $\mathsf{Decrypt}(PK, C, S, K_{\mathsf{ID}_1}) = K$ (with probability one).

Security. Chosen-plaintext security IBBE is similar to the corresponding definition for HIBE, with the main difference being that the adversary can only make key-reveal queries for single identities rather than identity-vectors. Let $\mathcal{E}$ be an IBBE system, CPA-security against an adversary A is defined by the following game between A and a "challenger": Both A and the challenger are given the parameters λ, ℓ as inputs.

Setup: The challenger chooses a "challenge bit" $\sigma \in_R \{0, 1\}$, runs $(PK, SK) \leftarrow \mathcal{E}.\mathsf{Setup}(\lambda, \ell)$, and gives PK to A.

Key-Reveal: The adversary A makes adaptive key-reveal queries to the challenger. Each key-reveal query consists of an identity $\mathsf{ID} \in \mathcal{ID}(PK)$. If the adversary already made the challenge query and $\mathsf{ID} \in S^*$ then the challenger ignores this query, and otherwise it returns $K_{\mathsf{ID}} \leftarrow \mathcal{E}.\mathsf{KeyGen}(PK, SK, \mathsf{ID})$ to the adversary.

Challenge: The adversary queries the challenger with the target identity set $S^* = \{\mathsf{ID}_1^*, \ldots, \mathsf{ID}_t^*\} \in \mathcal{ID}(PK)^{\leq \ell}$. If the adversary already made a challenge query before, or if made a key-reveal query for any identity $\mathsf{ID} \in S^*$, then the challenger ignores this query. Otherwise the challenger sets $(K_1, C) \leftarrow \mathcal{E}.\mathsf{KEM}(PK, \mathbf{ID}^*)$, chooses another key at random from the key space $K_0 \in_R \mathcal{K}(PK)$, and returns (K_σ, C) to the adversary (where σ is the bit chosen during Setup).

The adversary can make many Key-Reveal queries and one Challenge query, in whatever order. Then it halts, outputting a guess σ' for the bit σ. The IBBE advantage of A is

$$\mathsf{AdvIBBE}_A^{\mathcal{E}}(\lambda, \ell) \;=\; \Pr[A \Rightarrow 1 | \sigma = 1] \;-\; \Pr[A \Rightarrow 1 | \sigma = 0]$$

Definition 2 (CPA-secure IBBE). *The system $\mathcal{E}$ is CPA-secure if for any efficient adversary $\mathcal{A}$ and any $\ell = poly(\lambda)$ it holds that $\mathsf{AdvHIBE}_A^{\mathcal{E}}(\lambda, \ell(\lambda))$ is negligible in λ.*

CCA-security is defined similarly, where the adversary can also make decryption queries (except for decrypting the target ciphertext by any of the identities in the target set).

2.3 Key-Randomizable IBBE

To construct HIBE systems, we will use "augmented IBBE systems" that also have decryption keys corresponding to sets of identities: A decryption key corresponding to an identity-set S makes it possible to decrypt ciphertexts that were created with respect to this set. Formally, a *Key-Randomizable Identity-Based Broadcast Encryption* system (KR-IBBE) is an IBBE system with extended key generation KeyGen^*, extended decryption $\mathsf{Decrypt}^*$, and key-derivation $\mathsf{KeyDerive}$, as follows:

$\mathsf{KeyGen}^*(PK, SK, S)$ Takes as input the public key PK, master secret key SK, and an identity set $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_t\} \in \mathcal{ID}(PK)^{\leq \ell}$, and outputs a decryption key K_S for S. We require that $\mathsf{KeyGen}^*(PK, SK, S)$ degenerates to the original KeyGen when S is a singleton set $S = \{\mathsf{ID}\}$.

$\mathsf{KeyDerive}(PK, S, K_S, S')$ Takes as input the public key PK, an identity set S and corresponding decryption key K_S, and a superset $S' \supseteq S$, and outputs a decryption key $K_{S'}$ for S'.[3]

$\mathsf{Decrypt}^*(PK, C, S, K_S)$ Takes as input the public key PK, an identity set S, ciphertext C that was generated with respect to S, and the decryption key K_S for S. It outputs the KEM key K (or an error message $\perp$).

We stress that *we make no security requirements* regarding these additional procedures: the CPA-security game is still defined with respect to the original four procedures Setup,KeyGen,KEM, $\mathsf{Decrypt}$. However, we do make some functionality requirements, specifically the standard "completeness" requirement on $\mathsf{Decrypt}^*$ and a distribution requirement on $\mathsf{KeyDerive}$.

The "completeness" requirement says that for any $(PK, SK) \leftarrow \mathsf{Setup}(\lambda, \ell)$ and any set of identities S, if we set $K_S \leftarrow \mathsf{KeyGen}^*(PK, SK, S)$ and $(K, C) \leftarrow \mathsf{KEM}(PK, S)$, then we get $\mathsf{Decrypt}(PK, C, S, K_S) = K$ (with probability one).

The distribution requirement says for any $(PK, SK) \leftarrow \mathsf{Setup}(\lambda, \ell)$, any two sets of identities $S \subseteq S'$, and any decryption key $K_S \leftarrow \mathsf{KeyGen}^*(PK, SK, S)$, the output distributions of

[3] Note that in this setting of broadcast encryption, keys corresponding to smaller sets are "more powerful" than ones corresponding to larger sets: one can derive a key for the superset S' from any key for a subset S, but not the other way around.

$\mathsf{KeyGen}^*(PK, SK, S')$ and $\mathsf{KeyDerive}(PK, S, K_S, S')$ are almost identical. (That is, their statistical distance is negligible in λ.)

Remark. Due to the distribution requirement above, our transformation from key-randomizable IBBE to HIBE in Section 3 results in a system where the decryption keys generated by KeyDerive have the same distribution as the ones generated by KeyGen. Hence, the delegation issue in the HIBE definition, which was noted by Shi and Waters [19], does not apply to our system. Namely, proving security with respect to our Definition 1 implies also security with respect to the more complicated definition from [19].

3 From Key-Randomizable IBBE to HIBE

The transformation from key-randomizable IBBE to HIBE is quite straightforward: we use collision-resistant hashing to map identity-vectors to identity-sets, and then just use each of the procedures Setup, KeyGen*, KeyDerive, KEM, Decrypt* as-is.

The only non-trivial aspect of this transformation is the security reduction, since the HIBE adversary can make key-reveal queries on identity-vectors whereas the IBBE adversary can only ask for keys of "top level" single identities. We handle this difference by having the reduction algorithm generate decryption keys differently than is done in the system, which is where we need the distribution requirement of key randomization.

3.1 The Transformation

Let $\mathcal{E} = (\mathsf{Setup}, \mathsf{KeyGen}^*, \mathsf{KeyDerive}, \mathsf{KEM}, \mathsf{Decrypt}^*)$ be a key-randomizable IBBE system, and we assume that we have a "matching" collision resistant hash function H that can hash identity-vectors into the identity space of $\mathcal{E}$.[4] We use H to hash identity vectors in the HIBE system into identity sets for $\mathcal{E}$ by setting:

$$\boldsymbol{H}(\mathsf{ID}_1, \ldots, \mathsf{ID}_i) \stackrel{\text{def}}{=} \{H(\mathsf{ID}_1),\ H(\mathsf{ID}_1, \mathsf{ID}_2),\ \ldots,\ H(\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_i)\}$$

Note that short of finding collisions in H, we can only get $H(\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_i) \in \boldsymbol{H}(\mathbf{ID}')$ if $(\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_i)$ is a prefix of $\mathbf{ID}'$. Then we construct a HIBE system as follows:[5]

$\mathsf{HIBE.Setup}(\lambda, \ell)$: Set $(SK_0, PK_0) \leftarrow \mathcal{E}.\mathsf{Setup}(\lambda, \ell)$. Output SK and PK, which are the same as SK_0 and PK_0, except that each includes a description of the collision-resistant hash function H as above.

$\mathsf{HIBE.KeyGen}(PK, SK, \mathbf{ID})$: Set $S \leftarrow \boldsymbol{H}(\mathbf{ID})$ (as above), compute $K_S \leftarrow \mathcal{E}.\mathsf{KeyGen}^*(PK_0, SK_0, S)$ and output $K_{\mathbf{ID}} = K_S$.

$\mathsf{HIBE.KeyDerive}(PK, \mathbf{ID}, K_{\mathbf{ID}}, \mathbf{ID}')$: Set $S \leftarrow \boldsymbol{H}(\mathbf{ID})$ and $S' \leftarrow \boldsymbol{H}(\mathbf{ID}')$, and note that $S \subseteq S'$ since $\mathbf{ID}$ is a prefix of $\mathbf{ID}'$. Also let $K_S = K_{\mathbf{ID}}$, compute $K_{S'} \leftarrow \mathcal{E}.\mathsf{KeyDerive}(PK_0, S, K_S, S')$ and output $K_{\mathbf{ID}'} = K_{S'}$.

$\mathsf{HIBE.KEM}(PK, S)$: Set $S \leftarrow \boldsymbol{H}(\mathbf{ID})$, compute $(K, C) \leftarrow \mathcal{E}.\mathsf{KEM}(PK_0, S)$ and output (K, C).

$\mathsf{HIBE.Decrypt}(PK, C, \mathbf{ID}, K_{\mathbf{ID}})$: Set $S \leftarrow \boldsymbol{H}(\mathbf{ID})$, $K_S = K_{\mathbf{ID}}$, and return $\mathcal{E}.\mathsf{Decrypt}^*(PK_0, C, S, K_S)$.

[4] In our IBBE system from Section 5, the identity space is $\mathbb{Z}_q$ for a large prime q, so "matching" a hash function is not a problem.

[5] Note that this transformation is completely black box; in particular, it does not depend on whether or not the IBBE system uses a bilinear map.

3.2 A Tight Reduction

Theorem 1. *Suppose that there exists a HIBE adversary $\mathcal{A}$ that breaks CPA security (resp. CCA security) of the HIBE construction with advantage ϵ. Then, there exists an IBBE adversary $\mathcal{B}$ and a collision finder $\mathcal{B}'$, both running in about the same time as $\mathcal{A}$, such that $\mathcal{B}'$ finds a hash function collision with some probability ϵ' and $\mathcal{B}$ breaks the CPA security (resp. CCA security) of the underlying KR-IBBE system $\mathcal{E}$ with advantage $\epsilon - \epsilon'$.*

Proof. $\mathcal{B}$ attacks receives a public key PK_0 from the challenger. It sets PK to be PK_0 with the hash function H, and sends PK to $\mathcal{A}$.

When $\mathcal{A}$ requests a HIBE key for identity vector $\mathsf{\mathbf{ID}} = (\mathsf{ID}_1, \dots, \mathsf{ID}_t)$, $\mathcal{B}$ sets $S \leftarrow \boldsymbol{H}(\mathsf{\mathbf{ID}}) = \{\mathsf{ID}'_1, \dots, \mathsf{ID}'_t\}$ (with $\mathsf{ID}'_i = H(\mathsf{ID}_1, \dots, \mathsf{ID}_i)$). $\mathcal{B}$ asks the IBBE challenger for the decryption key $K_{\mathsf{ID}'_t}$, corresponding to the last identity $\mathsf{ID}'_t = H(\mathsf{ID}_1, \dots, \mathsf{ID}_t)$. If $t = 1$, it forward K_{ID_t} to $\mathcal{A}$; else, it returns to $\mathcal{A}$ the decryption key $K_{\mathsf{\mathbf{ID}}} = K_S \leftarrow \mathcal{E}.\mathsf{KeyDerive}(PK_0, \{\mathsf{ID}'_t\}, K_{\mathsf{ID}'_t}, S)$.

In the CCA case, $\mathcal{B}$ handles decryption queries in the obvious way (forwarding, after replacing $\mathsf{\mathbf{ID}}$ with $S = \boldsymbol{H}(\mathsf{\mathbf{ID}})$).

When $\mathcal{A}$ asks for the challenge ciphertext with respect to identity vector $\mathsf{\mathbf{ID}}^* = (\mathsf{ID}^*_1, \dots, \mathsf{ID}^*_t)$, $\mathcal{B}$ again sets $S^* \leftarrow \boldsymbol{H}(\mathsf{\mathbf{ID}}^*)$, makes a challenge query to the IBBE challenger with target identity-set S^* (unless there is a hash collision, see below), and return to $\mathcal{A}$ whatever it got back from its challenger.

Finally, when $\mathcal{A}$ halts with a guess bit σ', then $\mathcal{B}$ forward the same bit σ' to its IBBE challenger.

The collision finder $\mathcal{B}'$ proceeds just like $\mathcal{B}$, except that it saves all the queries that $\mathcal{A}$ makes and looks for collisions of the form $H(\mathsf{ID}_1, \dots, \mathsf{ID}_i) = H(\mathsf{ID}'_1, \dots, \mathsf{ID}'_j)$ (where $(\mathsf{ID}_1, \dots, \mathsf{ID}_i)$, $(\mathsf{ID}'_1, \dots, \mathsf{ID}'_j)$ are prefixes of queries made by $\mathcal{A}$). Denote by ϵ' the probability that $\mathcal{B}'$ finds any such collision.

We claim that $\mathcal{B}$'s simulation appears to $\mathcal{A}$ to be perfectly distributed as long as no collisions occur. This is true, because the distribution requirement on $\mathcal{E}.\mathsf{KeyDerive}$ implies that the HIBE key that $\mathcal{B}$ gets for identity vector $\mathsf{\mathbf{ID}} = (\mathsf{ID}_1, \dots, \mathsf{ID}_t)$ has almost the same distribution, regardless of whether it is obtained by KeyGen (as it would in an actual attack on the HIBE system) or by $\mathsf{KeyDerive}$ from $\{\mathsf{ID}'_t\}$ (as computed by $\mathcal{B}$). Hence, the advantage of $\mathcal{B}$ is at least $\epsilon - \epsilon'$. □

Remark. Observe that the proof above only uses the fact that $\mathsf{KeyDerive}$ has the same distribution as KeyGen when it starts from a singleton key $K_{\{\mathsf{ID}'_t\}}$. This is because we are using here the simplistic security definition for HIBE that does not address the delegation issue. To prove the more elaborate notion of Shi and Waters [19] we would need to use the full power of the distribution requirement.

4 Notations and Preliminaries

We now introduce notations and hardness assumption that are used to establish our key-randomizable IBBE in Section 5. We denote the set of integers from m to n (inclusive) by $[m, n]$. We denote polynomials by uppercase letters in San-serif font, for example P, Q, T, etc. We use the following simple fact about polynomials:

Lemma 1. *For any polynomial $\mathsf{P}(x)$ and any scalar a, $\mathsf{P}(x) - \mathsf{P}(a)$ is divisible by $x - a$. In other words, $\frac{\mathsf{P}(x)-\mathsf{P}(a)}{x-a}$ is a polynomial (without denominator) of degree $deg(\mathsf{P}) - 1$.*

4.1 Bilinear maps and our additive notations

Our system (and its security proof) make heavy use of linear algebra. We therefore use additive notations for all the groups that are involved in the system. Specifically, we use $\mathbb{Z}_q$ — the field of integers modulo a prime q — as our base scalar field, and we have two order-q groups that we call the *source group* $\mathbb{G}$ and *target group* $\mathbb{G}_T$, both of which can be viewed as vector spaces over $\mathbb{Z}_q$.

Throughout the writeup we denote elements of the source group with a hat over lowercase letters (e.g., $\hat{a}, \hat{b}$, etc.) and elements of the target group with a tilde ($\tilde{a}, \tilde{b}$, etc.). Scalars will be denoted with no decorations (e.g., a, b, and sometimes also τ, ρ, etc.)

We will make use of an efficiently computable bilinear map from the source group to the target group $\mathsf{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, such that for any two source-group elements $\hat{a}, \hat{b} \in \mathbb{G}$ and any two scalars $u, v \in \mathbb{Z}_q$ it holds that

$$\mathsf{e}(u \cdot \hat{a}, v \cdot \hat{b}) = uv \cdot \mathsf{e}(\hat{a}, \hat{b})$$

The neutral elements in the groups $\mathbb{G}, \mathbb{G}_T$ are denoted by $\hat{0}$, $\tilde{0}$, respectively. We also denote by $\hat{1}$ some fixed generator in $\mathbb{G}$, which we consider to be part of the description of $\mathbb{G}$. We require that the mapping e is non-trivial, which means that $\mathsf{e}(\hat{1}, \hat{1})$ is a generator in $\mathbb{G}_T$, and we denote this generator by $\tilde{1} = \mathsf{e}(\hat{1}, \hat{1})$.

More generally, for a scalar $a \in \mathbb{Z}_q$, we denote the source-group element $a \cdot \hat{1}$ by $\hat{a}$, and the target-group element $a \cdot \tilde{1} = \mathsf{e}(\hat{a}, \hat{1})$ by $\tilde{a}$. Conversely, for an element $\hat{a} \in \mathbb{G}$, its discrete-logarithm based $\hat{1}$ is denoted $a \in \mathbb{Z}_q$. (Readers who are used to multiplicative notations may find it easier to think of $\hat{a}$, $\tilde{a}$ as denoting "a in the exponent" in the appropriate groups.) Note also that in these notations, the discrete-logarithm of $\hat{a}$ with respect to $\hat{b}$ is just their "ratio" $\hat{a}/\hat{b}$, which is a scalar.

With these notations, we usually omit the map e altogether, and simply denote it as a "product" of two source-group elements:

$$\hat{a} \cdot \hat{b} \;\stackrel{\text{def}}{=}\; \mathsf{e}(\hat{a}, \hat{b}) \;=\; \widetilde{ab} \;\in\; \mathbb{G}_T$$

Note that the bi-linearity of e looks in these notations just like the natural commutative property of products $\widehat{ua} \cdot \widehat{vb} = uv \cdot \widetilde{ab}$.

Below we slightly abuse notations to denote "powers of group elements": If $\hat{a}$ is a group element with discrete-logarithm a, then we denote $\hat{a}^i \stackrel{\text{def}}{=} a^i \cdot \hat{1}$ and we call $\hat{a}^i$ the i'th power of $\hat{a}$. [6]

Vectors and matrices. We extend our notations to vectors and matrices: A vector of scalars is denoted with no decoration $\boldsymbol{a} = [a_1, a_2, \ldots, a_n]$, a vector of source-group elements denoted with a hat, $\hat{\boldsymbol{a}} = [\hat{a}_1, \hat{a}_2, \ldots, \hat{a}_n]$, and a vector of source-group elements denoted with a tilde $\tilde{\boldsymbol{a}} = [\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n]$. All these vectors are considered row vectors.

Matrices are denoted by uppercase letters, e.g., A for a matrix of scalars, $\hat{A}$ for a matrix of source-group elements, and $\tilde{A}$ for a matrix of target-group elements. We denote the i'th row of A by A_i, the sub-matrix consisting of rows i, j, k by $A_{i,j,k}$, and the sub-matrix consisting of rows i through j is denoted $A_{i..j}$. As usual, the transposed matrix of A is denoted A^t.

We denote by $\mathsf{span}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ the linear space that is spanned by the vectors $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$, and also use the same notation to denote the uniform distribution over this space. For example, we use $\hat{\boldsymbol{u}} \leftarrow \hat{\boldsymbol{w}} + \mathsf{span}(\hat{A}_{1,2,4})$ as a shorthand for the process of choosing three random scalars $a, b, c \in_R \mathbb{Z}_p$ and setting $\hat{\boldsymbol{u}} \leftarrow \hat{\boldsymbol{w}} + a\hat{A}_1 + b\hat{A}_2 + c\hat{A}_4$.

[6] This abuse of notation may take some getting used to: notice that the a's themselves should be thought of as being "in the exponent." In multiplicative notation, this power of $\hat{a}$ would be denoted as something like $g^{(a^i)}$.

Inner and outer-products. For vectors $\boldsymbol{a}, \boldsymbol{b}$, we denote their inner product by $\langle \boldsymbol{a}, \boldsymbol{b} \rangle \stackrel{\text{def}}{=} \sum_i a_i b_i$. We use the same inner-product notations also for vectors of source-group elements, namely:

$$\left\langle \boldsymbol{a}, \hat{\boldsymbol{b}} \right\rangle = \left\langle \hat{\boldsymbol{b}}, \boldsymbol{a} \right\rangle \stackrel{\text{def}}{=} \sum_i a_i \hat{b}_i = \langle \boldsymbol{a}, \boldsymbol{b} \rangle \cdot \hat{1} \in \mathbb{G}, \text{ and } \left\langle \hat{\boldsymbol{a}}, \hat{\boldsymbol{b}} \right\rangle \stackrel{\text{def}}{=} \sum_i \hat{a}_i \hat{b}_i = \sum_i \mathsf{e}(\hat{a}_i, \hat{b}_i) = \langle \boldsymbol{a}, \boldsymbol{b} \rangle \cdot \tilde{1} \in \mathbb{G}_T$$

It is easy to check that all the commutative, associative, and distributive properties of inner products hold for both scalars and group elements.

Similar notations apply to matrix multiplication, for either scalar matrices or group-element matrices. For example, if A is an $\ell \times m$ scalar matrix and $\hat{B}$ is an $m \times n$ matrix of source-group elements, then $A\hat{B} \in \mathbb{G}[\ell \times n]$ is a matrix of source-group elements whose i, j element is the inner product of the i'th row of A by the j'th column of $\hat{B}$. We also use $\boldsymbol{a} \times \boldsymbol{b}$ to denote the outer product of two vectors. Namely, the outer product of the m-vector $\boldsymbol{a}$ by the n-vector $\boldsymbol{b}$ is the $m \times n$ matrix obtained as the matrix product of the $m \times 1$ matrix $\boldsymbol{a}^t$ by the $1 \times n$ matrix $\boldsymbol{b}$. The same notation applies to vectors of group elements.

Linear algebra. All the standard concepts from linear algebra behave just the same with either scalars or group elements. For example, if $\hat{A} \in \mathbb{G}[n \times n]$ is a square matrix of source-group elements and A is the matrix of the discrete logarithm of all the elements in $\hat{A}$ (with respect to the fixed generator $\hat{1}$), then the inverse of $\hat{A}$ is $\hat{A}^{-1} = A^{-1} \cdot \hat{1}$. (Equivalently, the inverse of $\hat{A}$ is the unique matrix $\hat{B}$ such that $\hat{A} \cdot \hat{B} = \tilde{I}$.) Similarly, the rank of a scalar matrix is defined as usual, and the rank of a matrix of group elements is defined as the rank of their discrete-logarithm matrix.

4.2 The BDHE-Set assumption

The BDHE-Set assumption (used also in [15]) is a parameterized generalization of the t-BDHI problem from [1].[7] Recall that a t-BDHI adversary is given $t+1$ powers of a random source-group element, $\hat{1}, \hat{a}, \hat{a}^2, \ldots \hat{a}^t$, and it needs to distinguish the target-group element $\tilde{a}^{-1}$ from random.

An instance of the BDHE-Set assumption is parameterized by a set of integers $\mathcal{S} \subset \mathbb{Z}$ and another "target integer" m. The BDHE-Set adversary is given some powers of a random source-group element, $\{\hat{a}^i \ : \ i \in \mathcal{S}\}$, and it (roughly) needs to distinguish the target-group element $\tilde{a}^m$ from random. Denoting $\mathcal{S} +_q \mathcal{S} \stackrel{\text{def}}{=} \{i + j \bmod \lambda(q) \ : i, j \in \mathcal{S}\}$, where $\lambda(q)$ is the order of elements modulo q, it is easy to see that if $\mathbb{G}$ is an order-q bilinear-map group and $m \in \mathcal{S} +_q \mathcal{S}$ then the problem is easy: Just choose some $i, j \in \mathcal{S}$ such that $i + j = m \bmod \lambda(q)$ and compute the bilinear map

$$\mathsf{e}(\hat{a}^i, \hat{a}^j) \ = \ \hat{a}^i \cdot \hat{a}^j \ = \ \tilde{a}^{i+j} \ = \ \tilde{a}^m$$

However, when $m \notin \mathcal{S} +_q \mathcal{S}$ then there does not seem to be an easy way of distinguishing $\tilde{a}^m$ from random given the source-group elements $\{\hat{a}^i \ : \ i \in \mathcal{S}\}$. The formal BDHE-Set assumption below is somewhat stronger, however, giving the adversary not the target group element $\tilde{a}^m$ itself, but rather two random source group elements whose product is $\tilde{a}^m$. Even so, this may be a reasonable assumption to make.

Definition 3 (Decision BDHE-Set). *Fix a prime number q, a set of integers $\mathcal{S}$ and another integer $m \notin \mathcal{S} +_q \mathcal{S}$. Also fix two order-$q$ groups $\mathbb{G}$ and $\mathbb{G}_T$, admitting a non-trivial, efficiently computable bilinear map $\mathsf{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$.*

[7] This assumption is called q-BDHI in [1], but we use q as our group order.

The $(\mathcal{S}, m)$-BDHE-Set problem with respect to $\mathbb{G}$ and $\mathbb{G}_T$ consists of the following experiment: Choose at random a scalar $a \in_R \mathbb{Z}_q^*$ and a bit $\sigma \in_R \{0,1\}$. If $\sigma = 0$ then choose two random scalars $z_1, z_2 \in_R \mathbb{Z}_q^*$, and if $\sigma = 1$ then choose a random scalar $z_1 \in_R \mathbb{Z}_q^*$ and set $z_2 \leftarrow a/z_1 \bmod q$. The BDHE-Set adversary gets as input $\hat{a}^i = a^i \cdot \hat{1}$ for all $i \in \mathcal{S}$ and also $\hat{z}_1, \hat{z}_2$, and its goal is to guess the bit σ. The advantage of an adversary A is defined as

$$\mathsf{AdvBDHE}_A^{\mathcal{S},m}(\mathbb{G}, \mathbb{G}_T) \stackrel{\text{def}}{=} \Pr\left[a, z_1 \in_R \mathbb{Z}_q^*,\ z_2 \leftarrow \frac{a}{z_1},\ A\left(\{\hat{a}^i : i \in \mathcal{S}\}, \hat{z}_1, \hat{z}_2\right) \Rightarrow 1\right] - \Pr\left[a, z_1, z_2 \in_R \mathbb{Z}_q^*,\ A\left(\{\hat{a}^i : i \in \mathcal{S}\}, \hat{z}_1, \hat{z}_2\right) \Rightarrow 1\right]$$

Informally, the asymptotic Decision BDHE-Set assumption states that for any $m \notin \mathcal{S}+\mathcal{S}$ and a large enough prime q, efficient adversaries (that work in time $\mathrm{poly}(|\mathcal{S}|, \log q)$ only have insignificant advantage in the experiment from above. Making this formal is rather straightforward (though getting the quantification right takes some care).

Jumping ahead, for our system we use the assumption above with the target integer $m = -1$ and the set $\mathcal{S}$ defined as:

$$\begin{aligned} \mathcal{S} \;=\; & [-2h-2\ell,\ -2h-\ell-2] \;\cup\; [-h-\ell,\ -\ell-1] \\ & \cup\; [0, \ell-1] \;\cup\; [h+\ell, 2h+\ell] \;\cup\; [2h+2\ell, 3h+2\ell+1] \end{aligned} \tag{1}$$

where ℓ is the depth of the identity-hierarchy of the system and h is some other parameter. (Specifically, if q^* is a bound on the number of queries then $h = q^* + \ell + 2$.) This set $\mathcal{S}$ is depicted in Figure 1, which makes it clear that $\mathcal{S}+\mathcal{S}$ indeed does not include the target integer $m = -1$.

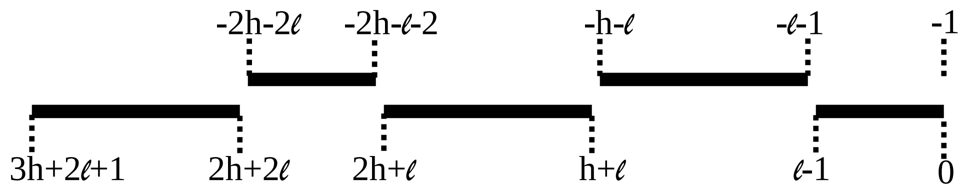

Fig. 1. A graphic depiction of the set $\mathcal{S}$, "folded" around the point $m/2 = -1/2$.

4.3 The Linear Assumption

The decision linear assumption, first defined in [4], states (in our additive notations) that given the six source group elements $\hat{a}$,$\hat{b}$,$\hat{c}$,$\hat{d}$,$\hat{e}$,$\hat{f}$, it is hard to distinguish the case where these elements are completely random from the case where they are chosen at random subject to the condition $\hat{f}/\hat{c} = \hat{e}/\hat{b} + \hat{d}/\hat{a}$. (I.e., the discrete logarithm of f relative to c is the sum of the discrete logarithm of e relative to b and the discrete logarithm of d relative to a.) Note that this assumption is equivalent to saying that given the matrix of group elements

$$M \;=\; \begin{pmatrix} \hat{a} & \hat{0} & \hat{c} \\ \hat{0} & \hat{b} & \hat{c} \\ \hat{d} & \hat{e} & \hat{f} \end{pmatrix}$$

it is hard to decide if this matrix is invertible or has rank two. In this work we use a slightly weaker variant of this assumption, only assuming that given a 3×3 matrix of source-group elements, it is hard to distinguish the case where this is a random invertible matrix from the case where it is a random rank-two matrix. (This weaker assumption is implied both by the standard linear assumption and by our BDHE-Set assumption, but we make it a separate assumption just to make the exposition of our security-proof easier.)

5 A Key-Randomizable IBBE system

Our system operates in prime-order bilinear-map groups. In the description below we assume that these order-q groups are fixed "once and for all" and everyone knows their description. (An alternative description will include the group-generation as part of the Setup procedure.) We also fix the hierarchy-depth of the system to some integer ℓ.

The identity space of the system is the scalar field $\mathbb{Z}_q$, except that we have ℓ "forbidden identities" within this range: $\ell - 1$ of them are arbitrary (and we set them to be $0, 1, \ldots, \ell - 2$), and the last one is a random scalar a that is chosen during Setup (see below).

Setup: Choose three random scalars $a, b, s \in Z_q$ and a random invertible matrix $A \in G[7 \times 7]$, and set $\hat{B} = (\hat{A}^{-1})^t$. We note that the system only uses the top four rows of $\hat{A}$ and five rows of $\hat{B}$. The seventh dimension is only used in the security proof. Below we denote by $\boldsymbol{a}_i$ the vector $\boldsymbol{a}_i \stackrel{\text{def}}{=} [1\ a\ a^2 \ldots\ a^i]$.

- The master secret key is $SK = (\hat{B}_{1..6}, s, \boldsymbol{a}_\ell)$.
- The public key consists of three parts, $PK = (PK_1, PK_2, PK_3)$ with PK_1 consisting of a target-group element that is used to compute the KEM key, PK_2 consisting of multiples of the rows of $\hat{A}$ that are used to compute the ciphertext, and PK_3 consisting of multiples of the rows of $\hat{B}$ that are used only for key randomization. Specifically we have

$$
\begin{aligned}
PK_1 &= a^{\ell-1}\tilde{s} \qquad (2)\\
PK_2 &= \Big\{ \underbrace{\{a^i \hat{A}_1 :\ i = 0, \ldots, \ell\}}_{\boldsymbol{a}_\ell \times \hat{A}_1},\ \ s\hat{A}_2,\ \ \underbrace{\{a^i \hat{A}_3 :\ i = 0, \ldots, \ell-1\}}_{\boldsymbol{a}_{\ell-1} \times \hat{A}_3},\ \ \hat{A}_4 \Big\}\\
PK_3 &= \Big\{ bs\hat{B}_1,\ \ abs\hat{B}_1,\ \ \hat{B}_5,\ \ \hat{B}_6,\\
&\qquad \underbrace{\{a^i b\hat{B}_1 :\ i = 0, \ldots, \ell\}}_{b(\boldsymbol{a}_\ell \times \hat{B}_1)},\ \ \underbrace{\{a^i b\hat{B}_2 :\ i = 0, \ldots, \ell\}}_{b(\boldsymbol{a}_\ell \times \hat{B}_2)},\ \ \underbrace{\{a^i b\hat{B}_3 :\ i = 0, \ldots, \ell+1\}}_{b(\boldsymbol{a}_{\ell+1} \times \hat{B}_3)} \Big\}
\end{aligned}
$$

KeyGen (PK, SK, ID): Choose a key of $3\ell - 3$ 7-dimensional vectors of source-group elements as follows: Pick at random $r_{\mathsf{ID}} \in Z_q$ and set $\hat{K}_{\mathsf{ID}} = (\hat{\boldsymbol{u}}_{\mathsf{ID}}, \hat{V}_{\mathsf{ID}}, \hat{W}_{\mathsf{ID}}, \hat{X}_{\mathsf{ID}}, \hat{\boldsymbol{y}}_{\mathsf{ID}})$, where

$$\begin{aligned}
\hat{\boldsymbol{u}}_{\mathsf{ID}} &= \frac{s - r_{\mathsf{ID}}}{a - \mathsf{ID}} \hat{B}_1, \\
\hat{V}_{\mathsf{ID}} &= r_{\mathsf{ID}}(\boldsymbol{a}_{\ell-2} \times \hat{B}_1) \quad \left(= \{r_{\mathsf{ID}} a^i \hat{B}_1 \ : \ i = 0, \ldots, \ell-2\}\right), \\
\hat{W}_{\mathsf{ID}} &= \boldsymbol{a}_{\ell-1} \times \hat{B}_2 \qquad \left(= \{a^i \hat{B}_2 \ : \ i = 0, \ldots, \ell-1\}\right), \\
\hat{X}_{\mathsf{ID}} &= r_{\mathsf{ID}}(\boldsymbol{a}_{\ell-2} \times \hat{B}_3) \quad \left(= \{r_{\mathsf{ID}} a^i \hat{B}_3 \ : \ i = 0, \ldots, \ell-2\}\right), \\
\hat{\boldsymbol{y}}_{\mathsf{ID}} &= r_{\mathsf{ID}} a^{\ell-1} \hat{B}_3 + \mathsf{span}(\hat{B}_{5,6})
\end{aligned} \tag{3}$$

Note that the $\hat{W}_{\mathsf{ID}}$ component is the same for all identities (so it really belongs in the public key). It is included in the secret key only for the purpose of the key-randomization procedure below.

KEM (PK, S): If $|S| < \ell$ then add to S the first $\ell - |S|$ of the "forbidden identities" $0, 1, \ldots$. Denote the resulting ℓ identities by $\{\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_\ell\}$.

- Set the monic degree-ℓ polynomial $\mathsf{P}(x) \stackrel{\text{def}}{=} \prod_{i=1}^{\ell}(x - \mathsf{ID}_i)$, let $p_0, \ldots, p_\ell$ be the coefficients of P and denote $\boldsymbol{p} \stackrel{\text{def}}{=} [p_0 \ \ldots \ p_\ell]$ (so $\mathsf{P}(a) = \langle \boldsymbol{p}, \boldsymbol{a}_\ell \rangle$).
- Choose at random $f_0, \ldots, f_{\ell-1} \in Z_q$ and denote $\boldsymbol{f} \stackrel{\text{def}}{=} [f_0 \ f_1 \ \ldots \ f_{\ell-1}]$ and $\mathsf{F}(x) \stackrel{\text{def}}{=} \sum_{i=0}^{\ell-1} f_i x^i$. Make sure that $\mathsf{F}(\mathsf{ID}_i) \neq 0$ for all $i = 1, \ldots, \ell$ (otherwise re-choose F until this condition holds).
- Choose a random scalar $t \in Z_q$.
- Output the ciphertext containing the polynomial F and the vector

$$\hat{\boldsymbol{c}} = t\Big(\underbrace{\mathsf{P}(a)\hat{A}_1}_{\boldsymbol{p}(\boldsymbol{a}_\ell \times \hat{A}_1)} \ + \ s\hat{A}_2 \ + \ \underbrace{\mathsf{F}(a)\hat{A}_3}_{\boldsymbol{f}(\boldsymbol{a}_{\ell-1} \times \hat{A}_3)} \Big) + \mathsf{span}(\hat{A}_4) \tag{4}$$

The implied KEM key is the target-group element $\tilde{k} = t \cdot PK_1 = a^{\ell-1} t \tilde{s}$.

Remark. Note that the ciphertext include seven source group elements and ℓ scalars (to specify F). The ciphertext size can be reduced in a particular way, so that when encrypting to a set S of size $m < \ell$ we only have m scalars in the ciphertext: Instead of choosing F completely at random, we impose the condition that $\mathsf{F}(ID) = 1$ for each of the "forbidden identities" that were added to S. This way, the encryptor can specify F using only the m scalars $\mathsf{F}(\mathsf{ID}_i)$ for all $\mathsf{ID}_i \in S$. This optimization requires a small change to the proof of security, see remark at the end of Section 6. We also note that we can get a constant-size ciphertext by moving to the random-oracle model: the encryptor just sends some nonce, and F is determined by applying the random oracle to this nonce.

Decrypt $(PK, (\mathsf{F}, \hat{\boldsymbol{c}}), S, \mathsf{ID}, \hat{K}_{\mathsf{ID}})$, where $\mathsf{ID} \in S$. If $|S| < \ell$ then add to S the first $\ell - |S|$ of the "forbidden identities" $0, 1, \ldots$. Denote the resulting ℓ identities by $\{\mathsf{ID}_1, \mathsf{ID}_2, \ldots, \mathsf{ID}_\ell\}$. Parse the key as $\hat{K}_{\mathsf{ID}} = (\hat{\boldsymbol{u}}_{\mathsf{ID}}, \hat{V}_{\mathsf{ID}}, \hat{W}_{\mathsf{ID}}, \hat{X}_{\mathsf{ID}}, \hat{\boldsymbol{y}}_{\mathsf{ID}})$, recalculate the monic ℓ-degree polynomial $\mathsf{P}(x) = \prod_{i=1}^{\ell}(x - \mathsf{ID}_i)$, and do the following:

- Set $\mathsf{Q}_{\mathsf{ID}}(x) \stackrel{\text{def}}{=} \dfrac{\mathsf{P}(x)}{x-\mathsf{ID}}$ and $\mathsf{Q'}_{\mathsf{ID}}(x) = \mathsf{Q}_{\mathsf{ID}}(x) - a^{\ell-1}$. (That is, $\mathsf{Q'}$ is the polynomial Q without the top coefficient of $1 \cdot x^{\ell-1}$.) Denote the coefficient vector of $\mathsf{Q'}_{\mathsf{ID}}$ by $\boldsymbol{q'}_{\mathsf{ID}} = [q_0\ q_1\ \dots\ q_{\ell-2}]$.
- Set $\mathsf{G}_{\mathsf{ID}}(x) \stackrel{\text{def}}{=} \dfrac{\mathsf{F}(x)-\mathsf{F}(\mathsf{ID})}{x-\mathsf{ID}}$ and denote the coefficient vector of G_{ID} by $\boldsymbol{g}_{\mathsf{ID}} = [g_0\ g_1\ \dots\ g_{\ell-2}]$.
- Set

$$\hat{\boldsymbol{d}}_{\mathsf{ID}} = \hat{\boldsymbol{u}}_{\mathsf{ID}} - \boldsymbol{q'}_{\mathsf{ID}} \cdot \hat{W}_{\mathsf{ID}} - \frac{\boldsymbol{g}_{\mathsf{ID}} \cdot \hat{V}_{\mathsf{ID}} - \boldsymbol{q'}_{\mathsf{ID}} \cdot \hat{X}_{\mathsf{ID}} - \hat{\boldsymbol{y}}_{\mathsf{ID}}}{\mathsf{F}(\mathsf{ID})} \tag{5}$$

Finally, recover the KEM key as $\tilde{k} = \left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}} \right\rangle$.

5.1 Correctness

To argue correctness, we can rewrite

$$\begin{aligned}
\hat{\boldsymbol{d}}_{\mathsf{ID}} &= \underbrace{\frac{s-r_{\mathsf{ID}}}{a-\mathsf{ID}}\hat{B}_1}_{\hat{\boldsymbol{u}}_{\mathsf{ID}}} \quad - \quad \boldsymbol{q'}_{\mathsf{ID}} \cdot (\underbrace{\boldsymbol{a}_{\ell-2} \times \hat{B}_2}_{\hat{W}_{\mathsf{ID}}}) \\
&\quad -\Bigg(\boldsymbol{g}_{\mathsf{ID}} \cdot (\underbrace{r_{\mathsf{ID}}\boldsymbol{a}_{\ell-2} \times \hat{B}_1}_{\hat{V}_{\mathsf{ID}}}) - \boldsymbol{q'}_{\mathsf{ID}} \cdot (\underbrace{r_{\mathsf{ID}}\boldsymbol{a}_{\ell-2} \times \hat{B}_3}_{\hat{X}_{\mathsf{ID}}}) - \underbrace{(r_{\mathsf{ID}}a^{\ell-1}\hat{B}_3 + \mathsf{span}(\hat{B}_{5,6}))}_{\hat{\boldsymbol{y}}_{\mathsf{ID}}}\Bigg)/\mathsf{F}(\mathsf{ID}) \\
&= \frac{s-r_{\mathsf{ID}}}{a-\mathsf{ID}}\hat{B}_1 \quad - \quad \left\langle \boldsymbol{q'}_{\mathsf{ID}}, \boldsymbol{a}_{\ell-2} \right\rangle \hat{B}_2 \\
&\quad -\frac{r_{\mathsf{ID}}}{\mathsf{F}(\mathsf{ID})}\Bigg(\left\langle \boldsymbol{g}_{\mathsf{ID}}, \boldsymbol{a}_{\ell-2} \right\rangle \hat{B}_1 - \left(\left\langle \boldsymbol{q'}_{\mathsf{ID}}, \boldsymbol{a}_{\ell-2} \right\rangle + a^{\ell-1}\right)\hat{B}_3 - \mathsf{span}(\hat{B}_{5,6})\Bigg) \\
&= \left(\frac{s-r_{\mathsf{ID}}}{a-\mathsf{ID}} - \frac{r_{\mathsf{ID}}\mathsf{G}_{\mathsf{ID}}(a)}{\mathsf{F}(\mathsf{ID})}\right)\hat{B}_1 - (\mathsf{Q}_{\mathsf{ID}}(a) - a^{\ell-1})\hat{B}_2 + \frac{r_{\mathsf{ID}}}{\mathsf{F}(\mathsf{ID})}\Big(\mathsf{Q}_{\mathsf{ID}}(a)\hat{B}_3 + \mathsf{span}(\hat{B}_{5,6})\Big)
\end{aligned}$$

Further developing the coefficient of $\hat{B}_1$ we get

$$\begin{aligned}
\left(\frac{s-r_{\mathsf{ID}}}{a-\mathsf{ID}} - \frac{r_{\mathsf{ID}}\mathsf{G}_{\mathsf{ID}}(a)}{\mathsf{F}(\mathsf{ID})}\right) &= \frac{\mathsf{F}(\mathsf{ID})(s-r_{\mathsf{ID}}) - r_{\mathsf{ID}}\mathsf{G}_{\mathsf{ID}}(a)(a-\mathsf{ID})}{\mathsf{F}(\mathsf{ID})(a-\mathsf{ID})} \\
&= \frac{\mathsf{F}(\mathsf{ID})(s-r_{\mathsf{ID}}) - r_{\mathsf{ID}}(\mathsf{F}(a)-\mathsf{F}(\mathsf{ID}))}{\mathsf{F}(\mathsf{ID})(a-\mathsf{ID})} \\
&= \frac{s \cdot \mathsf{F}(\mathsf{ID}) - r_{\mathsf{ID}} \cdot \mathsf{F}(a)}{\mathsf{F}(\mathsf{ID})(a-\mathsf{ID})}
\end{aligned}$$

Examining the inner-product of $\hat{\boldsymbol{c}}$ with $\hat{\boldsymbol{d}}_{\mathsf{ID}}$, we use the fact that $\left\langle \hat{A}_i, \hat{B}_j \right\rangle$ is either 0 (when $i \neq j$) or $\tilde{1}$ (when $i = j$). Hence the span's of $\hat{A}_4$ and of $\hat{B}_{5,6}$ drop out completely, and we are left with the product of the matching coefficients only:

$$\left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}} \right\rangle = \Bigg(\underbrace{t\mathsf{P}(a)\frac{s \cdot \mathsf{F}(\mathsf{ID}) - r_{\mathsf{ID}} \cdot \mathsf{F}(a)}{\mathsf{F}(\mathsf{ID})(a-\mathsf{ID})}}_{\text{coefficients of } \hat{A}_1,\hat{B}_1} - \underbrace{ts(\mathsf{Q}_{\mathsf{ID}}(a) - a^{\ell-1})}_{\text{coefficients of } \hat{A}_2,\hat{B}_2} + \underbrace{t\mathsf{F}(a)\frac{r_{\mathsf{ID}}}{\mathsf{F}(\mathsf{ID})}\mathsf{Q}_{\mathsf{ID}}(a)}_{\text{coefficients of } \hat{A}_3,\hat{B}_3}\Bigg) \cdot \tilde{1}$$

The first term in the parenthesis above can be simplified using $\mathsf{Q}_{\mathsf{ID}}(a) = \mathsf{P}(a)/(a - \mathsf{ID})$, so we get

$$\begin{aligned}
\left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}} \right\rangle &= t\Big(\mathsf{Q}_{\mathsf{ID}}(a)\frac{s \cdot \mathsf{F}(\mathsf{ID}) - r_{\mathsf{ID}} \cdot \mathsf{F}(a)}{\mathsf{F}(\mathsf{ID})} - s(\mathsf{Q}_{\mathsf{ID}}(a) - a^{\ell-1}) + \mathsf{F}(a)\frac{r_{\mathsf{ID}}}{\mathsf{F}(\mathsf{ID})}\mathsf{Q}_{\mathsf{ID}}(a)\Big) \cdot \tilde{1} \\
&= t\Big(\mathsf{Q}_{\mathsf{ID}}(a)s - \frac{r_{\mathsf{ID}}\mathsf{Q}_{\mathsf{ID}}(a)\mathsf{F}(a)}{\mathsf{F}(\mathsf{ID})} - \mathsf{Q}_{\mathsf{ID}}(a)s + a^{\ell-1}s + \frac{r_{\mathsf{ID}}\mathsf{Q}_{\mathsf{ID}}(a)\mathsf{F}(a)}{\mathsf{F}(\mathsf{ID})}\Big) \cdot \tilde{1} \\
&= a^{\ell-1}ts \cdot \tilde{1} \;=\; \tilde{k}
\end{aligned}$$

□

5.2 Key randomization

Our key-randomization follows Boyen's idea from [8], where the key for identity-set $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_m\}$ consists of m "shifted versions" of the keys, $r'_{\mathsf{ID}_1}\hat{K}_{\mathsf{ID}_1}, \ldots, r'_{\mathsf{ID}_n}\hat{K}_{\mathsf{ID}_m}$, such that $\sum_i r'_{\mathsf{ID}_i} = 1 \pmod q$. Namely, the augmented procedure $\mathsf{KeyGen}^*(PK, SK, S)$ uses the same KeyGen procedure from above m times to get

$$\hat{K}_{\mathsf{ID}_i} \leftarrow \mathsf{KeyGen}(PK, SK, \mathsf{ID}_i)$$

Then for $i = 1 \ldots m$ it chooses $r'_{\mathsf{ID}_i} \in Z_q$ at random subject to the constraint $\sum_i r'_{\mathsf{ID}_i} = 1 \pmod q$, and outputs the secret key

$$\hat{K}_S \;=\; [r'_{\mathsf{ID}_1}\hat{K}_{\mathsf{ID}_1}, \;\ldots,\; r'_{\mathsf{ID}_m}\hat{K}_{\mathsf{ID}_m}]$$

where $r'_{\mathsf{ID}_i}\hat{K}_{\mathsf{ID}_i}$ means multiplying all the elements in $\hat{K}_{\mathsf{ID}_i}$ by the scalar r'_{ID_i}. Below we call $\hat{K}_{\mathsf{ID}_i}$ the *singleton key* corresponding to ID_i, and $r'_{\mathsf{ID}_i}\hat{K}_{\mathsf{ID}_i}$ is the *shifted singleton key* for ID_i.[8] Note that for the special case $m = 1$, we have $r'_{\mathsf{ID}} = 1$, so KeyGen^* degenerates to the original KeyGen.

Extended decryption. The extended decryption procedure $\mathsf{Decrypt}^*$ is given a ciphertext $(F, \hat{c})$ together with a set of identities $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_m\}$ $(m \le \ell)$ and a matching decryption key $\hat{K}_S$. It parses the decryption key as $\hat{K}_S = [\hat{K}'_{\mathsf{ID}_1}, \ldots, \hat{K}'_{\mathsf{ID}_m}]$ where the $\hat{K}'_{\mathsf{ID}_i}$'s are shifted singleton keys. Namely we have $\hat{K}'_{\mathsf{ID}_i} = r'_{\mathsf{ID}_i}\hat{K}_{\mathsf{ID}_i}$ where the $\hat{K}_{\mathsf{ID}_i}$'s are singleton keys and $\sum_i r'_{\mathsf{ID}_i} = 1 \pmod q$. Then we use each shifted singleton key to produce $\hat{\boldsymbol{d}}'_{\mathsf{ID}_i}$ just as in Eq. (5), sets $\hat{\boldsymbol{d}}_S = \sum_i \hat{\boldsymbol{d}}'_{\mathsf{ID}_i}$, and recover $\tilde{k} = \left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_S \right\rangle$.

Correctness holds since the decryption process in linear: Denote by $\hat{\boldsymbol{d}}_{\mathsf{ID}_i}$ the vector that would have been obtained from the singleton key $\hat{K}_{\mathsf{ID}_i}$ using Eq. (5). Then on one hand decryption is linear so we have $\hat{\boldsymbol{d}}'_{\mathsf{ID}_i} = r'_{\mathsf{ID}_i}\hat{\boldsymbol{d}}'_{\mathsf{ID}_i}$. On the other hand by correctness of the basic decryption procedure we know that $\left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}_i} \right\rangle = \tilde{k}$. We therefore get

$$\left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_S \right\rangle \;=\; \sum_i \left\langle \hat{\boldsymbol{c}},\; \hat{\boldsymbol{d}}'_{\mathsf{ID}_i} \right\rangle \;=\; \sum_i \left\langle \hat{\boldsymbol{c}},\; r'_{\mathsf{ID}_i}\hat{\boldsymbol{d}}_{\mathsf{ID}_i} \right\rangle \;=\; \sum_i r'_{\mathsf{ID}_i}\left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}_i} \right\rangle \;=\; \sum_i r'_{\mathsf{ID}_i}\tilde{k} \;=\; \tilde{k}$$

[8] Note that $\hat{K}_S$ includes also the information of which shifted singleton key corresponds to what identity, so we denote it as an ordered tuple of shifted keys rather than a set.

Key derivation. Key-derivation uses Boyen's idea of reciprocal keys [8]. Namely, given the public key and any two identities ID_1 and ID_2, anyone can compute a pair of shifted singleton keys $\delta\hat{K}_{\mathsf{ID}_1}$ and $\delta\hat{K}_{\mathsf{ID}_2}$ for the same (unknown) scalar factor δ. The procedure for generating these reciprocal keys (which is used as a subroutine for key derivation) is as follows:

ReciprocalKeys $(PK, \mathsf{ID}_1, \mathsf{ID}_2)$: recall that the public key PK depends on the unknown scalars a, b, s (among other things).

- Choose at random $z \in Z_q$. The shifted singleton keys $\delta\hat{K}_{\mathsf{ID}_i}$ will have $\delta = bz(a - \mathsf{ID}_1)(a - \mathsf{ID}_2)$.
- Choose at random $r_1, r_2 \in Z_q$ (which will play the role of r_{ID_1} and r_{ID_2} in the reciprocal keys).
- Compute $\delta\hat{K}_{\mathsf{ID}_1}$ as

$$\begin{aligned}
\delta \cdot \frac{s - r_1}{a - \mathsf{ID}_1}\hat{B}_1 &= (abs - bs\mathsf{ID}_2 - abr_1 + br_1\mathsf{ID}_2)z\hat{B}_1 \\
\delta \cdot r_1(\boldsymbol{a}_{\ell-2} \times \hat{B}_1) &= br_1z(a^{i+2} - a^{i+1}(\mathsf{ID}_1 + \mathsf{ID}_2) + a^i\mathsf{ID}_1\mathsf{ID}_2)\hat{B}_1,\ i = 0, \ldots, \ell - 2 \\
\delta \cdot (\boldsymbol{a}_{\ell-2} \times \hat{B}_2) &= bz(a^{i+2} - a^{i+1}(\mathsf{ID}_1 + \mathsf{ID}_2) + a^i\mathsf{ID}_1\mathsf{ID}_2)\hat{B}_2,\ i = 0, \ldots, \ell - 2 \\
\delta \cdot r_1(\boldsymbol{a}_{\ell-2} \times \hat{B}_3) &= br_1z(a^{i+2} - a^{i+1}(\mathsf{ID}_1 + \mathsf{ID}_2) + a^i\mathsf{ID}_1\mathsf{ID}_2)\hat{B}_3,\ i = 0, \ldots, \ell - 2 \\
\delta \cdot \Big(r_1a^{\ell-1}\hat{B}_3 + \mathsf{span}(\hat{B}_{5,6})\Big) &= br_1z(a^{\ell+1} - a^{\ell}(\mathsf{ID}_1 + \mathsf{ID}_2) + a^{\ell-1}\mathsf{ID}_1\mathsf{ID}_2)\hat{B}_3 + \mathsf{span}(\hat{B}_{5,6})
\end{aligned}$$

  and similarly for $\delta\hat{K}_{\mathsf{ID}_2}$ (using r_2 instead of r_1 and swapping the roles of $\mathsf{ID}_1, \mathsf{ID}_2$). Notice that the terms $a^ib\hat{B}_j$ for $i \in [0, \ell]$,$j = 1, 2, 3$, as well as $bs\hat{B}_1$, $abs\hat{B}_1$, $a^{\ell+1}b\hat{B}_3$, and $\hat{B}_{5,6}$, are all part of the PK_3 component of the public key.

From the description above it is clear that when $\mathsf{ID}_1, \mathsf{ID}_2 \neq a$, then **ReciprocalKeys** indeed returns the correct distribution, namely two shifted singleton keys $\delta\hat{K}_{\mathsf{ID}_1}$, $\delta\hat{K}_{\mathsf{ID}_2}$ where each $\hat{K}_{\mathsf{ID}}$ is drawn from the same distribution as the singleton keys for ID in **KeyGen** and δ is chosen at random in Z_q (and independently of $\hat{K}_{\mathsf{ID}_1}, \hat{K}_{\mathsf{ID}_2}$).

KeyDerive $(PK, S, \hat{K}_S, S')$ (where $S' = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_m\}$ and $S \subseteq S'$). Assume (w.l.o.g.) that S consists of the first n identities in S', namely $S = \{\mathsf{ID}_1, \ldots, \mathsf{ID}_n\}$ with $n \leq m$. Denote $\hat{K}_S = \{\hat{K}'_{\mathsf{ID}_1}, \ldots, \hat{K}'_{\mathsf{ID}_n}\}$, where $\hat{K}'_{\mathsf{ID}_i}$ is the shifted singleton key for ID_i (consisting of $3\ell - 3$ 7-dimensional vectors of source-group elements).

For $i = 1, \ldots, m$, run the $\mathsf{ReciprocalKeys}$ procedure from above with identities ID_i and ID_{i+1} (indexing mod m) to get two shifted singleton keys for these ID's, which we denote by $\hat{L}_{\mathsf{ID}_i}, \hat{M}_{\mathsf{ID}_{i+1}}$, respectively. Namely, set

$$(\hat{L}_{\mathsf{ID}_i}, \hat{M}_{\mathsf{ID}_{i+1}}) \leftarrow \mathsf{ReciprocalKeys}(PK, \mathsf{ID}_i, \mathsf{ID}_{i+1})$$

Then for $i \in [1, n]$ set $\hat{K}^*_{\mathsf{ID}_i} = \hat{K}'_{\mathsf{ID}_i} + \hat{L}_{\mathsf{ID}_i} - \hat{M}_{\mathsf{ID}_i}$, and for $i \in [n+1, m]$ set $\hat{K}^*_{\mathsf{ID}_i} = \hat{L}_{\mathsf{ID}_i} - \hat{M}_{\mathsf{ID}_i}$ (where addition and subtraction is element-wise). The new key is $\hat{K}_S = [\hat{K}^*_{\mathsf{ID}_1}, \ldots, \hat{K}^*_{\mathsf{ID}_m}]$. In Lemma 2 below we show that this $\mathsf{KeyDerive}$ procedure induces almost the same distribution as KeyGen over the decryption key $\hat{K}_{S'}$.

Lemma 2. *For every $S \subseteq S'$ (with $|S'| = m$) and every secret key $\hat{K}_S$ corresponding to S, the procedure* $\mathsf{KeyDerivation}(PK, S, \hat{K}_S, S')$ *draws from a distribution at most $O(m/q)$ away from that of* $\mathsf{KeyGen}(PK, SK, S')$.

Proof. Observe that every 7-vector in a singleton key $\hat{K}_{\mathsf{ID}}$ corresponding to identity ID (as computed by KeyGen) is of the form

$$(\mathsf{expr}(a, s, \mathsf{ID}) + r_{\mathsf{ID}}\mathsf{expr}'(a, s, \mathsf{ID})) \cdot \hat{B}_k$$

where r_{ID} is the scalar that was chosen for this singleton key, $\hat{B}_k$ is one specific row of the matrix $\hat{B}$, and $\mathsf{expr}(a, s, \mathsf{ID})$, $\mathsf{expr}'(a, s, \mathsf{ID})$ are two fixed scalar-valued expressions that depend only on the scalars a, s from the master secret key and on the identity ID. (Note that either $\mathsf{expr}(a, s, \mathsf{ID})$ or $\mathsf{expr}'(a, s, \mathsf{ID})$ can be zero, but not both.)

Considering the same vector in all the shifted singleton keys in $\hat{K}_S$, we have a collection of n vectors, $\hat{\boldsymbol{x}}_1, \ldots, \hat{\boldsymbol{x}}_n$, where

$$\hat{\boldsymbol{x}}_i = r'_{\mathsf{ID}_i}(\mathsf{expr}(a, s, \mathsf{ID}_i) + r_{\mathsf{ID}_i}\mathsf{expr}'(a, s, \mathsf{ID}_i)) \cdot \hat{B}_k$$

where the scalars r'_{ID_i} satisfy $\sum_i r'_{\mathsf{ID}_i} = 1$. For notational convenience, for $i \in [n+1, m]$ we denote $r_{\mathsf{ID}_i} = r'_{\mathsf{ID}_i} = 0$ and $\hat{\boldsymbol{x}}_i = \hat{\mathbf{0}}$ (so we still have $\hat{x}_i$'s of the right format with $\sum_i r'_{\mathsf{ID}_i} = 1$, even when we consider all m elements).

Similarly considering the same vector in all the shifted singleton keys that are generated by $\mathsf{ReciprocalKeys}$, we have vectors $\hat{\boldsymbol{y}}_1 \ldots \hat{\boldsymbol{y}}_m$ (from the $\hat{L}_{\mathsf{ID}_i}$'s) and $\hat{\boldsymbol{z}}_1 \ldots \hat{\boldsymbol{z}}_m$ (from the $\hat{M}_{\mathsf{ID}_i}$'s) of the form

$$\begin{aligned} \hat{\boldsymbol{y}}_i \ &= \ \delta_i(\mathsf{expr}(a, s, \mathsf{ID}_i) + \rho_i\ \mathsf{expr}'(a, s, \mathsf{ID}_i)) \cdot \hat{B}_k \\ \text{and } \hat{\boldsymbol{z}}_i \ &= \delta_{i-1}(\mathsf{expr}(a, s, \mathsf{ID}_i) + \tau_i\ \mathsf{expr}'(a, s, \mathsf{ID}_i)) \cdot \hat{B}_k \end{aligned}$$

where all the scalars δ_i, ρ_i, τ_i, $i = 1 \ldots m$, are chosen at random in Z_q (and indexing is mod m, so $\delta_0 = \delta_m$). Hence the corresponding element in the shifted singleton key $\hat{K}^*_{\mathsf{ID}_i}$ is

$$\hat{\boldsymbol{x}}_i + \hat{\boldsymbol{y}}_i - \hat{\boldsymbol{z}}_i \ = \ \Big((r'_{\mathsf{ID}_i} + \delta_i - \delta_{i-1})\mathsf{expr}(a, s, \mathsf{ID}_i) + (r'_{\mathsf{ID}_i} r_{\mathsf{ID}_i} + \delta_i\rho_i - \delta_{i-1}\tau_i)\mathsf{expr}'(a, s, \mathsf{ID}_i)\Big)\, \hat{B}_k$$

Assuming that $r'_{\mathsf{ID}_i} + \delta_i - \delta_{i-1} \neq 0$, we can denote

$$r^{**}_{\mathsf{ID}_i} \ \stackrel{\text{def}}{=} \ r'_{\mathsf{ID}_i} + \delta_i - \delta_{i-1} \quad \text{and} \quad r^{*}_{\mathsf{ID}_i} \ \stackrel{\text{def}}{=} \ \frac{r'_{\mathsf{ID}_i} r_{\mathsf{ID}_i} + \delta_i\rho_i - \delta_{i-1}\tau_i}{r'_{\mathsf{ID}_i} + \delta_i - \delta_{i-1}}$$

and then we have

$$\hat{\boldsymbol{x}}_i + \hat{\boldsymbol{y}}_i - \hat{\boldsymbol{z}}_i \ = \ r^{**}_{\mathsf{ID}_i}(\mathsf{expr}(a, s, \mathsf{ID}_i) + r^{*}_{\mathsf{ID}_i}\mathsf{expr}'(a, s, \mathsf{ID}_i))\hat{B}_k$$

which is of the right form, and indeed the scalars $r^{**}_{\mathsf{ID}_i}$ satisfy

$$\sum_{i=1}^{m} r^{**}_{\mathsf{ID}_i} \ = \ \sum_{i=1}^{m} r'_{\mathsf{ID}_i} + \sum_{i=1}^{m} \delta_i - \sum_{i=1}^{m} \delta_{i-1} \ = \ \sum_{i=1}^{m} r'_{\mathsf{ID}_i} \ = \ 1$$

Since the δ_i's are random and independent then the $r^{**}_{\mathsf{ID}_i}$'s are also random and independent subject to the constraint that their sum is one. Finally, assuming that none of the $r^{**}_{\mathsf{ID}_i}$'s is zero and also none of the δ_i's are zero (which happens with probability at least $1 - O(m/q)$) then all the $r^{*}_{\mathsf{ID}_i}$'s are random and independent (since the τ_i's and ρ_i's are). $\square$

6 Security Proof

We now prove security of our system based on the decision BDHE-Set and linear assumptions. On a very high level, the proof follows the hash-proof approach: The simulator (in Proposition 3 below) will generate the challenge ciphertext so that this is either a valid ciphertext or an invalid one, depending on whether the input of the simulator is a YES instance or a NO instance of the decision BDHE-Set problem. In our case, a valid ciphertext is spanned by the rows $\hat{A}_{1,2,3,4}$, and an invalid ciphertext also has a component of $\hat{A}_7$. The secret keys will have a random $\hat{B}_7$ component in them, so an invalid ciphertext will be decrypted to a random KEM key (while a valid ciphertext will always be decrypted to the "right KEM key").

We now proceed to describe the actual proof, consisting of four games: roughly, Game 0 is the actual interaction of the adversary with our system, in Game 1 we use decryption rather than encryption to compute the KEM key corresponding to the challenge ciphertext, in Game 2 we add a component of $\hat{B}_7$ to the secret keys, and in Game 3 we add a component of $\hat{A}_7$ to the challenge ciphertext vector. Fix an adversary $\mathcal{A}$, and we denote the event that the adversary wins game i by W_i, and will show that $\Pr[W_i] \approx \Pr[W_{i+1}]$ and that $\Pr[W_3] = \frac{1}{2}$.

Game 0 is the semantic-security game from Definition 2, so $\Pr[W_0] - \frac{1}{2}$ is half the CPA-advantage of the adversary against our system. We describe this game for self-containment: The challenger begins by choosing a random invertible 7×7 scalar matrix $\hat{A} \in G[7 \times 7]$ and random scalars $a, b, s \in Z_q$, and sets $\hat{B} = (\hat{A}^{-1})^t$. From $a, b, s, \hat{A}_{1..4}$ and $\hat{B}_{1..6}$, the challenger computes the secret key $SK = (\hat{B}_{1..6}, s, \hat{\boldsymbol{a}}_\ell)$ and the public key $PK = (PK_1, PK_2, PK_3)$ as in Eq. (2). (Note that the challenger never uses the last three rows of $\hat{A}$ in the public key or anywhere else. We will use that fact in the proof of Proposition 1 below.)

Then the challenger send PK to the IBBE adversary. When the adversary makes a key-reveal query for identity ID, the challenger runs the procedure $\mathsf{KeyGen}(SK, \mathsf{ID})$ from Section 5 to get $\hat{K}_{\mathsf{ID}} = (\hat{\boldsymbol{u}}_{\mathsf{ID}}, \hat{V}_{\mathsf{ID}}, \hat{W}_{\mathsf{ID}}, \hat{X}_{\mathsf{ID}}, \hat{\boldsymbol{y}}_{\mathsf{ID}})$. When the adversary makes a challenge ciphertext query for identity-set S^* then the challenger runs the procedure $\mathsf{KEM}(PK, S^*)$ to get the KEM key and ciphertext

$$(\tilde{k}_1, (\mathsf{F}, \hat{\boldsymbol{c}})) = \mathsf{KEM}(PK, S^*)$$

It also picks another random target-group element $\tilde{k}_0$ and a random bit σ, and returns to the adversary $\tilde{k}_\sigma$ and $(\mathsf{F}, \hat{\boldsymbol{c}})$. (Since the KEM procedure always "pads" the set S to size ℓ, below we assume w.l.o.g. that the target set S given by the adversary is always of size exactly ℓ.) The adversary can make more key-reveal queries, and eventually it halts with output bit σ'. It wins the game if it never made key-reveal query on any of the identities $\mathsf{ID}_i^* \in S^*$ and yet it guessed correctly $\sigma' = \sigma$.

Game 1 proceeds like Game 0, except that we change the KEM procedure that the challenger uses to produce the challenge ciphertext and KEM key. Specifically, the challenger still uses the KEM procedure to get the challenge ciphertext $(\mathsf{F}, \hat{\boldsymbol{c}})$, but uses the decryption procedure to derive the KEM key. Namely, instead of deriving the corresponding KEM key as $\tilde{k}_1 = t \cdot a^{\ell-1} s \tilde{g}$ (as done in the KEM procedure) it uses KeyGen to derive a key $\hat{K}_{\mathsf{ID}_1^*}$ and computes the KEM key as $\tilde{k}_1 = \mathsf{Decrypt}(PK, S, (\mathsf{F}, \hat{\boldsymbol{c}}), \mathsf{ID}_1^*, \hat{K}_{\mathsf{ID}_1^*})$.

Clearly, since decryption always recovers the correct KEM key then the same value of $\tilde{k}_1$ is returned in either Game 0 or Game 1. These games are therefore identical and we have $\Pr[W_0] = \Pr[W_1]$.

Game 2 proceeds like Game 1, except that all the vectors $\hat{\boldsymbol{y}}_{\mathsf{ID}}$ in the secret keys have a $\hat{B}_7$ component in them. Specifically, the challenger sets

$$\hat{\boldsymbol{y}}_{\mathsf{ID}} \;=\; r_{\mathsf{ID}}(a^{\ell-1}\hat{B}_3 + \hat{B}_7) + \mathsf{span}(\hat{B}_{5,6}) \tag{6}$$

where $\hat{B}_7$ is the last row of $\hat{B}$ (that so far was not used anywhere in the system).

This alternative procedure for generating $\hat{\boldsymbol{y}}_{\mathsf{ID}}$ is used both in answering key-reveal queries and when deriving the key $\hat{K}_{\mathsf{ID}_1^*}$ in the alternative KEM procedure that is used for answering the challenge ciphertext query. We now prove that under the decision linear assumption, the attacker cannot distinguish Game 1 from Game 2.

Proposition 1. *There is an adversary $\mathcal{B}$ for the linear problem that has advantage of at least $\Pr[W_2] - \Pr[W_1] - \frac{1}{q}$.*

Proof. The linear adversary $\mathcal{B}$ gets a 3×3 matrix of source-group elements $\hat{M} \in G[3 \times 3]$ and it needs to decide if $\hat{M}$ is an invertible matrix or a rank-two matrix. $\mathcal{B}$ chooses a random invertible matrix $A \in Z_q[7 \times 7]$ and computes $B = (A^{-1})^t$. Let $B_{5,6,7}$ be the 3×7 matrix consisting of the last three rows of B'. It sets $\hat{A} = A \cdot \hat{1}$ and $\hat{B} = B \cdot \hat{1}$, and then replaces the last three rows $\hat{B}_{5,6,7}$ with $\hat{B'}_{5,6,7} = \hat{M} \cdot B_{5,6,7}$. We let $\hat{B}'$ be the matrix with top four rows $\hat{B}_{1,2,3,4}$ and bottom three rows $\hat{B'}_{5,6,7}$. Next $\mathcal{B}$ chooses at random also a, b, s and proceeds just like the challenger in Game 2 using a, b, s, $\hat{A}_{1,2,3,4}$ and $\hat{B}'$.

It is clear that when $\hat{M}$ is an invertible matrix then the distribution over $(\hat{A}_{1..4}, \hat{B})$ is identical to the distribution over $(\hat{A}_{1..4}, \hat{B}')$, so the view of the adversary is distributed identically to Game 2.

When $\hat{M}$ is a rank-two matrix then except with probability $1/q$, the first two rows of $\hat{M}$ are linearly independent and the last row depends on the first two. If this is indeed the case then (a) the distribution over $(\hat{A}_{1..4}, \hat{B}_{1..6})$ is identical to the distribution over $(\hat{A}_{1..4}, \hat{B'}_{1..6})$, and (b) $\hat{B'}_7$ is spanned by $\hat{B'}_{5,6}$, so the distributions $\mathsf{span}(\hat{B'}_{5,6})$ and $r_{\mathsf{ID}}\hat{B'}_7 + \mathsf{span}(\hat{B'}_{5,6})$ are identical. Hence the view of the adversary is distributed identically to Game 1. □

Game 3 proceeds like Game 2, except for the challenge ciphertext, to which the challenger adds an $\hat{A}_7$ component. Namely, the challenger computes a vector $\hat{\boldsymbol{c}'}$ using Eq. (4) as before, but then it chooses a random scalar t', sets $\hat{\boldsymbol{c}} = \hat{\boldsymbol{c}'} + t'\hat{A}_7$, and returns $(\mathsf{F}, \hat{\boldsymbol{c}})$ as the challenge ciphertext. As in Games 1 and 2, the challenger computes the KEM key $\tilde{k}_1$ by applying the decryption procedure to the challenge ciphertext, namely $\tilde{k}_1 = \mathsf{Decrypt}(PK, S, (\mathsf{F}, \hat{\boldsymbol{c}}), \mathsf{ID}_1^*, \hat{K}_{\mathsf{ID}_1^*})$.

We now show that (a) the view of the adversary $\mathcal{A}$ in Game 3 is independent of the bit σ (and therefore $\Pr[W_3] = \frac{1}{2}$), and (b) under the decision BDHE-Set assumption the adversary cannot distinguish Game 2 from Game 3.

Proposition 2. $\Pr[W_3] = \frac{1}{2}$.

Proof. Recall that the KEM key $\tilde{k}_0$ is computed by the challenger using the decryption procedure, namely $\tilde{k}_1 = \left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}_1^*} \right\rangle$, and we can express $\hat{\boldsymbol{c}} = \hat{\boldsymbol{c}}' + t'\hat{A}_7$ where $\hat{\boldsymbol{c}}'$ is spanned by $\hat{A}_{1,2,3,4}$. Also recall from Eq. (5) that

$$\hat{\boldsymbol{d}}_{\mathsf{ID}_1^*} = \hat{\boldsymbol{u}}_{\mathsf{ID}_1^*} - \boldsymbol{q}_{\mathsf{ID}_1^*} \cdot \hat{W}_{\mathsf{ID}_1^*} - \frac{\boldsymbol{g}_{\mathsf{ID}_1^*} \cdot \hat{V}_{\mathsf{ID}_1^*} - \boldsymbol{q}_{\mathsf{ID}_1^*} \cdot \hat{X}_{\mathsf{ID}_1^*} - \hat{\boldsymbol{y}}_{\mathsf{ID}_1^*}}{\mathsf{F}(\mathsf{ID}_1^*)},$$

where $\hat{\boldsymbol{u}}_{\mathsf{ID}_1^*}$, $\hat{V}_{\mathsf{ID}_1^*}$, $\hat{W}_{\mathsf{ID}_1^*}$ and $\hat{X}_{\mathsf{ID}_1^*}$ depend only on the first three rows $\hat{B}_{1,2,3}$, and from Eq. (6) we have $\hat{\boldsymbol{y}}_{\mathsf{ID}_1^*} = r_{\mathsf{ID}_1^*}(a^{\ell-1}\hat{B}_3 + \hat{B}_7) + \mathsf{span}(\hat{B'}_{5,6})$. Hence we can express $\hat{\boldsymbol{d}}_{\mathsf{ID}_1^*} = \hat{\boldsymbol{d}}' + \frac{r_{\mathsf{ID}_1^*}}{\mathsf{F}(\mathsf{ID}_1^*)}\hat{B}_7$, where $\hat{\boldsymbol{d}}'$ is spanned by $\hat{B'}_{1,2,3,5,6}$.

The correctness argument from Section 5.1 still applies to the vectors $\hat{\boldsymbol{c}}'$ and $\hat{\boldsymbol{d}}'$ since the coefficients of $\hat{A}_{1,2,3}$ and $\hat{B'}_{1,2,3}$ are the same and all the others drop from expression, so we have $\left\langle \hat{\boldsymbol{c}}', \hat{\boldsymbol{d}}' \right\rangle = t \cdot a^{\ell-1}\tilde{s}$. Also, since $\hat{\boldsymbol{c}}'$, $\hat{\boldsymbol{d}}'$ are orthogonal to $\hat{B}_7$, $\hat{A}_7$, respectively, we have

$$\tilde{k}_1 \;=\; \left\langle \hat{\boldsymbol{c}}, \hat{\boldsymbol{d}}_{\mathsf{ID}_1^*} \right\rangle \;=\; \left\langle \hat{\boldsymbol{c}}', \hat{\boldsymbol{d}}' \right\rangle + \left\langle t'\hat{A}_7,\; \frac{r_{\mathsf{ID}_1^*}}{\mathsf{F}(\mathsf{ID}_1^*)}\hat{B}_7 \right\rangle \;=\; \left(a^{\ell-1}st + \frac{r_{\mathsf{ID}_1^*}t'}{\mathsf{F}(\mathsf{ID}_1^*)} \right)\tilde{g}$$

Now observe that $r_{\mathsf{ID}_1^*}$ is chosen by the challenger at random for the purpose of deriving the key $\hat{K}_{\mathsf{ID}_1^*}$ when answering the challenge ciphertext query, and is never used anywhere else. It follows that $\tilde{k}_1$ — just like $\tilde{k}_0$ — is a random target-group element, independent of anything else in the view of $\mathcal{A}$. Therefore the view of $\mathcal{A}$ is independent of the bit σ that chooses between $\tilde{k}_0$ and $\tilde{k}_1$, and so $\Pr[\sigma' = \sigma] = \frac{1}{2}$. □

6.1 The main reduction

We now proceed to show that under the BDHE-Set assumption, the adversary cannot distinguish Game 2 from Game 3. Recall that the BDHE-Set problem is parametrized by a set of integers $\mathcal{S}$ and another integer $m \notin \mathcal{S} + \mathcal{S}$: The adversary gets source group elements $\hat{a}^i = a^i \cdot \hat{1}$ for all $i \in \mathcal{S}$ (where a is chosen at random in Z_q^*), and (roughly) it cannot distinguish $\tilde{a}^m = a^m \cdot \tilde{1}$ in the target group from random. The formal assumption is somewhat stronger, however, giving the adversary not the target group element $\tilde{a}^m$ itself, but rather two random source group elements whose product is $\tilde{a}^m$. Namely, the adversary gets $\hat{z}_1, \hat{z}_2$ such that either the $\hat{z}_i$'s are random or they satisfy $\hat{z}_1\hat{z}_2 = \tilde{a}^m$.

Below we set the parameters $\mathcal{S}$ and m as needed for the reduction. The main challenge is to make the set $\mathcal{S}$ "large enough" so the simulator can produce the entire view of the adversary from the elements $\{\hat{a^i} : i \in \mathcal{S}\}$ that it knows (and $\hat{z}_i, \hat{z}_2$), while at the same time ensuring that $\mathcal{S}$ is "small enough" so that the target integer m is not in $\mathcal{S} + \mathcal{S}$ (since otherwise the problem becomes easy). Before presenting the reduction itself, we describe some tools that are used in this proof.

From BDHE-Set to linear (in)dependence. Recall that the difference between Game 2 and Game 3 is that the ciphertext vector in one is spanned by $\hat{A}_{1,2,3,4}$ and in the other it is not. We move from the BDHE-Set problem to linear dependence using the following simple trick: consider a 3×3 matrix over Z_q:

$$M = \begin{pmatrix} \hat{a}^{n_1} & \hat{a}^{n_2} & 0 \\ 0 & -\hat{z}_1 & \hat{a}^{n_3} \\ \hat{a}^{n_4} & 0 & \hat{z}_2 \end{pmatrix}$$

Its determinant is $\mathsf{det}(M) = -a^{n_1} z_1 z_2 + a^{n_2+n_3+n_4}$, so the three rows of M are linearly dependent if and only if $z_1 z_2 = a^{-n_1+n_2+n_3+n_4}$. We can therefore use $n_1, n_2, n_3, n_4 \in \mathcal{S}$ and set $m = -n_1 + n_2 + n_3 + n_4$. Jumping ahead, we note that the 3×3 sub-matrix consisting of entries $3, 4, 7$ in rows A_3, A_4, A'_7 in the proof of Proposition 3 below is exactly this matrix M.

Polynomials and resultants. The reduction to BDHE-Set below uses several polynomials in the secret scalar a (that the simulator does not know). To compute the appropriate terms, the simulator will have to use powers of the source-group element $\hat{a}$ from its input. It will therefore be crucial that whenever the simulator *is not given* some power $\hat{a}^i$ in its input, the coefficients of a^i in the relevant polynomials be zero. To this end, we use the following lemma:

Lemma 3 (As in [15]). *Let* H, P *be two polynomials of degrees* h *and* p *respectively, that do not share any common roots. Then there exists a polynomial* T *of degree* $h+p-1$ *that satisfies the following three conditions:*

1. *The coefficient of* x^h *in the polynomial* $\mathsf{T} \cdot \mathsf{H}$ *is one.*
2. *For any* $i \in [h+1, h+p-1]$*, the coefficient of* x^i *in the polynomial* $\mathsf{T} \cdot \mathsf{H}$ *is zero.*
3. *For any* $i \in [p, h+p-1]$*, the coefficient of* x^i *in the polynomial* $\mathsf{T} \cdot \mathsf{P}$ *is zero.*

Moreover, the coefficients of T *can be computed efficiently from those of* H *and* P*.*

Proof. Recall that the *Sylvester matrix* [20] of H and P is an $(h+p) \times (h+p)$ matrix, obtained by shifting the coefficients of H and P as follows:

$$S(\mathsf{H}, \mathsf{P}) \;=\; \begin{pmatrix} H_h & \dots & H_1 & H_0 & & \\ & & \ddots & & & \\ & & H_h & \dots & H_1 & H_0 \\ P_p & \dots & P_0 & & & \\ & P_p & \dots & P_0 & & \\ & & \ddots & & & \\ & & & P_p & \dots & P_0 \end{pmatrix}$$

The determinant of $S(\mathsf{H}, \mathsf{P})$ equals the resultant of H and P, hence $S(\mathsf{H}, \mathsf{P})$ is invertible when H and P do not share any common roots. The conditions on T from above are equivalent to

$$S(\mathsf{H}, \mathsf{P}) \cdot \begin{bmatrix} T_0 \\ T_1 \\ \vdots \\ T_{h+p-1} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

hence the coefficients of T can be obtained as $S(\mathsf{H}, \mathsf{P})^{-1} \cdot [1\ 0\ \dots\ 0]^t$. $\square$

Proposition 3. *There is an adversary $\mathcal{C}$ for the BDHE-Set problem that has advantage at least* $\Pr[W_3] - \Pr[W_2] - O(\ell/q)$.

Proof. The BDHE-Set adversary $\mathcal{C}$ gets as input source-group elements $\hat{a^i} = a^i \cdot \hat{1}$ for all $i \in \mathcal{S}$ and two additional source-group elements $\hat{z}_1, \hat{z}_2$. Below we call $\mathcal{C}$ "the simulator."

On a high level, $\mathcal{C}$ chooses a random polynomial $\mathsf{H}(x)$ of high-enough degree over Z_q, and derives much of the randomness that is needed from this one polynomial. Specifically, it will set $s = \mathsf{H}(a)$ for the master secret key, $r_{\mathsf{ID}} = H(\mathsf{ID})$ in all the key-reveal queries, and (roughly) $\mathsf{F} = \mathsf{H} \bmod \mathsf{P}$ for the challenge ciphertext. Let q^* be some bound on the number of key-reveal queries and ℓ be the depth of the hierarchy, then the degree of H (which we denote by h) must be at least $h \geq \ell + q^* + 2$, since we need to handle q^* key-reveal queries, one more key-derivation to generate the KEM key in the challenge ciphertext query, ℓ degrees of freedom for the polynomial F, and one more degree of freedom for s.

In addition to the h-degree polynomial H, the simulator also chooses a random invertible matrix $R \in Z_q[7 \times 7]$, and a random scalar β. It will set up a simulation using the a term from its input, $b = \beta a^{3h+3\ell}$, $s = \mathsf{H}(a)$, and the matrices

$$A_{1..4} = \begin{pmatrix} a^{h+\ell} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a^{2h+2\ell} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a^{2h+2\ell} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -z_2 & 0 & 0 & a^{\ell-1} \end{pmatrix} \cdot R$$

$$A'_7 = \begin{pmatrix} 0 & 0 & a^{2h+\ell} & 0 & 0 & 0 & z_1 \end{pmatrix} \cdot R$$

and

$$B_{1,2,3,5,6,7} = \begin{pmatrix} a^{-h-\ell} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & a^{-2h-2\ell} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a^{-2h-2\ell} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -a^{-2h-\ell-1} & a^{\ell-1} & 0 & 0 & z_2 \end{pmatrix} \cdot (R^{-1})^t$$

One can check that these matrices satisfy the following conditions:

- $A_{1,2,3,4}$ are linearly independent and so are $B_{1,2,3,5,6,7}$;
- $\langle A_i, B_i \rangle = 1$ for $i = 1, 2, 3$ and $\langle A_i, B_j \rangle = 0$ for all $i \neq j$;
- If $z_1 z_2 = a^{-1}$ then $A'_7 = a^{-\ell} A_3 + z_1 a^{-\ell+1} A_4$ and $\langle A'_7, B_7 \rangle = 0$;
- If $z_1 z_2 \neq a^{-1}$ then A'_7 is independent of $A_{1,2,3,4}$ and $\langle A'_7, B_7 \rangle \neq 0$.

Moreover, since R is a random invertible matrix, then A, B above are random subject to these conditions.

The public key. The simulator provides the adversary a public key which is consistent with the matrices $A_{1..4}, B_{1,2,3,5,6}$ above. However, the simulator $\mathcal{C}$ does not know a, so it uses its input (which includes $\hat{a}^i$ for $i \in \mathcal{S}$) to compute the powers of $\hat{a}$ that are needed for this public key. This implies the some constraints on the set of indexes $\mathcal{S}$ as follows:

– The simulator must be able to produce the target group element $a^{\ell-1}\tilde{s} = a^{\ell-1}\mathsf{H}(a)\cdot\tilde{1}$. This means that the interval $[\ell-1, h+\ell-1]$ must be "covered" by $\mathcal{S}+\mathcal{S}$.
– The other components of the public key consist of source-group elements, which the simulator must compute from the powers of $\hat{a}$ that it is given in its input. Specifically, we need the following:

To be able to compute the terms	$\mathcal{S}$ must include
$a^i\hat{A}_1 \quad (i\in[0,\ell])$	$[h+\ell,\ h+2\ell]$
$s\hat{A}_2 \ = \mathsf{H}(a)\hat{A}_2$	$[2h+2\ell,\ 3h+2\ell]$
$a^i\hat{A}_3 \quad (i\in[0,\ell-1])$	$[2h+2\ell,\ 4h+3\ell-1]$
$\hat{A}_4$	$\{\ell-1\}$
$bs\hat{B}_1 \ = \beta a^{3h+3\ell}\cdot\mathsf{H}(a)\cdot\hat{B}_1$	$[2h+2\ell,\ 3h+2\ell]$
$abs\hat{B}_1 = \beta a^{3h+3\ell+1}\cdot\mathsf{H}(a)\cdot\hat{B}_1$	$[2h+2\ell+1,\ 3h+2\ell+1]$
$a^ib\hat{B}_1 = \beta a^{3h+3\ell+i}\hat{B}_1 \quad (i\in[0,\ell])$	$[2h+2\ell,\ 2h+3\ell]$
$a^ib\hat{B}_2 = \beta a^{3h+3\ell+i}\hat{B}_2 \quad (i\in[0,\ell])$	$[h+\ell,\ h+2\ell]$
$a^ib\hat{B}_3 = \beta a^{3h+3\ell+i}\hat{B}_2 \quad (i\in[0,\ell+1])$	$[h+\ell,\ h+2\ell+1]$
$\hat{B}_{5,6}$	$\{0\}$
Summing up these conditions:	$\{0,\ell-1\}\cup[h+\ell,h+2\ell+1]\cup[2h+2\ell,3h+2\ell+1]$

Key-reveal queries. On a key-reveal query for identity ID, the simulator verifies that $\mathsf{ID}\neq a$ (otherwise it have learned a and it can abort the simulation and directly solve the decision problem). Then it sets $r_{\mathsf{ID}} = H(\mathsf{ID})$, computes the polynomial of degree $(h-1)$

$$\mathsf{H}'_{\mathsf{ID}}(x) = (\mathsf{H}(x)-\mathsf{H}(\mathsf{ID}))/(x-\mathsf{ID})$$

(so $\mathsf{H}'_{\mathsf{ID}}(a) = \frac{\mathsf{H}(a)-\mathsf{H}(\mathsf{ID})}{a-\mathsf{ID}} = \frac{s-r_{\mathsf{ID}}}{a-\mathsf{ID}}$), and reply with $\hat{K}_{\mathsf{ID}} = (\hat{\boldsymbol{u}}_{\mathsf{ID}}, \hat{V}_{\mathsf{ID}}, \hat{W}_{\mathsf{ID}}, \hat{X}_{\mathsf{ID}}, \hat{\boldsymbol{y}}_{\mathsf{ID}})$, where

$$\begin{aligned}
&\hat{\boldsymbol{u}}_{\mathsf{ID}} = \mathsf{H}'_{\mathsf{ID}}(a)\hat{B}_1\\
&\hat{V}_{\mathsf{ID}} = r_{\mathsf{ID}}(\boldsymbol{a}_{\ell-2}\times\hat{B}_1),\quad \hat{W}_{\mathsf{ID}} = \boldsymbol{a}_{\ell-2}\times\hat{B}_2,\quad \hat{X}_{\mathsf{ID}} = r_{\mathsf{ID}}(\boldsymbol{a}_{\ell-2}\times\hat{B}_3)\\
&\hat{\boldsymbol{y}}_{\mathsf{ID}} = r_{\mathsf{ID}}(a^{\ell-1}\hat{B}_3+\hat{B}_7)+\mathsf{span}(\hat{B}_{5,6})
\end{aligned}$$

Again, to compute these values the simulator $\mathcal{C}$ needs to use powers of $\hat{a}$ from its input, which implies the following conditions on $\mathcal{S}$:

To be able to compute the terms	$\mathcal{S}$ must include
$\mathsf{H}'(a)\hat{B}_1$	$[-h-\ell,\ -\ell-1]$
$r_{\mathsf{ID}}a^i\hat{B}_1 \quad (i\in[0,\ell-2])$	$[-h-\ell,\ -h-2]$
$a^i\hat{B}_2,\ r_{\mathsf{ID}}a^i\hat{B}_3 \quad (i\in[0,\ell-2])$	$[-2h-2\ell,\ -2h-\ell-2]$
$r_{\mathsf{ID}}(a^{\ell-1}\hat{B}_3+\hat{B}_7)+\mathsf{span}(\hat{B}_{5,6})$	$\{0,\ \ell-1\}$
Summing up these conditions:	$[-2h-2\ell,\ -2h-\ell-1]\cup[-h-\ell,\ -\ell-1]\cup\{0,\ell-1\}$

The target ciphertext. When the adversary asks for the target ciphertext with target identities $\{\mathsf{ID}_1^*,\ldots,\mathsf{ID}_\ell^*\}$, the simulator $\mathcal{C}$ first verifies that $\mathsf{ID}_i^*\neq a$ for all i. (Otherwise it can abort the simulation and directly solve the decision problem.) Then, $\mathcal{C}$ also verifies that $\mathsf{H}(\mathsf{ID}_i^*)\neq 0$ for all i, and that the adversary never made a key-reveal query for any of the ID_i^*'s, and otherwise it aborts and outputs a random bit.

The simulator sets the polynomial $\mathsf{P}(x) = \prod_i (x - \mathsf{ID}_i^*)$, and by the verification steps above we know that $\mathsf{H}(x)$, $\mathsf{P}(x)$ have no common root. By Lemma 3, there exists a polynomial T of degree $h+\ell-1$ such that (a) the coefficient of x^h in the polynomial $\mathsf{T}(x)\mathsf{H}(x)$ is one; (b) the coefficients of x^i for $i \in [h+1,\ h+\ell-1]$ in $\mathsf{T}(x)\mathsf{H}(x)$ are all zero; and (c) the coefficients of x^j for $j \in [\ell,\ h+\ell-1]$ in $\mathsf{T}(x)\mathsf{P}(x)$ are all zero. Moreover, $\mathcal{C}$ can efficiently compute the polynomial T from H and P. The simulator factors the polynomial T, checking that none of its roots is the secret scalar a. (If it is then the simulator can abort the simulation and directly solve the decision problem.)

Then the simulator $\mathcal{C}$ chooses random scalars ρ, τ and sets the following polynomials

$$\mathsf{F}(x) = \rho\mathsf{H}(x) \bmod \mathsf{P}(x), \quad \mathsf{Q}_1(x) = \mathsf{T}(x)\mathsf{P}(x), \quad \mathsf{Q}_2(x) = \mathsf{T}(x)\mathsf{H}(x), \quad \mathsf{Q}_3(x) = \mathsf{T}(x)\mathsf{F}(x) - \rho x^h$$

Then it computes

$$\hat{\boldsymbol{c}} = \tau \cdot a^{-h-\ell}\left(\mathsf{Q}_1(a)\hat{A}_1 + \mathsf{Q}_2(a)\hat{A}_2 + \mathsf{Q}_3(a)\hat{A}_3\right) + \rho\tau\hat{A}'_7 + \mathsf{span}(\hat{A}_4)$$

and sets the target ciphertext $(\mathsf{F}, \hat{\boldsymbol{c}})$. Next the simulator derives the key $\hat{K}_{\mathsf{ID}_1^*}$ just as in the key-reveal queries above and uses it to recover the KEM key, setting $\tilde{k}_1 = \mathsf{Decrypt}(PK, \{\mathsf{ID}_1^*, \ldots, \mathsf{ID}_\ell^*\}, (\mathsf{F}, \hat{\boldsymbol{c}}), \mathsf{ID}_1^*, \hat{K}_{\mathsf{ID}_1^*})$. Finally, it chooses another random target group element $\tilde{k}_0$ and a random bit σ, and returns to the adversary the target ciphertext $(\mathsf{F}, \hat{\boldsymbol{c}})$ and the key $\tilde{k}_\sigma$.

The set $\mathcal{S}$. As before, also in the target-ciphertext query the simulator must use the powers of $\hat{a}$ from its input to compute $\hat{\boldsymbol{c}}$, hence we get some more conditions on $\mathcal{S}$. Before describing these conditions, however, we examine the coefficients of the polynomials $\mathsf{Q}_{1,2,3}$.

- By definition of T, the polynomial $\mathsf{Q}_1(x) = \mathsf{T}(x)\mathsf{P}(x)$ has degree $h + 2\ell - 1$, and the coefficients of x^j for $j \in [\ell,\ h + \ell - 1]$ are all zero.
- Similarly, $\mathsf{Q}_2(x) = \mathsf{T}(x)\mathsf{H}(x)$ has degree $2h + \ell - 1$, and we know that the coefficient of x^h is one and the coefficients of x^j for $j \in [h + 1,\ h + \ell - 1]$ are all zero.
- To analyze the degree-$(h+2\ell-2)$ polynomial $\mathsf{Q}_3(x) = \mathsf{T}(x) \cdot (\rho\mathsf{H}(x) \bmod \mathsf{P}(x)) - \rho x^h$, we write it explicitly as

  $$\mathsf{Q}_3(x) \ = \ \mathsf{T}(x) \cdot \left(\rho\mathsf{H}(x) - \mathsf{Q}^*(x)\mathsf{P}(x)\right) - \rho x^h \ = \ \rho\left(\mathsf{Q}_2(x) - x^h\right) - \mathsf{Q}_1(x)\mathsf{Q}^*(x)$$

  where the quotient polynomial $\mathsf{Q}^*(x)$ is of degree $h - \ell$. Now, we know that all the coefficients of x^j for $j \in [\ell,\ h+\ell-1]$ in $\mathsf{Q}_1(x)$ are zero, and since $\mathsf{Q}^*(x)$ is of degree $h-\ell$ it follows that at most $h-\ell$ coefficients at the bottom of the interval $[\ell,\ h+\ell-1]$ can be non-zero in $\mathsf{Q}_1(x)\mathsf{Q}^*(x)$. In other words, all the coefficients of x^j for $j \in [h,\ h + \ell - 1]$ in $\mathsf{Q}_1(x)\mathsf{Q}^*(x)$ must be zero. We also know from above that the coefficients of x^j for $j \in [h,\ h + \ell - 1]$ in $\mathsf{Q}_2(x) - x^h$ are all zero, so we conclude that all these coefficients in $\mathsf{Q}_3(x)$ must also be zero.

Using these facts about the coefficients of $\mathsf{Q}_{1,2,3}$, we can now describe the conditions that the target ciphertext imposes on $\mathcal{S}$:

To compute the terms	where Q_i has non-zero coefficients	$\mathcal{S}$ must include
$a^{-h-\ell}\mathsf{Q}_1(a)\hat{A}_1$	$[0, \ell-1] \cup [h+\ell, h+2\ell-1]$	$[0,\ \ell-1] \cup [h+\ell,\ h+2\ell-1]$
$a^{-h-\ell}\mathsf{Q}_2(a)\hat{A}_2$	$[0, h] \cup [h+\ell, 2h+\ell-1]$	$[h+\ell,\ 2h+\ell] \cup [2h+2\ell,\ 3h+2\ell-1]$
$a^{-h-\ell}\mathsf{Q}_3(a)\hat{A}_3$	$[0, h-1] \cup [h+\ell, h+2\ell-2]$	$[h+\ell,\ 2h+\ell] \cup [2h+2\ell,\ 2h+3\ell-1]$
$\hat{A}_4, \hat{A}'_7$		$\{\ell-1,\ 2h+\ell\}$

Summing up all the intervals that must be included in $\mathcal{S}$ for the public key, secret keys, and the target ciphertext, we get

$$\begin{aligned}\mathcal{S} &= [-2h-2\ell,\ -2h-\ell-2] \cup [-h-\ell,\ -\ell-1] \\ &\quad \cup [0,\ell-1] \cup [h+\ell, 2h+\ell] \cup [2h+2\ell, 3h+2\ell+1]\end{aligned}$$

This set $\mathcal{S}$ is depicted in Figure 1, which makes it clear that $\mathcal{S}+\mathcal{S}$ indeed does not include the target integer $m=-1$. On the other hand, $\mathcal{S}+\mathcal{S}$ does include the interval $[\ell-1, h+\ell-1]$ that is needed for the target group element in PK_1, due to

$$[-2h-2\ell, -2h-\ell-2] + [2h+2\ell, 3h+2\ell+1] = [0, h+\ell-1] \supseteq [\ell-1, h+\ell-1]$$

Concluding the simulation. When the adversary halts with output bit σ', the simulator also halts, outputting 1 if $\sigma' = \sigma$ and 0 otherwise.

6.2 Analysis of the main reduction

Having shown that the simulator can actually carry out the simulation from above efficiently, it is left to show that the advantage of the simulator is solving BDHE-Set is as promised. Note that if any of the identities that the adversary specifies happens to be equal to a, or if a is a root of the polynomial T, then the simulator will solve the BDHE-Set instance correctly. Below we therefore assume that this never happens. Similarly, we assume that the adversary never asks a key-reveal query for any of the target identities ID_i^* (since this preclude the adversary from winning).

With these assumptions, we begin by arguing that the probability of the simulator aborting is small. If the adversary never asks about the identity a, then the simulator only aborts if one of the target identities ID_i^* happens to be a root the polynomial H. But prior to specifying the target identities, the only information that the adversary has on H is the values that it takes $s = \mathsf{H}(a)$ and $r_{\mathsf{ID}} = \mathsf{H}(\mathsf{ID})$ for the ID's from the key-revel queries. As H has high-enough degree (more than q^*+1), it follows that the values that it takes on any other point (including on each of the ID_i^*'s) is random. Thus the probability that any specific target identity is a root of H is $1/q$, and by the union bound the probability that the simulator aborts is at most ℓ/q.

Next we assume that none of the target identities is a root of H and prove that the view of the adversary has the right distribution. Namely, the view of the adversary in this simulation is as in Game 2 when the BDHE-Set input is a YES instance, and as in Game 3 when it is a NO instance. We have already argued that the matrices $A_{1,2,3,4}$ and $B_{1,2,3,5,6,7}$ that are used for the public and secret keys are drawn from the right distribution. This is also true of the scalars a (from the input) and b (since it depends on the random β that the simulator chose) and of $s = \mathsf{H}(a)$ (since H is random). Hence the public key in the simulation has the correct distribution.

Let $\mathsf{ID}_1, \ldots, \mathsf{ID}_{q^*}$ be the identities that the adversary asks in key-reveal queries, and $\mathsf{ID}_1^*, \ldots, \mathsf{ID}_\ell^*$ the ones specified in the target-ciphertext query. Note that all of these identities are distinct and they all different from a (or else the simulator aborts). Since H is a random polynomial of degree $h \geq q^*+\ell+1$, then all the values $\mathsf{H}(a)$, $\mathsf{H}(\mathsf{ID}_i)$, $\mathsf{H}(\mathsf{ID}_j^*)$ are random and independent. As the public key in the simulation depends on H only via $\mathsf{H}(a)$, it follows that $\mathsf{H}(\mathsf{ID}_i)$ and $\mathsf{H}(\mathsf{ID}_j^*)$ are random and independent of the public key. Since the simulator uses $r_{\mathsf{ID}} = \mathsf{H}(\mathsf{ID})$ for key-reveal queries, it follows that the secret keys that it returns in these queries have the correct distribution. Moreover, note that the ID_j^*'s are roots of P, so for all $j \in [1, \ell]$ we have

$$\mathsf{F}(\mathsf{ID}_j^*) \;=\; (\rho\mathsf{H} \bmod \mathsf{P})(\mathsf{ID}_j^*) \;=\; \rho\mathsf{H}(\mathsf{ID}_j^*) - \mathsf{Q}^*(\mathsf{ID}_j^*)\mathsf{P}(\mathsf{ID}_j^*) \;=\; \rho\mathsf{H}(\mathsf{ID}_j^*)$$

Thus all the values $\mathsf{F}(\mathsf{ID}_j^*)$ are random and independent of the public key and the secret keys in the simulation, and thus F itself is a random degree-$(\ell-1)$ polynomial, independent of these keys. Moreover, due to the random factor of ρ then F is also independent of the value of $r_{\mathsf{ID}_1^*} = H(\mathsf{ID}_1^*)$ (since we assume $H(\mathsf{ID}_1^*) \neq 0$).

So far we demonstrated that the distribution of the public key, the secret keys $\hat{K}_{\mathsf{ID}}$ in reveal queries, the secret key $\hat{K}_{\mathsf{ID}_1^*}$ in the target-ciphertext query, and the polynomial F are drawn from the correct distribution. All that is left to show is that the vector $\hat{\boldsymbol{c}}$ also has the right distribution. Here we have two cases, depending on whether the BDHE-Set input in the simulation is a YES instance or a NO instance.

Case 1: a YES instance. In this case, the elements $\hat{z}_1, \hat{z}_2$ satisfy $\langle \hat{z}_1, \hat{z}_2 \rangle = a^{-1} \cdot \tilde{1}$, which implies that $\hat{A'}_7 \in a^{-\ell}\hat{A}_3 + \mathsf{span}(\hat{A}_4)$, so we can re-write the ciphertext vector as

$$\begin{aligned}
\hat{\boldsymbol{c}} &= \tau a^{-h-\ell}\left(\mathsf{Q}_1(a)\hat{A}_1 + \mathsf{Q}_2(a)\hat{A}_2 + \mathsf{Q}_3(a)\hat{A}_3\right) + \rho\tau a^{-\ell}\hat{A}_3 + \mathsf{span}(\hat{A}_4) \\
&= \tau a^{-h-\ell}\left(\mathsf{Q}_1(a)\hat{A}_1 + \mathsf{Q}_2(a)\hat{A}_2 + (\mathsf{Q}_3(a) + \rho a^h)\hat{A}_3\right) + \mathsf{span}(\hat{A}_4) \\
&= \underbrace{\tau a^{-h-\ell}\mathsf{T}(a)}_{t}\left(\mathsf{P}(a)\hat{A}_1 + \mathsf{H}(a)\hat{A}_2 + \mathsf{F}(a)\hat{A}_3\right) + \mathsf{span}(\hat{A}_4) \\
&= t\left(\mathsf{P}(a)\hat{A}_1 + s\hat{A}_2 + \mathsf{F}(a)\hat{A}_3\right) + \mathsf{span}(\hat{A}_4)
\end{aligned}$$

where t is random and independent of anything else (since τ is random and $\mathsf{T}(a) \neq 0$).

Case 2: a NO instance. Observe that $\hat{A'}_7$ (and therefore z_1) are used in the simulation only when producing the ciphertext vector $\hat{\boldsymbol{c}}$, so we can think of z_1 as chosen at random only then.

Assuming that $a \neq 0$ and $z_2 \neq 0$, we consider the vector $A_7 = (0\ 0\ 0\ 0\ 0\ 0\ \frac{1}{z_2}) \cdot R$ (where R is the random invertible matrix chosen by the simulator at the beginning). This vector is orthogonal to $B_{1,2,3,5,6}$, and it satisfies $\langle A_7, B_7 \rangle = 1$. Note that we have

$$A'_7 \;=\; a^{-\ell}A_3 \;+\; \frac{a^{-1}}{z_2}A_4 \;+\; (0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ z_1 - \frac{a^{-1}}{z_2}) \cdot R$$

and therefore choosing z_1 at random is equivalent to choosing a random scalar μ and setting $A'_7 = a^{-\ell}A_3 + \frac{a^{-1}}{z_2}A_4 + \mu A_7$. The process of choosing $\hat{\boldsymbol{c}}$ in the simulation is therefore equivalent to choosing a random μ and setting

$$\hat{\boldsymbol{c}} = \underbrace{\tau a^{-h-\ell}\left(\mathsf{Q}_1(a)\hat{A}_1 + \mathsf{Q}_2(a)\hat{A}_2 + \mathsf{Q}_3(a)\hat{A}_3\right) + \rho\tau a^{-\ell}\hat{A}_3 + \mathsf{span}(\hat{A}_4)}_{\hat{\boldsymbol{c}}'} + \underbrace{\rho\tau\mu}_{t'}\hat{A}_7$$

We have shown above (in the YES instance case) that $\hat{\boldsymbol{c}}'$ is distributed as in Game 2, and therefore $\hat{\boldsymbol{c}}$ is distributed as in Game 3. This concludes the proof of Proposition 3, and therefore also concludes the security proof of our system. □

Remark. Recall the variant suggested in the remark in Section 5 (which was meant to save somewhat on the ciphertext size): For target sets S of size $m < \ell$ we suggested to choose F such that $\mathsf{F}(\mathsf{ID}) = 1$ for all the "forbidden identities" that were added to "pad" S to size ℓ.

Proving security of this variant entails a small change to the proof above. Specifically, instead of choosing H completely at random, the simulator $\mathcal{C}$ chooses the scalar ρ (for the challenge ciphertext) at setup time, and then chooses H subject to the constraint that $\mathsf{H}(\mathsf{ID}) = 1/\rho$ for all the "forbidden identities" $0, 1, \ldots, \ell - 2$. Since we set $\mathsf{F} = \rho\mathsf{H} \bmod \mathsf{P}$ then this gives us the desired constraint on F. In this case we also needs to choose H of a higher degree, namely $h = 2\ell + q^* + 1$.

References

1. D. Boneh and X. Boyen. Efficient Selective-ID Secure Identity Based Encryption Without Random Oracles. In *Proceedings of Eurocrypt'04*, volume 3027 of *LNCS*, pages 223–238, Springer, 2004.
2. D. Boneh and X. Boyen. Secure Identity Based Encryption without Random Oracles. In *Proceedings of CRYPTO'04*, volume 3152 of *LNCS*, pages 443–259, Springer, 2004.
3. D. Boneh, X. Boyen and E.-J. Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertexts. In *Proceedings of EUROCRYPT'05*, volume 3494 of *LNCS*, pages 440–456, Springer, 2005.
4. D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *Proceedings of CRYPTO'04*, volume 3152 of *LNCS*, pages 41–55, Springer, 2004.
5. D. Boneh and M. Franklin. Identity Based Encryption from the Weil Pairing. In *Proceedings of CRYPTO'01*, volume 2139 of *LNCS*, pages 213–229, Springer, 2001.
6. D. Boneh, C. Gentry and M. Hamburg. Space Efficient Identity Based Encryption without Pairings. In *Proceedings of FOCS'07*, pages 647–657, IEEE, 2007.
7. D. Boneh, C. Gentry and B. Waters. Collusion Resistant Broadcast Encryption with Short Ciphertexts and Private Keys. In *Proceedings of CRYPTO'05*, volume 3621 of *LNCS*, pages 258–275, Springer, 2005.
8. X. Boyen. General Ad Hoc Encryption from Exponent Inversion IBE. In *Proceedings of EUROCRYPT'07*, volume 4515 of *LNCS*, pages 394–411, Springer, 2007.
9. R. Canetti, S. Halevi and J. Katz. A Forward-Secure Public-Key Encryption Scheme. In *Proceedings of EUROCRYPT'03*, volume 2656 of *LNCS*, pages 255–271, Springer, 2003.
10. R. Canetti, S. Halevi and J. Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *Proceedings of EUROCRYPT'04*, volume 3027 of *LNCS*, pages 207-222, Springer, 2004.
11. C. Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. In *IMA Int. Conf. '01.*
12. R. Cramer and V. Shoup, Universal Hash Proofs and a Paradigm for Adaptive Chosen Ciphertext Secure Public-Key Encryption. In *Proceedings of Eurocrypt'02*, volume 2332 of *LNCS*, pages 45–64, Springer, 2002.
13. C. Gentry. Practical Identity-Based Encryption without Random Oracles. In *Proceedings of EUROCRYPT'06*, volume 4004 of *LNCS*, pages 445-464, Springer, 2006.
14. C. Gentry and A. Silverberg. Hierarchical ID-Based Cryptography. In *Proceedings of ASIACRYPT'02*, volume 2501 of *LNCS*, pages 548–566, Springer, 2002.
15. C. Gentry and B. Waters, Adaptive Security in Broadcast Encryption Systems. Manuscript, 2008. Available at `http://eprint.iacr.org/2008/268`.
16. J. Horwitz and B. Lynn. Toward Hierarchical Identity-Based Encryption. In *Proceedings of EUROCRYPT'02*, volume 2332 of *LNCS*, pages 466–481, Springer, 2002.
17. Ueli M. Maurer and Yacov Yacobi. Non-interactive Public-Key Cryptography, In *Proceedings of EUROCRYPT'91*, pages 498–507, Springer, 1991.
18. A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In *Proceedings of CRYPTO'84*, pages 47–53, Springer, 1984.
19. E. Shi and B. Waters, Delegating Capabilities in Predicate Encryption Systems. In *Proceedings of ICALP'08 (track C)*, volume 5126 of *LNCS*, pages 560–578, Springer, 2008.
20. E.W. Weisstein. Sylvester Matrix. From MathWorld, a Wolfram Web Resource. `http://mathworld.wolfram.com/SylvesterMatrix.html`
21. Brent Waters. Efficient Identity Based Encryption without Random Oracles. In *Proceedings of EUROCRYPT'05*, volume 3494 of *LNCS*, pages 114–127, Springer, 2005.