Exponent-inversion Signatures and IBE under Static Assumptions

Tsz Hon Yuen *, Sherman S.M. Chow, Cong Zhang, and Siu Ming Yiu

[1] Huawei, Singapore
[2] Department of Information Engineering
Chinese University of Hong Kong, Hong Kong
[3] Department of Computer Science
University of Hong Kong, Hong Kong
sherman@ie.cuhk.edu.hk,{czhang2,smyiu}@cs.hku.hk

Abstract. Boneh-Boyen signatures are widely used in many advanced cryptosystems. It has a structure of "inversion in the exponent", and its unforgeability against q chosen-messages attack is proven under the non-static q-Strong Diffie-Hellman assumption. It has been an open problem whether the exponent-inversion signature, and its various applications, can be proved based on a weaker static assumption.

We propose a *dual-form Boneh-Boyen signature* and demonstrate how to prove the security for the exponent-inversion signature structure in the standard model under static assumptions. We apply our proof technique to a number of related cryptosystems employing similar structure, including anonymous credentials, identity-based encryption (IBE) and accountable authority IBE. Our results give the first exponent-inversion IBE in the standard model under static assumption. Our anonymous credentials and accountable authority IBE are also better than existing schemes in terms of both security and efficiency.

1 Introduction

The invention of the Boneh-Boyen signature [9] has a wide impact and forms the foundation of a wide range of advanced cryptosystems. Given a secret key α and a public key (g, g^α), the Boneh-Boyen signature for a message m is

$$\sigma = g^{\frac{1}{\alpha+m}}.$$

This "inversion in the exponent" structure can be modified to other signature schemes such as structure-preserving signatures, blind signatures [2]. It can also be used as the "second-tier" secret key, such as user secret keys of identity-based encryption (IBE) [18], identity-based broadcast encryption, hierarchical IBE with polynomially many levels [19]; or decryption keys of dynamic threshold decryption [17], user signing keys of group signatures [10], etc.

The security of the Boneh-Boyen signature is based on the q-Strong Diffie-Hellman (SDH) assumption — for α randomly selected from $\mathbb{Z}_p$, we have:

$$\text{Given } g, g^\alpha, g^{\alpha^2}, \ldots, g^{\alpha^q} \in \mathbb{G}, \text{ it is hard to output } (c, g^{\frac{1}{\alpha+c}}), \text{ where } c \in \mathbb{Z}_p.$$

While the group size $|\mathbb{G}| = p$ grows exponentially with the security parameter λ, the number q is polynomially bounded in λ. Specifically, this long problem instance of size $O(q)$ is used to simulate $(q-1)$ signing oracle queries. Due to this reliance, q-SDH is considered as a *non-static* assumption, in contrast to the traditional *static* ones like discrete logarithm or computational Diffie-Hellman.

* This work was done when the first author was in the University of Hong Kong.

Dual Form Boneh-Boyen Signatures. Recently, Gerbush *et al.* [20] proposed the framework of dual form signatures as an approach for proving security from static assumptions. A normal signature is given the signing algorithm $\texttt{Sign}_A$, while another type of signature can only be given by another algorithm $\texttt{Sign}_B$. The security proof involves a sequence of transformation of answering oracle query, from using $\texttt{Sign}_A$ to $\texttt{Sign}_B$. It also involves a challenge signature which can take either one of the forms. Gerbush *et al.* [20] constructed multiple signature schemes in the composite order group setting.

In this paper, we propose the *Dual Form Boneh-Boyen Signatures*, prove its security by the notion of dual form signatures using static assumption, and demonstrate how it helps in getting rid of static assumptions from a number of higher cryptographic applications. While the concept appears to be simple, the task is not trivial. Simply instantiating the Boneh-Boyen signature in a composite order group $\mathbb{G}_N$ where $N = p_1p_2p_3$ and adding additional randomization in $\mathbb{G}_{p_3}$ (say, $g_3^{x_1}$) does not work since there is no randomness in the subgroup $\mathbb{G}_{p_1}$. Existing static assumption contains a random part in $\mathbb{G}_{p_1}$ which is useful for simulating a challenge signature that can belong to either form of the signatures. Our second attempt is to use the randomized Boneh-Boyen signatures [9]: $(g_1^{\frac{1}{\alpha+m+\beta r}}, r)$, where $(\alpha, \beta) \in (\mathbb{Z}_N)^2$ is the secret key and $r \in \mathbb{Z}_N$. Yet, it is difficult to fit the random element from the problem instance with the inverted exponent to simulate a challenge signature for any given m.

This leads us to devise a signature with a randomized $\mathbb{G}_{p_1}$ component that can be easily simulated for any message without knowing the randomness directly. It turns out that our third attempt is similar to the key structure of Gentry-IBE [18], which is in the form of $\sigma_1 = (h_1 g_1^{-r})^{\frac{1}{\alpha-m}}, \sigma_2 = r$, where $h_1 \in \mathbb{G}_{p_1}$ is part of the public key. Recall our major challenge is to avoid the q-type non-static assumption. Therefore, we have to avoid the direct use of α for simulating signature. For this we made two changes. The first is to use h_1 as the private key instead. The public key now includes $\hat{e}(g_1, h_1)$. On the other hand, the hard problem instances used in most existing dual system schemes [25, 20] do not allow the leakage of the randomness r directly. Therefore, we tried to use g_1^r as part of the signature instead. However, given m, g_1^α from the public key and g_1^r from the signature uniquely determine $(g_1^{-r})^{\frac{1}{\alpha-m}}$, so we need to use another generator $u_1 \in \mathbb{G}_{p_1}$ instead. The signature on m thus becomes:

$$(\sigma_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha-m}} g_3^{x_1}, \quad \sigma_2 = g_1^r g_3^{x_2}),$$

where r, x_1, x_2 are randomly chosen from $\mathbb{Z}_N$. The signing oracles are simulated by h_1 or $h_1 X_2$ for some random $X_2 \in \mathbb{G}_{p_2}$, depending on what is given by the hard problem instance during different security games.

Why is the q-type Assumption not needed? Introducing h_1 as another part of the private key frees our simulation from relying on $g^{\alpha^2}, \ldots, g^{\alpha^q}$ as needed in the original Boneh-Boyen signatures [9]. Although α is no longer treated as part of the hard problem instance, the (partial) secrecy of α still plays an important role for the transition between two different types of signing oracle issuing dual form signatures. We only give g_1^α to the adversary which contains information about $\alpha \bmod p_1$, yet that is not correlated to $\alpha \bmod p_2$ by the Chinese remainder theorem, which will be used in our information-theoretic argument in ensuring the adversary will return something under our expectation with high probability.

Important Applications. We first extend our signature scheme to the prime order group, and it is secure under the standard decision linear assumption. A number of signature schemes were constructed based on the Boneh-Boyen signatures, such as anonymous credentials [13, 5, 4], group signatures [10, 22], etc. We showed the constructions of anonymous credentials without using any q-type non-static assumption. To the best of the authors' knowledge, there is no prior practical anonymous credential scheme which is secure under standard assumption only. In addition, our proposed scheme is more efficient than the existing anonymous credential scheme from structure-preserving signatures [1].

Dual System Gentry-IBE. Apart from exponent inversion, commutative blinding forms another major family of pairing-based IBE in the standard model [12]. Existing schemes based on

the dual system encryption technique [31, 25] are from the commutative blinding family. Gentry-IBE [18] uses exponent inversion in the key. Its security reduction is tight but is based on a q-type assumption. Comparatively, the security proof of the commutative blinding family has a security loss of a factor of q, while using static assumption.

We give the *first* dual-system IBE which has *both* the key structure based on the exponent inversion (same as our proposed dual form signatures) and the commutative blinding property across the key and the ciphertext for derivation of the session key. It is secure in the standard model under static assumption. The session key is of the form $\hat{e}(g, h_1)^s$, where h_1 is part of the master secret key[4]. We call our proposed scheme as the dual system Gentry-IBE due to the similarity. Its efficiency is more or less the same as Lewko-Waters IBE [25].

Accountable Authority IBE (A-IBE). The advantage of our IBE construction is that it can possibly inherit the nice properties Gentry-IBE to support higher application such as A-IBE [21, 26]. A-IBE features a tracing algorithm which can determine if a decryption key ("white-box") or a decoder box ("black-box") was created by the (malicious) private key generator (PKG), so it can be held accountable and proven guilty for any (unauthorized) leakage. We extended our scheme in a few dimensions (e.g., anonymity, interactive key generation protocol, and tracing algorithm) to build a fully-secure black-box A-IBE. The only existing scheme [28] achieves this level of security relies on the use of dummy identities to support black-box tracing with full security, which incur an overhead of a multiplicative factor $O(\lambda)$ for both key and ciphertext sizes (where λ is the security parameter). Our dual-Gentry IBE supports decryption oracle via the use of semi-functional keys, without this extra overhead. Previous A-IBE constructions based on Gentry-IBE at most support *weak black-box traceability* [26] which does not allow decryption oracle query in the dishonest PKG security.

2 Background

Let $\mathcal{G}$ be a bilinear groups generator, that takes a security parameter 1^λ as input where $\lambda \in \mathbb{N}$, outputs a description of bilinear group $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ and possibly generators for some subgroups of $\mathbb{G}$, where p_1, p_2, p_3 are distinct λ-bits primes, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N, and $\hat{e} : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a bilinear map such that $\forall g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_N$, $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$; $\hat{e}(g, g)$ generates $\mathbb{G}_T$ if g is a generator of $\mathbb{G}$. We require that the group operations in $\mathbb{G}$ and $\mathbb{G}_T$, and the bilinear map $\hat{e}$ are computable in polynomial time in λ.

We denote $\mathbb{G}_{p_i}$ as the subgroup of order p_i in $\mathbb{G}$ $(i = 1, 2, 3)$. Let g_i be the generator of the subgroup $\mathbb{G}_{p_i}$. Note that for all $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$, if $i \neq j$, $\hat{e}(h_i, h_j) = 1$. We also denote $\mathbb{G}_{p_1p_2}$ as the subgroup of order p_1p_2 in $\mathbb{G}$. For all $T \in \mathbb{G}_{p_1p_2}$, T can be written uniquely as the product of an element of $\mathbb{G}_{p_1}$ and an element of $\mathbb{G}_{p_2}$. We refer to these elements as the "$\mathbb{G}_{p_1}$ part of T" and the "$\mathbb{G}_{p_2}$ part of T" respectively. We also use this notation for $\mathbb{G}_{p_1p_3}$ and $\mathbb{G} = \mathbb{G}_{p_1p_2p_3}$ similarly. We now give three complexity assumptions [25] with respect to probabilistic polynomial time (PPT) adversaries. The notation $neg(\cdot)$ refers to some *negligible* function which is smaller than $1/p(\cdot)$ for any positive polynomial $p(\cdot)$ (for all sufficiently large inputs).

Assumption 1 [25]. Given $\mathcal{G}$, we define the following distribution:

$$(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g, X_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_3 \xleftarrow{R} \mathbb{G}_{p_3},$$
$$T_0 \xleftarrow{R} \mathbb{G}_{p_1p_2}, \quad T_1 \xleftarrow{R} \mathbb{G}_{p_1}, D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_3).$$

For any PPT algorithm $\mathcal{A}_1$ with output in $\{0, 1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_1} := |\Pr[(D, T_0) = 1] - \Pr[(D, T_1) = 1]| = neg(\lambda).$$

[4] While the session key of Gentry-IBE is $\hat{e}(g, h_1)^s$, h_1 there is a public key. Also, the session key of an exponent inversion IBE is independent of the master secret key.

Assumption 2 [25]. Given $\mathcal{G}$, we define the following distribution:

$$(N, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g, X_1, Z_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_i, Y_i, Z_i \xleftarrow{R} \mathbb{G}_{p_i} (i = 2, 3),$$
$$T_0 = Z_1 Z_3, \quad T_1 = Z_1 Z_2 Z_3. \quad D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1 X_2, X_3, Y_2 Y_3).$$

For any PPT algorithm $\mathcal{A}_2$ with output in $\{0, 1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_2} := |\Pr[(D, T_0) = 1] - \Pr[(D, T_1) = 1]| = neg(\lambda).$$

Assumption 3 [25]. Given $\mathcal{G}$, we define the following distribution:

$$(N, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \alpha, s \xleftarrow{R} \mathbb{Z}_N, g \xleftarrow{R} \mathbb{G}_{p_1}, X_2, Y_2, Z_2 \xleftarrow{R} \mathbb{G}_{p_2}, X_3 \xleftarrow{R} \mathbb{G}_{p_3},$$
$$T_0 = \hat{e}(g, g)^{\alpha s}, \quad T_1 \xleftarrow{R} \mathbb{G}_T. \quad D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^\alpha X_2, g^s Y_2, Z_2, X_3).$$

For any PPT algorithm $\mathcal{A}_3$ with output in $\{0, 1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_3} := |\Pr[(D, T_0) = 1] - \Pr[(D, T_1) = 1]| = neg(\lambda).$$

3 Signature Schemes

3.1 Dual Form Signatures

The security of our signature scheme can be proved by dual form signatures [20]. We first review the definitions of dual form signatures:

- $\mathtt{Setup}$: Given a parameter 1^λ, generate a public key pk and a private key sk.
- $\mathtt{Sign}_A$: Given sk and a message M, output a signature σ.
- $\mathtt{Sign}_B$: Given sk and a message M, output a signature σ.
- $\mathtt{Verify}$: Given pk, a signature σ and a message M, output 'true' or 'false'.

Forgery Class. We denote the set of signature-message pairs for which the $\mathtt{Verify}$ algorithm outputs 'true' as $\mathcal{V}$. We let $\mathcal{V}_I$ and $\mathcal{V}_{II}$ be two disjoint subsets of $\mathcal{V}$. In our applications, we will have $\mathcal{V} = \mathcal{V}_I \cup \mathcal{V}_{II}$. We refer to signatures from these sets as Type I and Type II forgeries, respectively, as two different types of forgeries received from an adversary in our proof of security. Type I forgeries will be related to signatures output by the $\mathtt{Sign}_A$ algorithm and Type II forgeries will be related those by the $\mathtt{Sign}_B$ algorithm. The precise relationships between the forgery types and the signing algorithms are explicitly defined by the following set of security properties for the dual form system.

Security Properties. We briefly review the following properties of dual form signatures [20], where the adversary is given only pk for $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathtt{Setup}(1^\lambda)$.

- A-I Matching. If an attacker is only given a signing oracle which returns outputs from $\mathtt{Sign}_A$, then it is hard to create anything but a Type I forgery.
- B-II Matching. If an attacker is only given a signing oracle which returns outputs from $\mathtt{Sign}_B$, then it is hard to create anything but a Type II forgery.
- Dual-Oracle Invariance. The attacker is given oracle access to $\mathtt{Sign}_A$ and $\mathtt{Sign}_B$. At some time, the attacker outputs a challenge message m. The challenger returns a challenge signature on m from either $\mathtt{Sign}_A$ or $\mathtt{Sign}_B$ with equal probability. Finally, the attacker outputs a forgery pair (m^*, σ^*), where m^* was not asked to any oracle. The attacker's probability of producing a Type I forgery when the challenge signature is from $\mathtt{Sign}_A$ is approximately the same as when the challenge signature is from $\mathtt{Sign}_B$.

A dual form signature scheme is secure if it satisfies all these properties.

Theorem 1 ([20]). *If* $(\mathtt{Setup}, \mathtt{Sign}_A, \mathtt{Sign}_B, \mathtt{Verify})$ *is a secure dual form signature scheme, then* $(\mathtt{Setup}, \mathtt{Sign}_A, \mathtt{Verify})$ *is existentially unforgeable under an adaptive chosen message attack.*

3.2 Dual Form Boneh-Boyen Signatures

$\mathtt{Setup}(1^\lambda)$: It runs $\mathcal{G}(1^\lambda)$ to get $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as defined in Section 2. Suppose $\mathcal{G}$ also gives generators $g_1, g_{2,3}$ and g_3 of the subgroups $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_2p_3}$ and $\mathbb{G}_{p_3}$ respectively. It randomly picks $\alpha \in \mathbb{Z}_N$, $u_1, h_1 \in \mathbb{G}_{p_1}$. The public key is

$$(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u_1, \hat{e}(g_1, h_1), g_1^\alpha).$$

The message space $\mathcal{M}$ is $\mathbb{Z}_N$. The secret key is $(h_1, \alpha, g_3, g_{2,3})$.

$\mathtt{Sign}_A(\mathsf{sk}, m)$: The signer randomly picks $X_3, X_3' \in \mathbb{G}_{p_3}$, $r \in \mathbb{Z}_N$, computes[5] $\frac{1}{(\alpha - m)} \bmod N$, and outputs the signature $\sigma = (\sigma_1, \sigma_2)$, where:

$$\sigma_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - m}} X_3, \quad \sigma_2 = g_1^r X_3'.$$

$\mathtt{Verify}(\mathsf{pk}, \sigma, m)$: Given a signature $\sigma = (\sigma_1, \sigma_2)$, output 'true' if:

$$\hat{e}(g_1^\alpha \cdot g_1^{-m}, \sigma_1) \cdot \hat{e}(u_1, \sigma_2) = \hat{e}(g_1, h_1).$$

Its security proof can be found in Appendix A.

3.3 Extension to the Prime Order Groups

We can turn our signature scheme in composite order groups into one with the prime order setting using the methods by Lewko [24]. Details are given in Appendix B. This scheme is essential for our anonymous credentials in §4.

4 Anonymous Credentials

Anonymous credential allows a prover to show the verifier that the prover possess a certificate from a credential issuer. The prover is anonymous such that he cannot be linked to the registration with the issuer nor any past authentication with any verifier. An extension which attracts interests recently is blacklistable anonymous credential where a *verifier (i.e., without using any trapdoor)* can put some past session into a blacklist, such that the same prover cannot be authenticated again, thus offering a balance between privacy and system management.

4.1 Anonymous Credentials from P-Signatures

A P-signature scheme [6] is a signature scheme with an efficient non-interactive zero-knowledge proof of knowledge (ZKPoK) of a signature on a committed message. Belenkiy *et al.* [6] showed that anonymous credentials are an immediate consequence of P-signatures. The detailed definitions can be found in [6].

The existing P-signature scheme [6] is F-unforgeable, where the adversary is only asked to output $(\sigma^*, F(m^*))$ instead of (σ^*, m^*) as the forgery, for some bijective function F. For the case of $F(x) = g^x$ where $g \in \mathbb{G}_{p_1}$, F is not bijective since we have $m^* \in \mathbb{Z}_N$. Therefore, we have to first transform our dual form Boneh-Boyen signature into the prime order group using Lewko's technique [24]. Then we define a suitable function F such that our signature scheme is F-unforgeable. The resulting signature scheme FBB is presented in Appendix B.1.

For ZKPoK we rely on the Groth-Sahai non-interactive zero-knowledge (NIZK) proof system $\mathsf{GS} = (\mathsf{Setup}, \mathsf{Commit}, \mathsf{Pf}, \mathsf{Verify})$, using the instantiation under the decision linear assumption. Some vector notations are defined in Appendix B. We are now ready to give our P-signature scheme as follows:

[5] If $\alpha - m$ has no inverse modulo N, it outputs $\perp$. We omit this step later for brevity.

$\mathtt{Setup}(1^\lambda)$: It runs $\mathsf{param}_{\mathsf{GS}} \leftarrow \mathsf{GS.Setup}(1^\lambda)$ and $(\mathsf{pk}', \mathsf{sk}') \leftarrow \mathsf{FBB.Setup}(1^\lambda)$ which uses the same pairing setting. It outputs $\mathsf{param} = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, Z, Y, \mathsf{param}_{\mathsf{GS}})$, $\mathsf{pk} = (g^{\overrightarrow{d_1}}, \ldots, g^{\overrightarrow{d_4}}, g^{\alpha\overrightarrow{d_1}}, g^{\alpha\overrightarrow{d_3}}, \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}})$, $\mathsf{sk} = \mathsf{sk}'$.

$\mathtt{Sign}(\mathsf{param}, \mathsf{sk}, m)$: It returns $\sigma \leftarrow \mathsf{FBB.Sign}(\mathsf{sk}, m)$.

$\mathtt{Verify}(\mathsf{param}, \mathsf{pk}, \sigma, m)$: It returns $\sigma \leftarrow \mathsf{FBB.Verify}(\mathsf{pk}, \sigma, m)$.

$\mathtt{Commit}(\mathsf{param}, m, \mathsf{Open})$: It returns $C \leftarrow \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open})$.

$\mathtt{ObtainSig}(\mathsf{param}, \mathsf{pk}, m, C, \mathsf{Open}) \leftrightarrow \mathtt{IssueSig}(\mathsf{param}, \mathsf{sk}, C)$:

1. The user chooses some random $\rho \in \mathbb{Z}_p$.
2. The issuer chooses some random $r_1, r_2 \in \mathbb{Z}_p$.
3. The user (with private input (m, ρ, Open)) and the issuer (with private input sk, r_1, r_2) engage in a secure two-party computation protocol. It can be efficiently implemented by garbled circuits [7, 3]. Denote $\mathsf{sk} = (g^{\overrightarrow{d_1^*}}, \ldots, g^{\overrightarrow{d_4^*}}, \alpha, \gamma)$. The issuer's private outputs are
$$\overrightarrow{\sigma'} = g^{(\frac{\gamma - r_1}{\alpha - m}\cdot\overrightarrow{d_1^*} + r_1\cdot\overrightarrow{d_2^*} + \frac{\gamma - r_2}{\alpha - m}\cdot\overrightarrow{d_3^*} + r_2\cdot\overrightarrow{d_4^*})\rho}, \quad C' = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}).$$
If $C \neq C'$, the issuer aborts. Otherwise, the issuer sends $\overrightarrow{\sigma'}$ to the user.
4. The user computes $\overrightarrow{\sigma} = \overrightarrow{\sigma'}^{1/\rho}$ and checks if the signature is valid.

$\mathtt{Prove}(\mathsf{param}, \mathsf{pk}, m, \overrightarrow{\sigma})$: It first picks some appropriate $\mathsf{Open}_1, \mathsf{Open}_2$ and forms the following GS commitments:

$$M_Z = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}), \quad M_Y = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Y^m, \mathsf{Open}),$$
$$\Sigma_i = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, \sigma_i, \mathsf{Open}) \quad \text{for } i \in [1, 6].$$

Compute the following proof:

$$\pi = \mathsf{GS.Pf}\{(m, \overrightarrow{\sigma}) : \hat{e}_6(g_1^{s_1(\alpha-m)\overrightarrow{d_1} + s_1\overrightarrow{d_2} + s_2(\alpha-m)\overrightarrow{d_3} + s_2\overrightarrow{d_4}}, \overrightarrow{\sigma}) = \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}s_1}\},$$

using the commitment M_Z, M_Y and $\overrightarrow{\Sigma} = (\Sigma_1, \ldots, \Sigma_6)$ and for some randomly chosen $s_1, s_2 \in \mathbb{Z}_p$. It outputs $\mathsf{comm} = (M_Z, M_Y, \overrightarrow{\Sigma})$ and π.

$\mathtt{VerifyPf}(\mathsf{param}, \mathsf{pk}, \mathsf{comm}, \pi)$: It outputs accept if π is a valid NIZK proof for comm and the language above.

$\mathtt{EqCommProve}(\mathsf{param}, m, \mathsf{Open}, \mathsf{Open}')$: It forms the following GS commitments:

$$\mathsf{comm}_1 = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}),$$
$$\mathsf{comm}_2 = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}').$$

Compute the following proof:

$$\pi = \mathsf{GS.Pf}\{(m, \mathsf{Open}, \mathsf{Open}') : \quad M_1 = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}) \wedge \; M_2 = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open}')\}.$$

The zero-knowledge proof of equality of committed exponents can be found in [6]. It outputs $\mathsf{comm}_1, \mathsf{comm}_2$ and π.

$\mathtt{EqCommVerify}(\mathsf{param}, \mathsf{Open}, \mathsf{Open}', \pi)$: It outputs accept if π is a valid NIZK proof for comm_1 and comm_2 above.

Theorem 2. *Our P-signature is secure under the decision linear assumption and the security of the two-party computation.*

Its proof can be found in Appendix C.

Comparison. To the best of the authors' knowledge, the most efficient anonymous credential scheme based on standard assumption is to construct from the structure-preserving signatures (SPS) [1]. One may use GS proof to prove the possession of SPS or P-signatures as an anonymous credential. The efficient SPS scheme [1] based on the decision linear assumption has 17 group elements On the other hand, our underlying signature scheme FBB only has 6 group elements (under the same assumption). Even counting the commitment Y^m needed for converting FBB to P-signatures, our credential scheme is still more efficient.

Finally, we remark that it is not clear that whether the modified CL signature [20] can be used in anonymous credential [13], since their $\mathtt{Sign}_A$ and $\mathtt{Sign}_B$ algorithms [20] handle the message differently. In particular, the message appears in the signature produced by $\mathtt{Sign}_A$ algorithm twice but just once in $\mathtt{Sign}_B$, but the message should be hidden in a commitment and associated with various proof-of-knowledge when it is used in an anonymous credential system. There is no immediate simple solution without changing the scheme or the proof.

4.2 Signature on a Block of Messages and its Application

We give a modified construction for signing a block of messages, for anonymous credential supporting multiple attributes. Security proof is given in Appendix D.

$\mathtt{Setup}(1^\lambda)$: Same as $\mathtt{Setup}()$ in Section 3.2, except that $u_0, \ldots, u_\ell, h_1 \in \mathbb{G}_{p_1}$ are included in the public key instead of a single u_1 The public key includes $(g_1, u_0, \ldots, u_\ell, \hat{e}(g_1, h_1), g_1^\alpha)$. The secret key is $(h_1, \alpha, g_3, g_{2,3})$.

$\mathtt{Sign}(\mathsf{sk}, (m_1, \ldots, m_\ell))$: For a message in $\mathbb{Z}_N^\ell$, the signer randomly picks $r, e \in \mathbb{Z}_N$[6], $X_3, X_3' \in \mathbb{G}_{p_3}$ and computes the signature $\sigma = (\sigma_1, \sigma_2, e)$, where $\sigma_1 = (h_1(u_0 u_1^{m_1} \cdots u_\ell^{m_\ell})^{-r})^{\frac{1}{\alpha - e}} X_3$, and $\sigma_2 = g_1^r X_3'$.

$\mathtt{Verify}(\mathsf{pk}, \sigma, (m_1, , \ldots m_\ell))$: Given a signature $\sigma = (\sigma_1, \sigma_2, e)$, output 'true' if: $\hat{e}(g_1^\alpha \cdot g_1^{-e}, \sigma_1) \cdot \hat{e}(\sigma_2, u_0 u_1^{m_1} \cdots u_\ell^{m_\ell}) = \hat{e}(g_1, h_1)$.

Applying to Blacklistable Anonymous Credentials. By using a signature on committed values with efficient protocols for proving knowledge and equality of committed values, we can obtain anonymous credential schemes [13, 5, 4]. We can simply change the block of messages $u_0 u_1^{m_1} \cdots u_\ell^{m_\ell}$ into a Pedersen commitment $u_0 u_1^{m_1} \cdots u_\ell^{m_\ell} u_{\ell+1}^t$ for some random $t \in \mathbb{Z}_N$.

Recent blacklistable anonymous credential schemes [5, 4] used BBS+ signature, a variant of Boneh-Boyen short signature [9], as a basic building block in a blackbox manner. Our new signature has almost the same structure, so it can be applied to these systems directly, weakening the assumption to static ones.

5 Identity-Based Encryption

Now we give a dual system version of the Identity-Based Encryption (IBE) by Gentry [18], to be proven in Appendix F. Review of IBE is given in Appendix E.

5.1 Dual-Form Gentry IBE

$\mathtt{Setup}(1^\lambda)$: Same as $\mathtt{Setup}()$ in Section 3.2. The master public key includes $(g_1, u_1, \hat{e}(g_1, h_1), g_1^\alpha)$. The secret key is (h_1, α, g_3). The identity space $\mathcal{I}$ is $\mathbb{Z}_N$.

$\mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$: Random picks $r \in \mathbb{Z}_N$, $X_3, X_3' \in \mathbb{G}_{p_3}$, and outputs $(K_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - \mathsf{ID}}} X_3, K_2 = g_1^r X_3')$.

$\mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M)$: To encrypt a message $M \in \mathbb{G}_T$ for $\mathsf{ID} \in \mathbb{Z}_N \backslash \{\alpha\}$, randomly pick $s \in \mathbb{Z}_N$ and output $(C_0 = M \cdot \hat{e}(g_1, h_1)^s, C_1 = g_1^{s(\alpha - \mathsf{ID})}, C_2 = u_1^s)$.

$\mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathfrak{C})$: Decrypt (C_0, C_1, C_2) by (K_1, K_2) via $C_0 / \hat{e}(C_1, K_1) \cdot \hat{e}(C_2, K_2)$.

[6] If $\alpha - e$ has no inverse modulo N, then it picks another e until the inverse exists.

5.2 Accountable Authority Identity-Based Encryption

Our concrete IBE construction follows Gentry-IBE that the element r jointly computed by the user and the PKG, yet cannot be re-randomized in $\mathbb{G}_{p_1}$ by the user, so $K_2 = g_1^r X_3'$ is useful for white-box tracing [21]. The definitions for A-IBE can be found in Appendix G, which include the regular IND-ID-CCA security, and the ComputeNewKey/FindNewKey-CCA security for dishonest user/PKG.

Our Techniques. We need to *"anonymize"* our dual system Gentry-IBE by a composite group of order $N = p_1p_2p_3p_4$ where p_1, p_2, p_3, p_4 are all primes, which is output by $\mathcal{G}'(1^\lambda)$ (defined like that in Section 2). For our basic IBE, one can check if $\hat{e}(C_2, g_1^{\alpha-\mathsf{ID}}) = \hat{e}(u_1, C_1)$ for a valid ciphertext decryptable by ID. The ciphertext component C_2 is now u_{14}^s where $u_{14} \in \mathbb{G}_{p_1p_4}$ is put in mpk, and its $\mathbb{G}_{p_1}$ value u_1 should be kept secret to prevent the same checking.

The *IND-ID-CCA security* can be obtained by extending our IBE scheme to 2-level HIBE and applying strong one-time signatures [14]. However, it makes the analysis of the distribution of SF keys and SF ciphertexts more complicated. The adversary only obtains two elements with unknown $\mathbb{G}_{p_2}$ component in our IBE's proof, and only their pairwise independence is needed. In our A-IBE construction, the adversary will obtain five elements with $\mathbb{G}_{p_2}$ component.

For the FindNewKey-CCA security, the simulator acts as an honest user and he can only obtain an SF key after interacting with the dishonest PKG. The simulator can still use the SF key to simulate the decryption oracle. Finally, we submit an SF ciphertext to the tracing algorithm. The simulator should not be able to decrypt the SF ciphertext and hence it can only outputs a random message. If the dishonest PKG can find a normal key, then the simulator can solve some static assumptions related to the subgroup. We also need a knowledge extractor for a ZKPoK to get some randomness and a part of the msk for computing the correct response of an honest user.

Concrete Construction.

$\mathtt{Setup}(1^\lambda)$: The PKG runs the bilinear group generator $\mathcal{G}'(1^\lambda)$ to get $(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e})$ as defined in Section 2. Suppose $\mathcal{G}$ also gives generators g_1, g_3 and g_4 of the subgroups $\mathbb{G}_{p_1}$, $\mathbb{G}_{p_3}$ and $\mathbb{G}_{p_4}$ respectively. The PKG randomly picks $\alpha, \nu \in \mathbb{Z}_N$, $h_1, y_1, w_1, v_1 \in \mathbb{G}_{p_1}$, $u_4 \in \mathbb{G}_{p_4}$ and computes $u_1 = y_1^\nu, u_{14} = u_1u_4$. Denote $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify})$ as a strong one-time signature scheme. Denote $(\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ as an (interactive) concurrent zero-knowledge proof of knowledge [27]. It runs $\mathsf{crs} \leftarrow \mathsf{CRSGen}(1^\lambda)$. It computes:

$$\mathsf{param} = (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, w_1, v_1, g_3, g_4, \mathsf{crs}), \quad \mathsf{mpk} = (u_{14}, \hat{e}(g_1, h_1), g_1^\alpha).$$

The message space $\mathcal{M}$ is $\mathbb{G}_T$ and the identity space $\mathcal{I}$ is $\mathbb{Z}_N \backslash \{\alpha\}$. The master secret key is (h_1, α, ν). Hence $u_1 = y_1^\nu$ can be computed by the msk.

$\mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$: The user and the PKG interacts as follows:

1. The PKG picks some random $r_2 \in \mathbb{Z}_N$, $Z_3, Z_3' \in \mathbb{G}_{p_3}$ and sends (A_1, A_2) to the user, where $A_1 = u_1^{r_2} Z_3 = y_1^{\nu r_2} Z_3, A_2 = g_1^{r_2} Z_3'$.
2. The PKG runs an interactive concurrent zero-knowledge proof of knowledge π of (ν, r_2) such that A_1 and A_2 are properly formed in $\mathbb{G}_{p_1}$ with $\hat{e}(A_1, g_1R_4) = \hat{e}(u_{14}, A_2)$, where R_4 is randomly chosen from $\mathbb{G}_{p_4}$. It implies $\hat{e}(y_1^{\nu r_2}, g_1) = \hat{e}(u_1, g_1^{r_2})$ and hence $u_1 = y_1^\nu$. The user continues if he accepts the proof π.
3. The user picks some random $r_0, \theta_1 \in \mathbb{Z}_p$ and sends to the PKG: $R_1 = A_1^{-r_0}(g_1^{\alpha-\mathsf{ID}})^{\theta_1}$.
4. The PKG randomly picks $r_1, t_1 \in \mathbb{Z}_N$, $X_3, X_3', X_3'', X_3''' \in \mathbb{G}_{p_3}$ and computes

$$\hat{K}_1 = (h_1 \cdot R_1 \cdot u_1^{-r_1})^{\frac{1}{\alpha-\mathsf{ID}}} w_1^{t_1} X_3, \ \hat{K}_2 = g_1^{r_1} X_3', \ \hat{K}_3 = g_1^{t_1(\alpha-\mathsf{ID})} X_3'', \ \hat{K}_4 = v_1^{t_1} X_3'''.$$

5. The user randomly picks $Y_3, Y_3', Y_3'', Y_3''' \in \mathbb{G}_{p_3}$ and computes

$$K_1 = \hat{K}_1 Y_3 / g_1^{\theta_1}, \quad K_2 = \hat{K}_2 A_2^{r_0} Y_3', \quad K_3 = \hat{K}_3 Y_3'', \quad K_4 = \hat{K}_4 Y_3'''.$$

Then the user checks if $\hat{e}(K_1, g_1^{\alpha-\mathsf{ID}}) = \hat{e}(g_1, h_1) \cdot \hat{e}(u_{14}, K_2) \cdot \hat{e}(w_1, K_3)$ and $\hat{e}(K_3, v_1) = \hat{e}(g_1^{\alpha-\mathsf{ID}}, K_4)$. If so, he sets the identity-based secret key as (K_1, K_2, K_3, K_4) for the key family n_F, where n_F is the $\mathbb{G}_{p_1}$ part of K_2 (two different key family numbers can be distinguished by pairing with the same $\mathbb{G}_{p_1}$ element). He outputs $\perp$ otherwise.

$\mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M)$: To encrypt a message M for ID, the sender randomly picks $s \in \mathbb{Z}_N$, $X_4, X_4', X_4'' \in \mathbb{G}_{p_4}$ and runs $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$. It outputs $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ where

$$C_0 = M \cdot \hat{e}(g_1, h_1)^s,\ C_1 = g_1^{s(\alpha - \mathsf{ID})} X_4,\ C_2 = u_{14}^s X_4',\ C_3 = (v_1^{\mathsf{vk}} w_1)^s X_4'',\ \sigma = \mathsf{Sign}(\mathsf{sk}, C_0||C_1||C_2||C_3).$$

$\mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathfrak{C})$: Given a ciphertext $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ and a secret key $\mathsf{sk}_{\mathsf{ID}} = (K_1, K_2, K_3, K_4)$, the recipient checks if $\mathsf{Verify}(\mathsf{vk}, C_0||C_1||C_2||C_3, \sigma) = 1$. If not, it outputs $\perp$. Otherwise, it calculates: $M = \frac{C_0 \cdot \hat{e}(C_3, K_3)}{\hat{e}(C_1, K_1 K_4^{\mathsf{vk}}) \cdot \hat{e}(C_2, K_2)}$.

$\mathtt{Trace}^{\mathbb{D}}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \epsilon)$: Given a valid $\mathsf{sk}_{\mathsf{ID}} = (K_1, K_2, K_3, K_4)$ for a user ID and an ϵ-useful decoder box $\mathbb{D}$, it checks by the following steps:

1. Set $ctr \leftarrow 0$ and repeat the following steps for $L = 16\lambda/\epsilon$ times:
   (a) Choose $s, s' \leftarrow \mathbb{Z}_N$, $X_4, X_4', X_4'' \in \mathbb{G}_{p_4}$ such that $s \neq s'$ and runs $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$. Set $C_1 = g_1^{s(\alpha - \mathsf{ID})} X_4$, $C_2 = u_{14}^{s'} X_4'$ and $C_3 = (v_1^{\mathsf{vk}} w_1)^s X_4''$.
   (b) Compute $C_0 = M \cdot \hat{e}(C_1, K_1) \cdot \hat{e}(C_2, K_2) / \hat{e}(C_3, K_3)$ for a random message $M \in \mathbb{G}_T$ and $\sigma = \mathsf{Sign}(\mathsf{sk}, C_0||C_1||C_2||C_3)$.
   (c) Feed the decoder box $\mathbb{D}$ with $(C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$. If $\mathbb{D}$ outputs the same M, increment ctr.
2. If $ctr = 0$, incriminate the PKG. Otherwise, incriminate the user.

All security proofs are given in Appendix G.

6 Conclusion

We give a modified Boneh-Boyen signatures scheme which is secure in the standard model under static assumption, and further propose a dual-system variant of Gentry's IBE. Our proof technique can be applied to a number of advanced cryptosystems, which we showcase by anonymous credentials and accountable IBE. In particular, these two applications outperform the existing counterparts.

References

1. M. Abe, M. Chase, B. David, M. Kohlweiss, R. Nishimaki, and M. Ohkubo. Constant-Size Structure-Preserving Signatures: Generic Constructions and Simple Assumptions. In *ASIACRYPT*, LNCS 7658, pages 4–24, 2012.
2. M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo. Structure-Preserving Signatures and Commitments to Group Elements. In *CRYPTO*, LNCS 6223, pages 209–236, 2010.
3. B. Applebaum, Y. Ishai, and E. Kushilevitz. How to Garble Arithmetic Circuits. In *FOCS*, pages 120–129. IEEE, 2011.
4. M. H. Au and A. Kapadia. PERM: Practical Reputation-based Blacklisting without TTPs. In *CCS*, pages 929–940, 2012.
5. M. H. Au, A. Kapadia, and W. Susilo. BLACR: TTP-Free Blacklistable Anonymous Credentials with Reputation. In *NDSS*, 2012.
6. M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. P-signatures and Noninteractive Anonymous Credentials. In *TCC*, LNCS 4948, pages 356–374, 2008.
7. M. Bellare, V. T. Hoang, and P. Rogaway. Foundations of Garbled Circuits. In *CCS*, pages 784–796. ACM, 2012.
8. D. Boneh and X. Boyen. Efficient Selective-ID Secure Identity-Based Encryption Without Random Oracles. In *EUROCRYPT*, LNCS 3027, pages 223–238, 2004.
9. D. Boneh and X. Boyen. Short Signatures Without Random Oracles. In *EUROCRYPT*, LNCS 3027, pages 56–73, 2004.
10. D. Boneh, X. Boyen, and H. Shacham. Short Group Signatures. In *CRYPTO*, LNCS 3152, pages 41–55, 2004.
11. D. Boneh and M. K. Franklin. Identity-Based Encryption from the Weil Pairing. In *CRYPTO*, LNCS 2139, pages 213–229, 2001.

12. X. Boyen. General *Ad Hoc* Encryption from Exponent Inversion IBE. In *EUROCRYPT*, LNCS 4515, pages 394–411, 2007.
13. J. Camenisch and A. Lysyanskaya. Signature Schemes and Anonymous Credentials from Bilinear Maps. In *CRYPTO*, LNCS 3152, pages 56–72, 2004.
14. R. Canetti, S. Halevi, and J. Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *EUROCRYPT*, LNCS 3027, pages 207–222, 2004.
15. L. Chen and Z. Cheng. Security Proof of Sakai-Kasahara's Identity-Based Encryption Scheme. Cryptology ePrint Archive, Report 2005/226, 2005.
16. A. De Caro, V. Iovino, and G. Persiano. Fully Secure Anonymous HIBE and Secret-Key Anonymous IBE with Short Ciphertexts. In *Pairing*, LNCS 6487, pages 347–366, 2010.
17. C. Delerablée and D. Pointcheval. Dynamic Threshold Public-Key Encryption. In *CRYPTO*, LNCS 5157, pages 317–334, 2008.
18. C. Gentry. Practical Identity-Based Encryption Without Random Oracles. In *EUROCRYPT*, LNCS 4004, pages 445–464, 2006.
19. C. Gentry and S. Halevi. Hierarchical Identity Based Encryption with Polynomially Many Levels. In *TCC*, LNCS 5444, pages 437–456, 2009.
20. M. Gerbush, A. B. Lewko, A. O'Neill, and B. Waters. Dual Form Signatures: An Approach for Proving Security from Static Assumptions. In *ASIACRYPT*, LNCS 7658, pages 25–42, 2012.
21. V. Goyal. Reducing Trust in the PKG in Identity Based Cryptosystems. In *CRYPTO*, LNCS 4622, pages 430–447, 2007.
22. J. Groth. Fully Anonymous Group Signatures Without Random Oracles. In *ASIACRYPT*, LNCS 4833, pages 164–180, 2007.
23. D. Jao and K. Yoshida. Boneh-Boyen Signatures and the Strong Diffie-Hellman Problem. In *Pairing*, LNCS 5671, pages 1–16, 2009.
24. A. Lewko. Tools for Simulating Features of Composite Order Bilinear Groups in the Prime Order Setting. In *EUROCRYPT*, LNCS 7237, pages 318–335, 2012.
25. A. Lewko and B. Waters. New Techniques for Dual System Encryption and Fully Secure HIBE with Short Ciphertexts. In *TCC*, LNCS 5978, pages 455–479, 2010.
26. B. Libert and D. Vergnaud. Towards Black-Box Accountable Authority IBE with Short Ciphertexts and Private Keys. In *PKC*, LNCS 5443, pages 235–255, 2009.
27. R. Pass and M. Venkitasubramaniam. On Constant-Round Concurrent Zero-Knowledge. In *TCC*, LNCS 4948, pages 553–570, 2008.
28. A. Sahai and H. Seyalioglu. Fully Secure Accountable-Authority Identity-Based Encryption. In *PKC*, LNCS 6571, pages 296–316, 2011.
29. R. Sakai and M. Kasahara. ID based Cryptosystems with Pairing on Elliptic Curve. Cryptology ePrint Archive, Report 2003/054, 2003.
30. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *EUROCRYPT*, LNCS 3494, pages 114–127, 2005.
31. B. Waters. Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In *CRYPTO*, LNCS 5677, pages 619–636, 2009.

A Security Proof for Our Dual-Form Signatures

We denote our original signing algorithm by $\texttt{Sign}_A$, and its dual form is $\texttt{Sign}_B$.

$\texttt{Sign}_B(\texttt{sk}, m)$: The signer randomly picks $r \in \mathbb{Z}_N$, $X_{2,3}, X'_{2,3} \in \mathbb{G}_{p_2p_3}$ and computes the signature $\sigma = (\sigma_1, \sigma_2)$, where:

$$\sigma_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - m}} X_{2,3}, \quad \sigma_2 = g_1^r X'_{2,3}.$$

Forgery Classes. We let $z \in \mathbb{Z}_N$ denote the exponent represented by the tuple $(0 \bmod p_1; 1 \bmod p_2; 0 \bmod p_3)$. We then divide the forgery types based on whether they have a $\mathbb{G}_{p_2}$ component.

- **Type I**. $\mathcal{V}_I = \{(m^*, \sigma^*) \in \mathcal{V} | (\sigma_2^*)^z = 1 \text{ and } (\sigma_1^*)^z = 1\}$.
- **Type II**. $\mathcal{V}_{II} = \{(m^*, \sigma^*) \in \mathcal{V} | (\sigma_2^*)^z \neq 1 \text{ or } (\sigma_1^*)^z \neq 1\}$.

Lemma 1. *If Assumption 1 holds, then our signature scheme is A-I Matching.*

Proof. Suppose that there exists an adversary $\mathcal{A}$ that can create a forgery that is not of Type I with probability ϵ given access to an oracle for the $\texttt{Sign}_A$ algorithm. Then we can create an algorithm $\mathcal{B}$ that breaks Assumption 1 with advantage negligibly close to ϵ.

Given (g, X_3, T) from Assumption 1, $\mathcal{B}$ chooses random $b \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_1 = g^b$, $g_3 = X_3$. $\mathcal{B}$ generates the rest of pk according to $\texttt{Setup}$ and the secret key is only $\mathsf{sk} = (h_1, \alpha, g_3)$.

For the oracle queries to $\texttt{Sign}_A$, $\mathcal{B}$ calculates the signature using sk and answers the query since $g_{2,3}$ is not needed for $\texttt{Sign}_A$.

In the challenge phase, $\mathcal{A}$ returns a valid signature $(\sigma^* = (\sigma_1^*, \sigma_2^*), m^*)$ to $\mathcal{B}$. $\mathcal{B}$ then tries to use this forgery to determine if T is in $\mathbb{G}_{p_1}$ or $\mathbb{G}_{p_1 p_2}$. $\mathcal{B}$ sets:

$$C_0^* = \hat{e}(T, h_1), \quad C_1^* = T^{\alpha - m^*}, \quad C_2^* = T^b,$$

and proceeds with a *backdoor verification* test by checking the following equality,

$$C_0^* \stackrel{?}{=} \hat{e}(C_1^*, \sigma_1^*) \cdot \hat{e}(C_2^*, \sigma_2^*).$$

If it does not hold, $\mathcal{B}$ outputs 1. If it does, $\mathcal{B}$ flips a coin $b \in \{0, 1\}$ and return b.

Since it is guaranteed that (σ^*, m^*) passes the real signature verification, we know that it will pass this backdoor verification if $T \in \mathbb{G}_{p_1}$, no matter $\mathcal{A}$ returns Type I or Type II forgery. Next, we consider the case for $T \in \mathbb{G}_{p_1 p_2}$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is non-zero modulus p_2. The backdoor verification equation proceeds as follows,

$$\hat{e}(C_1^*, \sigma_1^*)\hat{e}(C_2^*, \sigma_2^*) = C_0^* \hat{e}(\hat{g}_2^{\delta_1}, Y_2^{\alpha - m^*})\hat{e}(\hat{g}_2^{\delta_2}, Y_2^b) = C_0^* \hat{e}(\hat{g}_2, Y_2)^{\delta_1(\alpha - m^*) + \delta_2 b} \stackrel{?}{=} C_0^*.$$

If $\delta_1(\alpha - m^*) + \delta_2 b \neq 0 \bmod p_2$, then the equality does not hold and $\mathcal{B}$ will output 1. It remains to consider a Type II forgery which will pass the backdoor verification. In this case, the adversary must find some δ_1, δ_2 such that $\delta_1(\alpha - m^*) = -\delta_2 b \bmod p_2$ (under the restriction that either δ_1 or δ_2 is non-zero modulus p_2). Since b modulo p_2 is not revealed at any point during the query phase, so there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test. Upon receiving such a forgery, $\mathcal{B}$ will output 1 with probability 1/2. The advantage of $\mathcal{B}$ is

$$\begin{aligned} &\Big| \Pr[\mathcal{B}(D, T_0) = 1] - \Pr[\mathcal{B}(D, T_1) = 1] \Big| \\ =&\left| \epsilon\big(\delta' \cdot \frac{1}{2} + (1 - \delta')1\big) + (1 - \epsilon)\frac{1}{2} - \frac{1}{2} \right| \geq \frac{1}{2}\epsilon - \frac{1}{2}\epsilon\delta'. \end{aligned}$$

Thus, if ϵ is non-negligible, then $\mathcal{B}$ has non-negligible advantage against the Assumption 1 challenger. □

Lemma 2. *If Assumption 2 holds, our scheme satisfies dual-oracle invariance.*

Proof. Given $(g, X_1X_2, X_3, Y_2Y_3, T)$ from Assumption 2, $\mathcal{B}$ chooses random $b, \alpha \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_1 = g^b$, $g_{2,3} = Y_2Y_3$ and $g_3 = X_3$. $\mathcal{B}$ generates the rest of pk and the secret key $\mathsf{sk} = (h_1, \alpha, g_3, g_{2,3})$ according to $\texttt{Setup}$.

For the oracle queries to $\texttt{Sign}_A$, $\mathcal{B}$ randomly picks $r, w, v \in \mathbb{Z}_N$ and uses sk to compute the signature $\sigma = (\sigma_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - m}} X_3^w, \sigma_2 = g_1^r X_3^v)$.

For the oracle queries to $\texttt{Sign}_B$, $\mathcal{B}$ randomly picks $r, w, v \in \mathbb{Z}_N$ and computes the signature $\sigma = (\sigma_1, \sigma_2)$, where:

$$\sigma_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - m}} (Y_2Y_3)^w, \quad \sigma_2 = g_1^r (Y_2Y_3)^v.$$

By the Chinese remainder theorem, the values of v and w modulo p_2 and those modulo p_3 are uncorrelated.

Finally, $\mathcal{A}$ will query $\mathcal{B}$ on some challenge message m. $\mathcal{B}$ will choose some random $w, v \in \mathbb{Z}_N$, and calculates the signature:

$$\sigma_1 = h_1^{\frac{1}{\alpha - m}} \cdot T^{\frac{-b}{\alpha - m}} \cdot X_3^w, \quad \sigma_2 = T \cdot X_3^v.$$

If $T = Z_1 Z_3$, it is a signature from $\texttt{Sign}_A$ (by considering $g^r = Z_1$). If $T = Z_1 Z_2 Z_3$, it is from $\texttt{Sign}_B$ (b modulo p_2 is not revealed at any point during the query phase). So the $\mathbb{G}_{p_2}$ part of σ_1^* is randomly distributed from the view of $\mathcal{A}$.

Once A returns the forgery, (σ^*, m^*), $\mathcal{B}$ must first check that $\mathcal{A}$ has not seen a signature for m^* before and that (σ^*, m^*) verifies. If either of these checks fail then $\mathcal{B}$ will guess randomly. If both of these are true, then $\mathcal{B}$ determines what forgery class (σ^*, m^*) belongs to in order to determine what subgroup T is in, via a backdoor verification test similar to that in the previous proof. $\mathcal{B}$ sets:

$$C_0^* = \hat{e}(X_1 X_2, h_1), \quad C_1^* = (X_1 X_2)^{\alpha - m^*}, \quad C_2^* = (X_1 X_2)^b.$$

Denote $\sigma^* = (\sigma_1^*, \sigma_2^*)$. $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_0^* \stackrel{?}{=} \hat{e}(C_1^*, \sigma_1^*) \cdot \hat{e}(C_2^*, \sigma_2^*).$$

If it does not hold, $\mathcal{B}$ outputs 1. If it does, $\mathcal{B}$ flips a coin $b \in \{0, 1\}$ and return b.

If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test since it passes the real signature verification. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is non-zero modulus p_2. Then the backdoor verification equation proceeds as follows,

$$\hat{e}(C_1^*, \sigma_1^*) \cdot \hat{e}(C_2^*, \sigma_2^*) = C_0^* \cdot \hat{e}(\hat{g}_2^{\delta_1}, X_2^{\alpha - m^*}) \cdot \hat{e}(\hat{g}_2^{\delta_2}, X_2^b) = C_0^* \cdot \hat{e}(\hat{g}_2, X_2)^{\delta_1(\alpha - m^*) + \delta_2 b} \stackrel{?}{=} C_0^*.$$

Thus, if the forgery fails the test, then with probability 1 it is a Type II forgery. If the forgery passes the test then it can be either Type I or Type II. We claim that a Type II forgery can also pass the additional verification test, but only with negligible probability.

For such a Type II forgery, we have $\delta_1(\alpha - m^*) + \delta_2 b = 0 \bmod p_2$. Consider:

1. If $\delta_1 = 0 \bmod p_2$ and $\delta_2 \neq 0 \bmod p_2$, it implies $b = 0 \bmod p_2$. It happens with negligible probability since b is randomly chosen by $\mathcal{B}$ from $\mathbb{Z}_N$.
2. If $\delta_1 \neq 0 \bmod p_2$, we rewrite the equation as $(\alpha - m^*) + \delta b = 0 \bmod p_2$, where $\delta = \delta_2/\delta_1$. In order to create such a Type II forgery, an adversary must implicitly determine $(\alpha - m^*)/b$ modulo p_2. The adversary only knows $b/(\alpha - m)$ modulo p_2 from the challenge signature if $T = Z_1 Z_2 Z_3$. As long as $m \neq m^*$ modulo p_2, the adversary has no better than the negligible probability of achieving the correct value of δ modulo p_2.

We now consider the information obtained by the adversary. In the challenge signature, α and b modulo p_2 are only included in the first element of the challenge signature. Thus the attacker can only derive the single value $\frac{b}{\alpha - m}$ modulo p_2. However, this single equation has two unknowns α and b modulo p_2 and it is not possible to determine their unique values. Moreover, $\frac{b}{\alpha - m}$ is a pairwise independent function of m modulo p_2 (except with negligible probability that $\alpha = m \bmod p_2$). Therefore, the attacker cannot achieve the correct value of $\frac{b}{\alpha - m^*} \bmod p_2$ as long as $m \neq m^* \bmod p_2$, except with negligible probability. It is possible that $m = m^*$ modulo p_2, but $m \neq m^*$ modulo N. If this occurs with non-negligible probability, then $\mathcal{B}$ can extract a non-trivial factor of N by computing the greatest common divisor of N and $m - m^*$, and use it to break Assumption 2 with non-negligible advantage. Hence, if a forgery passes the additional verification test, then with high probability it is a Type I forgery. □

Lemma 3. *If Assumption 3 holds, then our signature scheme is B-II Matching.*

Proof. Suppose that there exists an adversary, $\mathcal{A}$, that can create a Type I forgery with non-negligible probability ϵ given access to an oracle for the $\texttt{Sign}_B$ algorithm. Then we can create an algorithm $\mathcal{B}$ that breaks Assumption 3 with non-negligible advantage.

Given $(g, g^a X_2, g^s Y_2, Z_2, X_3, T)$ from Assumption 3, $\mathcal{B}$ chooses random $b, \alpha \in \mathbb{Z}_N$ and sets

$$g_1 = g, \quad u_1 = g^b, \quad \hat{e}(g_1, h_1) = \hat{e}(g, g^a X_2).$$

$\mathcal{B}$ implicitly sets $h_1 = g^a$. $\mathcal{B}$ sends the public key $\texttt{pk}$ to $\mathcal{A}$.

$\mathcal{B}$ can answer the $\texttt{Sign}_B$ oracle as follows. $\mathcal{B}$ randomly picks $r \in \mathbb{Z}_N$, $R_2, R_2' \in \mathbb{G}_{p_2}$ and $R_3, R_3' \in \mathbb{G}_{p_3}$ and answers by:

$$\sigma_1 = (g^a X_2 \cdot u_1^{-r})^{\frac{1}{\alpha - m}} \cdot R_2 \cdot R_3, \quad \sigma_2 = g^r \cdot R_2' \cdot R_3'.$$

After the query phase, A will output a valid forgery, (σ^*, m^*). $\mathcal{B}$ can use this forgery for a backdoor verification test similar to the previous proofs to determine whether $T = \hat{e}(g, g)^{as}$. First, $\mathcal{B}$ sets:

$$C_0^* = T, \quad C_1^* = (g^s Y_2)^{\alpha - m^*}, \quad C_2^* = (g^s Y_2)^b.$$

Since α and b are chosen randomly modulo N, there will be no correlation between the $\mathbb{G}_{p_1}$ and the $\mathbb{G}_{p_2}$ components of C_1^* and C_3^*. Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_0^* \stackrel{?}{=} \hat{e}(C_1^*, \sigma_1^*) \cdot \hat{e}(C_2^*, \sigma_2^*).$$

If it does hold, $\mathcal{B}$ outputs 1. Otherwise, $\mathcal{B}$ flips a coin $b \in \{0, 1\}$ and returns b.

If T is a random group element in $\mathbb{G}_T$, it will not pass this verification equation (no matter $\mathcal{A}$ returns Type I or Type II forgery) with all but with some negligible probability, δ''. In this case $\mathcal{B}$ will output 1 with probability 1/2.

Next we consider the case for $T = \hat{e}(g, g)^{as}$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is non-zero modulus p_2. Then the backdoor verification equation proceeds as follows,

$$\hat{e}(C_1^*, \sigma_1^*) \cdot \hat{e}(C_2^*, \sigma_2^*) = C_0^* \hat{e}(\hat{g}_2^{\delta_1}, Y_2^{\alpha - m^*}) \hat{e}(\hat{g}_2^{\delta_2}, Y_2^b) = C_0^* \hat{e}(\hat{g}_2, Y_2)^{\delta_1(\alpha - m^*) + \delta_2 b} \stackrel{?}{=} C_0^*.$$

If $\delta_1(\alpha - m^*) + \delta_2 b \neq 0 \bmod p_2$, the test always fails and $\mathcal{B}$ will output 1 with probability 1/2. Else, $\mathcal{B}$ will output 1. However for an adversary to create such a Type II forgery, it must find some δ_1, δ_2 such that $\delta_1(\alpha - m^*) = -\delta_2 b \bmod p_2$. There are two cases:

1. If $\delta_2 = 0 \bmod p_2$, it implies $\alpha = m^* \bmod p_2$, since it is restricted that $\delta_1 \neq 0 \bmod p_2$ in this case. However α modulo p_2 is not revealed at any point during the query phase.
2. If $\delta_2 \neq 0 \bmod p_2$, it means that the adversary has to find some δ such that $\delta(\alpha - m^*) = -b \bmod p_2$. However b modulo p_2 is not revealed at any point during the query phase.

In both cases, there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test.

The advantage of $\mathcal{B}$ against the Assumption 3 challenger is

$$\begin{aligned}
&\Big| \Pr[\mathcal{B}(D, T_0) = 1] - \Pr[\mathcal{B}(D, T_1) = 1] \Big| \\
=&\Big| \Pr[\mathcal{B}(D, T_0) = 1] - \big(\delta'' \cdot \Pr[\mathcal{B}(D, T_0) = 1] + (1 - \delta'') \cdot \frac{1}{2}\big) \Big| \\
=&\Big| (1 - \delta'')\big(\Pr[\mathcal{B}(D, T_0) = 1] - \frac{1}{2}\big) \Big| \\
=&(1 - \delta'')\big(\epsilon + (1 - \epsilon)(\delta' + (1 - \delta') \cdot \frac{1}{2}) - \frac{1}{2}\big) \\
=&(1 - \delta'')(\frac{\epsilon}{2} + \frac{\delta'}{2} - \frac{\epsilon \delta'}{2}).
\end{aligned}$$

If ϵ is non-negligible, so does the advantage of $\mathcal{B}$ against its challenger. □

Combining the above three lemmata and Theorem 1, we have:

Theorem 3. *Our dual form Boneh-Boyen signature is existentially unforgeable under an adaptive chosen message attack if Assumptions 1, 2 and 3 hold.*

B Our Signature Scheme in Prime Order Bilinear Groups

We can convert our basic scheme in Section 3.2 into the prime order setting by Lewko's method [24]. We first review the definition of dual orthonormal bases.

Dual Pairing Vector Spaces. For a fixed dimension n, we will choose two random bases $\mathbb{B} := (\overrightarrow{b_1}, \ldots \overrightarrow{b_n})$ and $\mathbb{B}^* := (\overrightarrow{b_1^*}, \ldots \overrightarrow{b_n^*}))$ of $\mathbb{Z}_p^n$, subject to the constraint that they are "dual orthonormal", meaning that $\overrightarrow{b_i} \cdot \overrightarrow{b_j^*} = 0 \bmod p$ for all $i \neq j$, and $\overrightarrow{b_i} \cdot \overrightarrow{b_i^*} = \psi$ for all i, where ψ is a uniformly random element of $\mathbb{Z}_p$

We define $\hat{e}_n$ to denote the product of the component-wise pairings for vectors $\overrightarrow{v} = (v_1, \ldots, v_n)$, $\overrightarrow{w} = (w_1, \ldots, w_n)$:

$$\hat{e}_n(g^{\overrightarrow{v}}, g^{\overrightarrow{w}}) := \prod_{i=1}^{n} \hat{e}(g^{v_i}, g^{w_i}) = \hat{e}(g, g)^{\overrightarrow{v} \cdot \overrightarrow{w}}.$$

Choosing random dual orthonormal bases $(\mathbb{B}, \mathbb{B}^*)$ can equivalently be thought of as choosing a random basis $\mathbb{B}$, choosing a random vector $\overrightarrow{b_1^*}$ subject to the constraint that it is orthogonal to $\overrightarrow{b_2}, \ldots, \overrightarrow{b_2}$, defining $\psi = \overrightarrow{b_1} \cdot \overrightarrow{b_1^*}$, and then choosing $\overrightarrow{b_2^*}$ so that it is orthogonal to $\overrightarrow{b_1}, \overrightarrow{b_3}, \ldots, \overrightarrow{b_2}$, and has dot product with $\overrightarrow{b_2}$ equal to ψ, and so on.

For a fixed dimension n and prime p, we let $(\mathbb{B}, \mathbb{B}^*) \leftarrow \mathit{Dual}(\mathbb{Z}_p^n)$ denote choosing random dual orthonormal bases $\mathbb{B}$ and $\mathbb{B}^*$ of $\mathbb{Z}_p^n$.

Decision Linear Assumption. Given a prime order bilinear group generator $\mathcal{G}$, we define the following distribution:

$$(p, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g, f, v \xleftarrow{R} \mathbb{G}, \quad c_1, c_2 \xleftarrow{R} \mathbb{Z}_p, \quad T_0 := g^{c_1+c_2}, \quad T_1 \xleftarrow{R} \mathbb{G}.$$
$$D := (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, f, v, f^{c_1}, v^{c_2}).$$

Assume that for any PPT algorithm $\mathcal{A}$ with output in $\{0, 1\}$, the advantage

$$\mathit{Adv}_{\mathcal{G},\mathcal{A}} := |\Pr[(D, T_0) = 1] - \Pr[(D, T_1) = 1]| = \mathit{neg}(\lambda).$$

(k, n)-Subspace Assumption [24]. Given a prime order bilinear group generator $\mathcal{G}$, we define the following distribution:

$$(p, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad (\mathbb{B}, \mathbb{B}^*) \leftarrow \mathit{Dual}(\mathbb{Z}_p^n),$$
$$g \xleftarrow{R} \mathbb{G}, \quad \nu, \beta, \tau_1, \tau_2, \tau_3, \mu_1, \mu_2, \mu_3 \xleftarrow{R} \mathbb{Z}_p,$$
$$U_1 := g^{\mu_1 \overrightarrow{b_1} + \mu_2 \overrightarrow{b_{k+1}} + \mu_3 \overrightarrow{b_{2k+1}}}, \quad U_2 := g^{\mu_1 \overrightarrow{b_2} + \mu_2 \overrightarrow{b_{k+2}} + \mu_3 \overrightarrow{b_{2k+2}}}, \quad \ldots,$$
$$U_k := g^{\mu_1 \overrightarrow{b_k} + \mu_2 \overrightarrow{b_{2k}} + \mu_3 \overrightarrow{b_{3k}}},$$
$$V_1 := g^{\tau_1 \nu \overrightarrow{b_1^*} + \tau_2 \beta \overrightarrow{b_{k+1}^*}}, \quad V_2 := g^{\tau_1 \nu \overrightarrow{b_2^*} + \tau_2 \beta \overrightarrow{b_{k+2}^*}}, \quad \ldots, \quad V_k := g^{\tau_1 \nu \overrightarrow{b_k^*} + \tau_2 \beta \overrightarrow{b_{2k}^*}},$$
$$W_1 := g^{\tau_1 \nu \overrightarrow{b_1^*} + \tau_2 \beta \overrightarrow{b_{k+1}^*} + \tau_3 \overrightarrow{b_{2k+1}^*}}, \quad W_2 := g^{\tau_1 \nu \overrightarrow{b_2^*} + \tau_2 \beta \overrightarrow{b_{k+2}^*} + \tau_3 \overrightarrow{b_{2k+2}^*}}, \quad \ldots,$$
$$W_k := g^{\tau_1 \nu \overrightarrow{b_k^*} + \tau_2 \beta \overrightarrow{b_{2k}^*} + \tau_3 \overrightarrow{b_{3k}^*}},$$
$$D := (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g^{\overrightarrow{b_1}}, \ldots, g^{\overrightarrow{b_{2k}}}, g^{\overrightarrow{b_{3k+1}}}, \ldots,$$
$$g^{\overrightarrow{b_n}}, g^{\nu \overrightarrow{b_1^*}}, \ldots, g^{\nu \overrightarrow{b_k^*}}, g^{\beta \overrightarrow{b_{k+1}^*}}, \ldots, g^{\beta \overrightarrow{b_{2k}^*}}, g^{\overrightarrow{b_{2k+1}^*}}, \ldots, g^{\overrightarrow{b_n^*}}, U_1, \ldots, U_k, \mu_3).$$

Assume that for any PPT algorithm $\mathcal{A}$ with output in $\{0,1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}} := |\Pr[(D, V_1, \ldots, V_k) = 1] - \Pr[(D, W_1, \ldots, W_k) = 1]| = neg(\lambda).$$

The decision linear assumption implies the subspace assumption [24].

B.1 Prime-Order Group Version of Our Signature Scheme

$\mathtt{Setup}(1^\lambda)$: It runs the bilinear group generator $\mathcal{G}(1^\lambda)$ to get $(p, \mathbb{G}, \mathbb{G}_T, \hat{e})$ where $\mathbb{G}$ has prime order p and g is a generator of $\mathbb{G}$. It samples random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$ as defined in [24]. We let $\overrightarrow{d_1}, \ldots, \overrightarrow{d_6}$ denote the elements of $\mathbb{D}$ and $\overrightarrow{d_1^*}, \ldots, \overrightarrow{d_6^*}$ denote the elements of $\mathbb{D}^*$. The signer randomly picks $\alpha, \gamma, \theta, \delta, z_1, z_3, z_5, y_1, y_3, y_5 \in \mathbb{Z}_p$. The public key is

$$(p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g^{\overrightarrow{d_1}}, \ldots, g^{\overrightarrow{d_4}}, g^{\alpha\overrightarrow{d_1}}, g^{\alpha\overrightarrow{d_3}}, \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}},$$
$$Z = g^{z_1\overrightarrow{d_1}+z_3\overrightarrow{d_3}+z_5\overrightarrow{d_5}}, Y = g^{y_1\overrightarrow{d_1}+y_3\overrightarrow{d_3}+y_5\overrightarrow{d_5}})$$

where the elements Y and Z are only used for the proof of F-unforgeability only. The message space $\mathcal{M}$ is $\mathbb{G}_T$ and the identity space $\mathcal{I}$ is $\mathbb{Z}_p$. The secret key is

$$(g^{\theta\overrightarrow{d_1^*}}, g^{\theta\overrightarrow{d_2^*}}, g^{\delta\overrightarrow{d_3^*}}, g^{\delta\overrightarrow{d_4^*}}, \alpha, \gamma).$$

$\mathtt{Sign}(\mathsf{sk}, m)$: It randomly picks $r_1, r_2 \in \mathbb{Z}_p$, and computes the signature

$$\overrightarrow{\sigma} = g^{(\frac{\gamma - r_1}{\alpha - m})\theta\overrightarrow{d_1^*} + r_1\theta\overrightarrow{d_2^*} + (\frac{-r_2}{\alpha-m})\delta\overrightarrow{d_3^*} + r_2\delta\overrightarrow{d_4^*}}.$$

$\mathtt{Verify}(\mathsf{pk}, \overrightarrow{\sigma}, m)$: The recipient randomly picks $s_1, s_2 \in \mathbb{Z}_p$, calculates $\overrightarrow{C} = g_1^{s_1(\alpha-m)\overrightarrow{d_1}+s_1\overrightarrow{d_2}+s_2(\alpha-m)\overrightarrow{d_3}+s_2\overrightarrow{d_4}}$ and checks if

$$\hat{e}_6(\overrightarrow{C}, \overrightarrow{\sigma}) = \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}s_1}.$$

Security Proof. We denote the original signing algorithm as $\mathtt{Sign}_A$, and its dual form is $\mathtt{Sign}_B(\mathsf{sk}, m)$: It randomly picks $r_1, r_2, t_5, t_6 \in \mathbb{Z}_p$, and computes the signature

$$\overrightarrow{\sigma} = g^{(\frac{\gamma - r_1}{\alpha - m})\theta\overrightarrow{d_1^*} + r_1\theta\overrightarrow{d_2^*} + (\frac{-r_2}{\alpha-m})\delta\overrightarrow{d_3^*} + r_2\delta\overrightarrow{d_4^*} + t_5\overrightarrow{d_5^*} + t_6\overrightarrow{d_6^*}}.$$

Forgery Classes of Our Scheme. We will divide the forgery types based on whether the signature has a $\overrightarrow{d_5^*}$ or $\overrightarrow{d_6^*}$ in the component. Then we can define the forgery classes as follows:

- **Type I**. $\mathcal{V}_I = \{(m^*, \overrightarrow{\sigma^*}) \in \mathcal{V} | \hat{e}_6(g^{\overrightarrow{d_5}}, \overrightarrow{\sigma^*}) = 1 \text{ and } \hat{e}_6(g^{\overrightarrow{d_6}}, \overrightarrow{\sigma^*}) = 1\}$.
- **Type II**. $\mathcal{V}_{II} = \{(m^*, \overrightarrow{\sigma^*}) \in \mathcal{V} | \hat{e}_6(g^{\overrightarrow{d_5}}, \overrightarrow{\sigma^*}) \neq 1 \text{ or } \hat{e}_6(g^{\overrightarrow{d_6}}, \overrightarrow{\sigma^*}) \neq 1\}$.

We will show that our new signature scheme is secure by showing that it satisfies the three properties of a secure dual form signature scheme.

Lemma 4. *If the subspace assumption holds for $k = 2, n = 6$, then our signature scheme is A-I Matching.*

Proof. Suppose that there exists an adversary $\mathcal{A}$ that can create a forgery that is not of Type I with probability ϵ given access to an oracle for the $\mathtt{Sign}_A$ algorithm. Then we can create an algorithm $\mathcal{B}$ that breaks the subspace assumption with advantage negligibly close to ϵ.

Given $D = (g^{\overrightarrow{b_1}}, \ldots, g^{\overrightarrow{b_4}}, g^{\nu\overrightarrow{b_1^*}}, g^{\nu\overrightarrow{b_2^*}}, g^{\beta\overrightarrow{b_3^*}}, g^{\beta\overrightarrow{b_4^*}}, g^{\overrightarrow{b_5^*}}, g^{\overrightarrow{b_6^*}}, U_1, U_2, \mu_3)$ along with T_1, T_2 from the subspace assumption, $\mathcal{B}$ tries to decide if T_1, T_2 are distributed as $(V_1, V_2) = (g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}}, g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}})$

or as (W_1, W_2) which is in the form of $(g^{\tau_1\nu\overrightarrow{b_1}+\tau_2\beta\overrightarrow{b_3}+\tau_3\overrightarrow{b_5}}, g^{\tau_1\nu\overrightarrow{b_2}+\tau_2\beta\overrightarrow{b_4}+\tau_3\overrightarrow{b_6}})$. In this proof, the terms $g^{\overrightarrow{b_5^*}}, g^{\overrightarrow{b_6^*}}, U_1, U_2, \mu_3$ are not used.

$\mathcal{B}$ first chooses a random matrix $A \in \mathbb{Z}_p^{2\times 2}$ (with all but negligible probability, A is invertible). We define dual orthonormal bases $\mathbb{F}, \mathbb{F}^*$ by:

$$\begin{array}{llllll}
\overrightarrow{f_1} = \nu\overrightarrow{b_1^*}, & \overrightarrow{f_2} = \nu\overrightarrow{b_2^*}, & \overrightarrow{f_3} = \beta\overrightarrow{b_3^*}, & \overrightarrow{f_4} = \beta\overrightarrow{b_4^*}, & \overrightarrow{f_5} = \overrightarrow{b_5^*}, & \overrightarrow{f_6} = \overrightarrow{b_6^*}, \\
\overrightarrow{f_1^*} = \nu^{-1}\overrightarrow{b_1}, & \overrightarrow{f_2^*} = \nu^{-1}\overrightarrow{b_2}, & \overrightarrow{f_3^*} = \beta^{-1}\overrightarrow{b_3}, & \overrightarrow{f_4^*} = \beta^{-1}\overrightarrow{b_4}, & \overrightarrow{f_5^*} = \overrightarrow{b_5}, & \overrightarrow{f_6^*} = \overrightarrow{b_6},
\end{array}$$

Now $\mathcal{B}$ implicitly sets $\mathbb{D} = \mathbb{F}_A, \mathbb{D}^* = \mathbb{F}_A^*$, where A is applied as a change of basis matrix to $\overrightarrow{f_5}, \overrightarrow{f_6}$ and the transpose of A^{-1} is applied as a change of basis matrix to $\overrightarrow{f_5^*}, \overrightarrow{f_6^*}$, as described in [24][Section 3.1]. It implies that $\overrightarrow{d_i} = \overrightarrow{f_i}$ and $\overrightarrow{d_i^*} = \overrightarrow{f_i^*}$ in the public key. The distribution of $\mathbb{D}, \mathbb{D}^*$ after the change of basis matrix A is correct and reveal no information about A as shown in [24][Lemma 3].

$\mathcal{B}$ chooses random $\alpha, \gamma, \theta', \delta'$ and implicitly sets $\theta = \theta'\nu, \delta = \delta'\beta$ by

$$\hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\overrightarrow{d_1^*}} = \hat{e}_6(g^{\overrightarrow{b_1}}, g^{\nu\overrightarrow{b_1^*}})^{\gamma\theta'}.$$

$\mathcal{B}$ can produce $\mathsf{pk} = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g^{\overrightarrow{d_1}}, \ldots, g^{\overrightarrow{d_4}}, g^{\alpha\overrightarrow{d_1}}, g^{\alpha\overrightarrow{d_3}}, \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\overrightarrow{d_1^*}})$ and $\mathsf{sk} = (g^{\theta\overrightarrow{d_1^*}}, g^{\theta\overrightarrow{d_2^*}}, g^{\delta\overrightarrow{d_3^*}}, g^{\delta\overrightarrow{d_4^*}}, \alpha, \gamma)$.

In the challenge phase, $\mathcal{A}$ returns $(\overrightarrow{\sigma^*}, m^*)$ to $\mathcal{B}$. First, $\mathcal{B}$ will check that the forgery verifies, if not then $\mathcal{B}$ will output $b \in \{0,1\}$ uniformly at random. If the forgery verifies, then $\mathcal{B}$ sets:

$$C_1^* = \hat{e}_6(T_2, g^{\overrightarrow{b_1}})^{\theta'\gamma} = \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}\tau_1}, \quad \overrightarrow{C_2^*} = T_2(T_1)^{(\alpha-m^*)}.$$

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_1^* = \hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}).$$

If this equality is false, then $\mathcal{B}$ will output 1. If the equality is true, then $\mathcal{B}$ will flip a coin $b \in \{0,1\}$ and return b.

Since it is guaranteed that (σ^*, m^*) passes the real verification test, we know that it will pass this verification equation with if T_1, T_2 are equal to $g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}}, g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}}$ (no matter $\mathcal{A}$ returns Type I or Type II forgery). Next we consider the case for T_1, T_2 are equal to $g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}+\tau_3\overrightarrow{b_5^*}}, g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}+\tau_3\overrightarrow{b_6^*}}$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose $\hat{e}_6(g^{\overrightarrow{d_5}}, \overrightarrow{\sigma^*}) = \hat{e}(g,g)^{\zeta_5\overrightarrow{d_5}\cdot\overrightarrow{d_5^*}}$ or $\hat{e}_6(g^{\overrightarrow{d_6}}, \overrightarrow{\sigma^*}) = \hat{e}(g,g)^{\zeta_6\overrightarrow{d_6}\cdot\overrightarrow{d_6^*}}$, for some $\zeta_5, \zeta_6 \in \mathbb{Z}_p$ and either ζ_5 or ζ_6 is not equal to zero modulus p. Then the backdoor verification equation proceeds as follows,

$$\hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}) = \hat{e}_6(g^{(\alpha-m^*)(\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}+\tau_3\overrightarrow{b_5^*})+(\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}+\tau_3\overrightarrow{b_6^*})}, \overrightarrow{\sigma^*}) \stackrel{?}{=} C_1^*.$$

Note that the coefficients of $\overrightarrow{C_2^*}$ in the basis $\overrightarrow{b_5^*}, \overrightarrow{b_6^*}$ form the vector $(\tau_3(\alpha - m^*), \tau_3)$. To convert it into the basis $\overrightarrow{d_5}, \overrightarrow{d_6}$, the corresponding coefficient becomes $\tau_3 A^{-1}(\alpha - m^*, 1)$, where the matrix A was not revealed at any point during the game. If the dot product of (ζ_5, ζ_6) and $\tau_3 A^{-1}(\alpha - m^*, 1)$ is not equal to zero, then the test always fails and $\mathcal{B}$ will output 1. Else, $\mathcal{B}$ will output 1 with probability 1/2. However for an adversary to create a Type II forgery that passes the backdoor verification test, it must find some ζ_5, ζ_6 such that the dot product of (ζ_5, ζ_6) and $\tau_3 A^{-1}(\alpha - m^*, 1)$ is equal to zero. Since the matrix A is uniformly random, (ζ_5, ζ_6) can only be guessed by $\mathcal{A}$. So there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test.

Thus, we can calculate the advantage of $\mathcal{B}$ against the subspace assumption challenger

$$\begin{aligned}&\left|\Pr[\mathcal{B}(D,W_1,W_2)=1]-\Pr[\mathcal{B}(D,V_1,V_2)=1]\right|\\=&\left|\epsilon\big(\delta'\cdot\frac{1}{2}+(1-\delta')1\big)+(1-\epsilon)\frac{1}{2}-\frac{1}{2}\right|\geq\frac{1}{2}\epsilon-\frac{1}{2}\epsilon\delta'.\end{aligned}$$

Thus, if ϵ is non-negligible, then $\mathcal{B}$ has non-negligible advantage against the subspace assumption challenger. □

Lemma 5. *If the subspace assumption holds with $k=2,n=6$, then our signature scheme satisfies dual-oracle invariance.*

Proof. Given $D=(g^{\overrightarrow{b_1}},\ldots,g^{\overrightarrow{b_4}},g^{\nu\overrightarrow{b_1^*}},g^{\nu\overrightarrow{b_2^*}},g^{\beta\overrightarrow{b_3^*}},g^{\beta\overrightarrow{b_4^*}},g^{\overrightarrow{b_5^*}},g^{\overrightarrow{b_6^*}},U_1,U_2,\mu_3)$ along with T_1,T_2 from the subspace assumption, $\mathcal{B}$ tries to decide if T_1,T_2 are distributed as $(V_1,V_2)=(g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}},g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}})$ or as (W_1,W_2) which is in the form of $(g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}+\tau_3\overrightarrow{b_5^*}},g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}+\tau_3\overrightarrow{b_6^*}})$. In this proof, the terms μ_3 are not used.

$\mathcal{B}$ first chooses a random matrix $A\in\mathbb{Z}_p^{2\times2}$ (with all but negligible probability, A is invertible). Now $\mathcal{B}$ implicitly sets $\mathbb{D}=\mathbb{B}_A,\mathbb{D}^*=\mathbb{B}_A^*$, where A is applied as a change of basis matrix to $\overrightarrow{b_5},\overrightarrow{b_6}$ and the transpose of A^{-1} is applied as a change of basis matrix to $\overrightarrow{b_5^*},\overrightarrow{b_6^*}$, as described in [24][Section 3.1]. It implies that $\overrightarrow{d_i}=\overrightarrow{b_i}$ and $\overrightarrow{d_i^*}=\overrightarrow{b_i^*}$ in the public key. The distribution of $\mathbb{D},\mathbb{D}^*$ after the change of basis matrix A is correct and reveal no information about A as shown in [24][Lemma 3].

$\mathcal{B}$ chooses random $\alpha,\gamma\in\mathbb{Z}_p$ and implicitly sets $\theta=\nu,\delta=\beta$ by setting

$$\begin{aligned}\mathsf{pk}&=(p,\mathbb{G},\mathbb{G}_T,\hat{e},g^{\overrightarrow{d_1}},\ldots,g^{\overrightarrow{d_4}},g^{\alpha\overrightarrow{d_1}},g^{\alpha\overrightarrow{d_3}},\hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\overrightarrow{d_1^*}}=\hat{e}_6(g^{\overrightarrow{b_1}},g^{\nu\overrightarrow{b_1^*}})^\gamma),\\\mathsf{sk}&=(g^{\nu\overrightarrow{d_1^*}},g^{\nu\overrightarrow{d_2^*}},g^{\beta\overrightarrow{d_3^*}},g^{\beta\overrightarrow{d_4^*}},\alpha,\gamma).\end{aligned}$$

For the oracle queries to $\mathtt{Sign}_A$, $\mathcal{B}$ computes the signature using sk.

For the oracle queries to $\mathtt{Sign}_B$, $\mathcal{B}$ computes the signature using sk and $g^{\overrightarrow{b_5^*}},g^{\overrightarrow{b_6^*}}$. It can take random combinations of $g^{\overrightarrow{b_5^*}},g^{\overrightarrow{b_6^*}}$ to create random combinations of $g^{\overrightarrow{d_5^*}},g^{\overrightarrow{d_6^*}}$ in the exponent, since the span of $\overrightarrow{d_5^*}$ and $\overrightarrow{d_6^*}$ is equal to the span of $\overrightarrow{b_5^*}$ and $\overrightarrow{b_6^*}$.

Finally, $\mathcal{A}$ will query $\mathcal{B}$ on some challenge message m. $\mathcal{B}$ calculates the signature:

$$\overrightarrow{\sigma}=(g^{\nu\overrightarrow{b_1^*}})^{\frac{\gamma}{\alpha-m}}(T_0)^{\frac{-1}{\alpha-m}}T_1.$$

If T_1,T_2 are distributed as $(V_1,V_2)=(g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}},g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}})$, it is a signature from $\mathtt{Sign}_A$ (by considering $r_1=\tau_1,r_2=\tau_2$). If T_1,T_2 are distributed as $(W_1,W_2)=(g^{\tau_1\nu\overrightarrow{b_1^*}+\tau_2\beta\overrightarrow{b_3^*}+\tau_3\overrightarrow{b_5^*}},g^{\tau_1\nu\overrightarrow{b_2^*}+\tau_2\beta\overrightarrow{b_4^*}+\tau_3\overrightarrow{b_6^*}})$, it is a signature from $\mathtt{Sign}_B$, with exponent vector include $\frac{-\tau_3}{\alpha-m}\overrightarrow{b_5^*}+\tau_3\overrightarrow{b_6^*}$. Using the change of basis matrix A, the coefficients in the vector form is:

$$\tau_3A^t(\frac{-1}{\alpha-m},1)^t,$$

where t is the transpose. It is randomly distributed from the view of $\mathcal{A}$.

Once A returns the forgery, (σ^*,m^*), $\mathcal{B}$ must first check that $\mathcal{A}$ has not seen a signature for m^* before and that (σ^*,m^*) verifies. If either of these checks fail then $\mathcal{B}$ will guess randomly. If both of these are true, then $\mathcal{B}$ must determine what forgery class (σ^*,m^*) belongs to. To distinguish between the forgery types, $\mathcal{B}$ must use a backdoor verification test similar to the one used in the previous proof. $\mathcal{B}$ sets:

$$C_1^*=\hat{e}_6(U_1,g^{\nu\overrightarrow{b_1^*}})^\gamma,\quad\overrightarrow{C_2^*}=U_2(U_1)^{(\alpha-m^*)}.$$

Using the change of basis matrix A, the coefficients of $\overrightarrow{C_2^*}$ in the span of $\overrightarrow{d_5}, \overrightarrow{d_6}$ is:

$$\mu_3 A^{-1}(\alpha - m^*, 1)^t.$$

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_1^* = \hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}).$$

If this equality is false, then $\mathcal{B}$ will output 1. If the equality is true, then $\mathcal{B}$ will flip a coin $b \in \{0, 1\}$ and return b.

If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test since it passes the real verification test. If $\mathcal{A}$ returns a Type II forgery, suppose $\overrightarrow{\sigma^*}$ includes

$$\delta_5 \overrightarrow{d_5^*} + \delta_6 \overrightarrow{d_6^*},$$

as its component in the span of $\overrightarrow{d_5^*}, \overrightarrow{d_6^*}$, for some $\delta_5, \delta_6 \in \mathbb{Z}_p$ (either δ_5 or δ_6 is not equal to zero modulus p). Then the backdoor verification equation proceeds as follows,

$$\hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}) = C_1^* \cdot \hat{e}(g, g)^{\mu_3(\delta_5, \delta_6) \cdot A^{-1}(\alpha - m^*, 1)^t \cdot \overrightarrow{d_5} \cdot \overrightarrow{d_5^*}} \stackrel{?}{=} C_1^*.$$

Thus, if the forgery fails the test, then with probability 1 it is a Type II forgery. If the forgery passes the test then it can be either Type I or Type II. We claim that a Type II forgery can also pass the additional verification test, but only with negligible probability.

For a Type II forgery, we have $(\delta_5, \delta_6) \cdot A^{-1}(\alpha - m^*, 1)^t = 0 \bmod p$, and we have $(\delta_5, \delta_6) \neq (0, 0)$. Since the matrix A is hidden from the view of $\mathcal{A}$ (shown below), the adversary has no better than the negligible probability of achieving the correct value of (δ_5, δ_6).

We now consider the information obtained by the adversary. Suppose the matrix

$$A = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix},$$

for some $a_1, a_2, a_3, a_4 \in \mathbb{Z}_p$. In the challenge signature, the adversary knows

$$\begin{aligned} \tau_3 A^t(\frac{-1}{\alpha - m}, 1)^t &= \tau_3(\frac{-a_1}{\alpha - m} + a_3, \frac{-a_2}{\alpha - m} + a_4)^t \\ &= \frac{-\tau_3}{\alpha - m}(a_1 - (\alpha - m)a_3, a_2 - (\alpha - m)a_4)^t. \end{aligned}$$

In the backdoor verification test, note that

$$\mu_3 A^{-1}(\alpha - m^*, 1)^t = \frac{\mu_3}{\det(A)}(a_4(\alpha - m^*) - a_2, a_1 - a_3(\alpha - m^*))^t.$$

By [24][Lemma 4], the distribution of the above two vectors are negligibly close to uniform distribution over $\mathbb{Z}_p^2$. □

Lemma 6. *If the subspace assumption holds for $k = 1, n = 6$, then our signature scheme is B-II Matching.*

Proof. Given $g^{\overrightarrow{b_1}}, g^{\overrightarrow{b_2}}, g^{\overrightarrow{b_4}}, g^{\overrightarrow{b_5}}, g^{\overrightarrow{b_6}}, g^{\nu \overrightarrow{b_1^*}}, g^{\beta \overrightarrow{b_2^*}}, g^{\overrightarrow{b_3^*}}, g^{\overrightarrow{b_4^*}}, g^{\overrightarrow{b_5^*}}, g^{\overrightarrow{b_6^*}}, U_1 = g^{\mu_1 \overrightarrow{b_1} + \mu_2 \overrightarrow{b_2} + \mu_3 \overrightarrow{b_3}}, \mu_3$, along with T_0 from the subspace assumption, $\mathcal{B}$ tries to decide if T_1 are distributed as $V_1 = g^{\tau_1 \nu \overrightarrow{b_1^*} + \tau_2 \beta \overrightarrow{b_2^*}}$ or as $W_1 = g^{\tau_1 \nu \overrightarrow{b_1^*} + \tau_2 \beta \overrightarrow{b_2^*} + \tau_3 \overrightarrow{b_3^*}}$.

$\mathcal{B}$ implicitly sets

$$\begin{array}{llllll} \overrightarrow{d_1} = \overrightarrow{b_6^*}, & \overrightarrow{d_2} = \overrightarrow{b_3^*}, & \overrightarrow{d_3} = \overrightarrow{b_5^*}, & \overrightarrow{d_4} = \overrightarrow{b_4^*}, & \overrightarrow{d_5} = \overrightarrow{b_2^*}, & \overrightarrow{d_6} = \overrightarrow{b_1^*}, \\ \overrightarrow{d_1^*} = \overrightarrow{b_6}, & \overrightarrow{d_2^*} = \overrightarrow{b_3}, & \overrightarrow{d_3^*} = \overrightarrow{b_5}, & \overrightarrow{d_4^*} = \overrightarrow{b_4}, & \overrightarrow{d_5^*} = \overrightarrow{b_2}, & \overrightarrow{d_6^*} = \overrightarrow{b_1}, \end{array}$$

$\mathcal{B}$ chooses $\gamma, \alpha, \theta, \delta$ randomly from $\mathbb{Z}_p$ and sets

$$\hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\overrightarrow{d_1^*}} = \hat{e}_6(g^{\overrightarrow{b_6}}, g^{\overrightarrow{b_6^*}})^{\gamma\theta}.$$

$\mathcal{B}$ can produce $\mathsf{pk} = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g^{\overrightarrow{d_1}}, \ldots, g^{\overrightarrow{d_4}}, g^{\alpha\overrightarrow{d_1}}, g^{\alpha\overrightarrow{d_3}}, \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\overrightarrow{d_1^*}})$.

$\mathcal{B}$ can answer the $\mathtt{Sign}_B$ oracle as follows, without knowing $g^{\overrightarrow{d_2}}$. $\mathcal{B}$ randomly picks $r_1', r_2, t_5', t_6' \in \mathbb{Z}_p$ and calculates the signature

$$\overrightarrow{\sigma} = U_1^{r_1'\theta} g^{\frac{(\gamma - r_1'\mu_3)\theta}{\alpha - m}\overrightarrow{d_1^*} - \frac{r_2\delta}{\alpha-m}\overrightarrow{d_3^*} + r_2\delta\overrightarrow{d_4^*} + t_5'\overrightarrow{d_5^*} + t_6'\overrightarrow{d_6^*}}.$$

Note that the coefficient of $\overrightarrow{d_2^*} = \overrightarrow{b_3}$ is $r_1\theta := r_1'\mu_3\theta$. Also the coefficients of $\overrightarrow{d_5^*}$ and $\overrightarrow{d_6^*}$ are uniformly random. Thus this is a properly distributed signature from $\mathtt{Sign}_B$.

After the query phase, A will output some forgery, (σ^*, m^*). First, $\mathcal{B}$ will check that the forgery correctly verifies. If the forgery fails verification, then $\mathcal{B}$ will guess randomly. If the forgery verifies, then $\mathcal{B}$ will use a backdoor verification test similar to the previous proofs. First, $\mathcal{B}$ picks some random s_1, s_2, w in $\mathbb{Z}_p$ and sets:

$$C_1^* = \hat{e}(g,g)^{\gamma\theta\overrightarrow{d_1}\cdot\overrightarrow{d_1^*}s_1}, \quad \overrightarrow{C_2^*} = g^{s_1(\alpha-m^*)\overrightarrow{d_1} + s_1\overrightarrow{d_2} + s_2(\alpha-m^*)\overrightarrow{d_3} + s_2\overrightarrow{d_4}} T_1.$$

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_1^* = \hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}).$$

If this equality is true, then $\mathcal{B}$ will output 1. If the equality is false, then $\mathcal{B}$ will flip a coin $b \in \{0,1\}$ and return b.

If $T_1 = g^{\tau_1\nu\overrightarrow{b_1^*} + \tau_2\beta\overrightarrow{b_2^*} + \tau_3\overrightarrow{b_3^*}}$, we know that it will not pass this verification equation (no matter $\mathcal{A}$ returns Type I or Type II forgery) with all but with some negligible probability, δ''. In this case $\mathcal{B}$ will output 1 with probability 1/2.

Next we consider the case for $T_1 = g^{\tau_1\nu\overrightarrow{b_1^*} + \tau_2\beta\overrightarrow{b_2^*}}$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose $\overrightarrow{\sigma^*}$ includes

$$\delta_5\overrightarrow{d_5^*} + \delta_6\overrightarrow{d_6^*},$$

as its component in the span of $\overrightarrow{d_5^*}, \overrightarrow{d_6^*}$, for some $\delta_5, \delta_6 \in \mathbb{Z}_p$ (either δ_5 or δ_6 is not equal to zero modulus p). Then the backdoor verification equation proceeds as follows,

$$\hat{e}_6(\overrightarrow{C_2^*}, \overrightarrow{\sigma^*}) = C_1^* \cdot \hat{e}(g,g)^{(\delta_5,\delta_6)\cdot(\tau_2\beta, \tau_1\nu)^t \cdot \overrightarrow{d_5}\cdot\overrightarrow{d_5^*}} \stackrel{?}{=} C_1^*.$$

If $(\delta_5, \delta_6) \cdot (\tau_2\beta, \tau_1\nu)^t \neq 0 \bmod p$, then the test always fails and $\mathcal{B}$ will output 1 with probability 1/2. Else, $\mathcal{B}$ will output 1. Since we have $(\delta_5, \delta_6) \neq (0,0)$, there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test.

Thus, we can calculate the advantage of $\mathcal{B}$ against the Assumption 1 challenger

$$\begin{aligned}
&\Big|\Pr[\mathcal{B}(D, W_1) = 1] - \Pr[\mathcal{B}(D, V_1) = 1]\Big| \\
=&\Big|\Pr[\mathcal{B}(D, W_1) = 1] - \big(\delta'' \cdot \Pr[\mathcal{B}(D, V_1) = 1] + (1-\delta'')\cdot\frac{1}{2}\big)\Big| \\
=&\Big|(1-\delta'')\big(\Pr[\mathcal{B}(D, W_1) = 1] - \frac{1}{2}\big)\Big| \\
=&(1-\delta'')\big(\epsilon + (1-\epsilon)(\delta' + (1-\delta')\cdot\frac{1}{2}) - \frac{1}{2}\big) \\
=&(1-\delta'')(\frac{\epsilon}{2} + \frac{\delta'}{2} - \frac{\epsilon\delta'}{2}).
\end{aligned}$$

Thus, if ϵ is non-negligible, then $\mathcal{B}$ has non-negligible advantage against the subspace assumption challenger. □

Recall that the decision linear assumption implies the subspace assumption [24]. Combining the above three lemmata and Theorem 1, we have:

Theorem 4. *Our signature scheme is existentially unforgeable under an adaptive chosen message attack if decision linear assumption hold.*

F-unforgeability. We now show the F-unforgeability of our signature scheme.

Theorem 5. *Let $F(x) = (Y^x, Z^x)$, where Y, Z are in the span of $\overrightarrow{d_1}, \overrightarrow{d_3}, \overrightarrow{d_5}$ in the exponent. Our signature scheme is F-unforgeable under an adaptive chosen message attack if decision linear assumption hold.*

Proof. The proof is similar to the proof of standard unforgeability. The major change is that the adversary will return $(\overrightarrow{\sigma^*}, F(m^*))$ instead of $(\overrightarrow{\sigma^*}, m^*)$. Therefore, we have to consider the cases that m^* was used to construct the backdoor verification test.

1. For A-I Matching, $\mathcal{B}$ picks a random $z', y_1, y_3, y_5 \in \mathbb{Z}_p$ and sets
$$Y = g^{y_1 \nu \overrightarrow{b_1^*} + y_3 \beta \overrightarrow{b_3^*} + y_5 \overrightarrow{b_5^*}}, \quad Z = T_1 Y^{z'}.$$
Note that Y and Z are correctly distributed since $z', y_1, y_3, y_5 \in_R \mathbb{Z}_p$ and the span of $\overrightarrow{b_1^*}, \overrightarrow{b_3^*}, \overrightarrow{b_5^*}$ is the same as the span of $\overrightarrow{d_1^*}, \overrightarrow{d_3^*}, \overrightarrow{d_5^*}$. When the adversary returns (Y^{m^*}, Z^{m^*}), then $\mathcal{B}$ can calculate
$$T_1^{m^*} = Z^{m^*} (Y^{m^*})^{-z'}.$$
$T_1^{m^*}$ is the only part of the proof that involve the computation of m^*.
2. For dual-oracle invariance, $\mathcal{B}$ picks a random $y_1, y_3, y_5 \in \mathbb{Z}_p$ and sets
$$Y = g^{y_1 \overrightarrow{b_1} + y_3 \overrightarrow{b_3}} U_1^{y_5}, \quad Z = U_1.$$
Note that Y and Z are correctly distributed since $y_1, y_3, y_5 \in_R \mathbb{Z}_p$ and the span of $\overrightarrow{b_1^*}, \overrightarrow{b_3^*}, \overrightarrow{b_5^*}$ is the same as the span of $\overrightarrow{d_1^*}, \overrightarrow{d_3^*}, \overrightarrow{d_5^*}$. When the adversary returns (Y^{m^*}, Z^{m^*}), then $\mathcal{B}$ can calculate $Z^{m^*} = U_1^{m^*}$ which the only part of the proof that involve the computation of m^*.
3. For B-II Matching, $\mathcal{B}$ picks a random $y_1, y_3, y_5, z' \in \mathbb{Z}_p$ and sets
$$Y = g^{y_1 \overrightarrow{b_6^*} + y_3 \overrightarrow{b_5^*} + y_5 \overrightarrow{b_2^*}} = g^{y_1 \overrightarrow{d_1} + y_3 \overrightarrow{d_3} + y_5 \overrightarrow{d_5}}, \quad Z = g^{s_1 \overrightarrow{b_6^*} + s_2 \overrightarrow{b_5^*}} Y^{z'} = g^{s_1 \overrightarrow{d_1} + s_2 \overrightarrow{d_3}} Y^{z'}.$$
Note that Y and Z are correctly distributed since $z', y_1, y_3, y_5 \in_R \mathbb{Z}_p$. When the adversary returns (Y^{m^*}, Z^{m^*}), then $\mathcal{B}$ can calculate
$$g^{(s_1 \overrightarrow{d_1} + s_2 \overrightarrow{d_3}) m^*} = Z^{m^*} (Y^{m^*})^{-z'},$$
which is the only part of the proof that involves the computation of m^*.

The rest of the proof follows the standard unforgeability proof. □

C Security Proof for Our P-Signature

Signer Privacy. We have to construct the algorithm $\mathsf{SimIssue}(\mathsf{param}, C, \overrightarrow{\sigma})$ and to simulate the adversary's view. Firstly, SimIssue will invoke the simulator of the two-party computation protocol, and extract the input of the adversary: in this case (ρ, m, Open). SimIssue checks if $C = \mathsf{GS.Commit}(\mathsf{param}_{\mathsf{GS}}, Z^m, \mathsf{Open})$; if it isn't, it terminates. Otherwise, it sends the adversary $\overrightarrow{\sigma'} = \overrightarrow{\sigma}^{\rho}$. Suppose the adversary can determine that it is interacting with a simulator. Then it breaks the security of the two-party computation.

User Privacy. We have to construct the algorithm $\mathsf{SimObtain}(\mathsf{param}, \mathsf{pk}, C)$ and to simulate the adversary's view. Firstly, SimObtain will invoke the simulator of the two-party computation protocol, and extract the input of the adversary: in this case (sk', r_1, r_2) (not necessarily the valid secret key used). The simulator picks a random $\rho \in \mathbb{Z}_p$ and calculates $\overrightarrow{\sigma'}$ using (sk', r_1, r_2) and m. It proceeds to interact with the adversary such that if the adversary completes the protocol, its output is $(\overrightarrow{\sigma'}, C)$. If the adversary can determine that it is interacting with a simulator, it breaks the security of the two-party computation.

Zero-knowledge/Witness Indistinguishability. By the security of the Groth-Sahai proof system, the simulator can run a setup simulation for $\mathsf{param}'_{\mathsf{GS}}$. The distribution of $\mathsf{param}'_{\mathsf{GS}}$ is computationally indistinguishable from the real $\mathsf{param}_{\mathsf{GS}}$. Using $\mathsf{param}'_{\mathsf{GS}}$, commitments are perfectly hiding. The simulator can compute the output (comm, π) using a simulation algorithm SimProve on $\mathsf{param}'_{\mathsf{GS}}$. It is witness indistinguishable for the real proof of the `Prove` algorithm with input $(m, \overrightarrow{\sigma})$. Similarly, the output of the `EqCommProve` can also be simulated by SimProve by the composable zero-knowledge property.

Unforgeability. Suppose an adversary $\mathcal{A}$ can break the unforgeability of our P-signature, the we construct an algorithm $\mathcal{B}$ to break the F-unforgeability of the underlying FBB signature, where $F(x) = (Y^x, Z^x)$. Firstly, $\mathcal{B}$ obtains pk' from the challenger of the FBB signature. By the security of the Groth-Sahai proof system, the $\mathcal{B}$ can run a setup simulation for $\mathsf{param}'_{\mathsf{GS}}$ and obtains an extraction trapdoor td. The distribution of $\mathsf{param}'_{\mathsf{GS}}$ is identical to the real $\mathsf{param}_{\mathsf{GS}}$. $\mathcal{B}$ gives param and pk to $\mathcal{A}$ using pk' and $\mathsf{param}'_{\mathsf{GS}}$. For all signing oracle query on message m, $\mathcal{B}$ forwards the query to its challenger to answer it.

Finally, if the adversary can output a proof π that `VerifyPf` accepts, then the simulator can use td to extract $(Z^m, Y^m, \overrightarrow{\sigma})$ from the commitments comm. If $\mathcal{A}$ wins the security game by:

1. $\overrightarrow{\sigma}$ is not a valid signature on m, or comm is not a commitment to m, it breaks the soundness of the Groth-Sahai proof system.
2. $\mathcal{A}$ has never queried the signing oracle on m, then $\mathcal{B}$ returns $(\overrightarrow{\sigma}, F(m) = (Y^m, Z^m))$ as the forgery to the FBB signature.

Therefore if $\mathcal{A}$ can break the unforgeability of our P-signature, we can solve the decision linear problem. □

A non-interactive anonymous credential scheme can be constructed based on any P-signature [6]. Therefore, we obtain a non-interactive anonymous credential scheme under the standard decision linear assumption.

D Security Proof for Signature on a Block of Messages

We denote our original signing algorithm as $\mathtt{Sign}_A$. The dual form of the signature is $\mathtt{Sign}_B(\mathsf{sk}, (m_1, \ldots, m_\ell))$: The signer randomly picks $r, e \in \mathbb{Z}_N$, $X_{2,3}, X'_{2,3} \in \mathbb{G}_{p_3}$ and computes the signature $\sigma = (\sigma_1, \sigma_2, e)$, where:

$$\sigma_1 = (h_1(u_0 u_1^{m_1} \cdots u_\ell^{m_\ell})^{-r})^{\frac{1}{\alpha - e}} X_{2,3}, \quad \sigma_2 = g_1^r X'_{2,3}.$$

Forgery Classes of Our Scheme. We will divide the forgery types based on whether they have a $\mathbb{G}_{p_2}$ component. We let $z \in \mathbb{Z}_N$ denote the exponent represented by the tuple (0 mod p_1;1 mod p_2; 0 mod p_3). Then we can define the forgery classes as follows:

- **Type I**. $\mathcal{V}_I = \{(m^*, \sigma^*) \in \mathcal{V} | (\sigma_2^*)^z = 1 \text{ and } (\sigma_1^*)^z = 1\}$.
- **Type II**. $\mathcal{V}_{II} = \{(m^*, \sigma^*) \in \mathcal{V} | (\sigma_2^*)^z \neq 1 \text{ or } (\sigma_1^*)^z \neq 1\}$.

We will show that our new signature scheme is secure under these assumptions by showing that it satisfies the three properties of a secure dual form signature scheme.

Lemma 7. *If Assumption 1 holds, then our signature scheme is A-I Matching.*

Proof. Suppose that there exists an adversary $\mathcal{A}$ that can create a forgery that is not of Type I with probability ϵ given access to an oracle for the $\texttt{Sign}_A$ algorithm. Then we can create an algorithm $\mathcal{B}$ that breaks Assumption 1 with advantage negligibly close to ϵ.

Given (g, X_3, T) from Assumption 1, $\mathcal{B}$ chooses random $b_0, \ldots, b_\ell \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_i = g^{b_i}$ for $i \in [0, \ell]$, $g_3 = X_3$. $\mathcal{B}$ generates the rest of pk and the secret key $\mathsf{sk} = (h_1, \alpha, g_3)$ according to $\texttt{Setup}$. Note that $\texttt{Sign}_A$ does not need to use $g_{2,3}$.

For the oracle queries to $\texttt{Sign}_A$, $\mathcal{B}$ calculates the signature using sk and answers the query.

In the challenge phase, $\mathcal{A}$ returns $(\sigma^* = (\sigma_1^*, \sigma_2^*, e^*), m^* = (m_1^*, \ldots, m_\ell^*))$ to $\mathcal{B}$. First, $\mathcal{B}$ will check that the forgery verifies, if not then $\mathcal{B}$ will output $b \in \{0, 1\}$ uniformly at random. If the forgery verifies, then $\mathcal{B}$ will try to use this forgery to determine whether T is in $\mathbb{G}_{p_1}$ or $\mathbb{G}_{p_1 p_2}$.

$\mathcal{B}$ sets:

$$C_0^* = \hat{e}(T, h_1), \quad C_1^* = T^\alpha, \quad C_2^* = T, \quad C_3^* = T^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}.$$

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_0^* = \hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*).$$

If this equality is false, then $\mathcal{B}$ will output 1. If the equality is true, then $\mathcal{B}$ will flip a coin $b \in \{0, 1\}$ and return b. Notice that if $\mathcal{B}$ tried to create its own signatures using T, if they were verifiable then they would always pass this additional verification test. This means that $\mathcal{B}$ cannot gain any advantage against the Assumption 1 challenger without using the output of $\mathcal{A}$.

Since it is guaranteed that (σ^*, m^*) passes the real verification test, we know that it will pass this verification equation with if $T = g^s$ (no matter $\mathcal{A}$ returns Type I or Type II forgery). Next we consider the case for $T = g^s Y_2$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is not equal to zero modulo p_2. Then the backdoor verification equation proceeds as follows,

$$\begin{aligned}
\hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*) &= C_0^* \cdot \hat{e}(\hat{g}_2^{\delta_1}, Y_2^{\alpha - e^*}) \cdot \hat{e}(\hat{g}_2^{\delta_2}, Y_2^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}) \\
&= C_0^* \cdot \hat{e}(\hat{g}_2, Y_2)^{\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*)} \stackrel{?}{=} C_0^*.
\end{aligned}$$

If $\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*) \neq 0$, then the test always fails and $\mathcal{B}$ will output 1. Else, $\mathcal{B}$ will output 1 with probability 1/2. However for an adversary to create a Type II forgery, it must find some δ_1, δ_2 such that $\delta_1(\alpha - e^*) = -\delta_2(b_0 - \sum_{i=1}^{\ell} b_i m_i^*)$. It has two cases: (1) $\delta_2 = 0 \bmod p_2$ and $\delta_1 \neq 0 \bmod p_2$. It implies finding $e^* = \alpha \bmod p_2$. However, α modulo p_2 is not revealed at any point during the query phase. (2) $\delta_2 \neq 0 \bmod p_2$. It implies finding $\delta = \delta_1/\delta_2$ such that $\delta(\alpha - e^*) = -(b_0 - \sum_{i=1}^{\ell} b_i m_i^*)$. However, $b_0, \ldots b_\ell$ modulo p_2 are not revealed at any point during the query phase. In both cases, there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test.

Thus, we can calculate the advantage of $\mathcal{B}$ against the Assumption 1 challenger

$$\begin{aligned}
\left| \Pr[\mathcal{B}(D, T_1) = 1] - \Pr[\mathcal{B}(D, T_0) = 1] \right| &= \left| \epsilon\big(\delta' \cdot \frac{1}{2} + (1 - \delta')1\big) + (1 - \epsilon)\frac{1}{2} - \frac{1}{2} \right| \\
&\geq \frac{1}{2}\epsilon - \frac{1}{2}\epsilon\delta'.
\end{aligned}$$

Thus, if ϵ is non-negligible, then $\mathcal{B}$ has non-negligible advantage against the Assumption 1 challenger. □

Lemma 8. *If Assumption 2 holds, then our signature scheme satisfies dual-oracle invariance.*

Proof. Given $(g, X_1 X_2, X_3, Y_2 Y_3, T)$, $\mathcal{B}$ chooses random $b_0, \ldots b_\ell, \alpha \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_i = g^{b_i}$ for $i \in [0, \ell]$, $g_3 = X_3$, and $g_{2,3} = Y_2 Y_3$. $\mathcal{B}$ generates the rest of pk and the secret key $\mathsf{sk} = (h_1, \alpha, g_3, g_{2,3})$ according to $\texttt{Setup}$.

For the oracle queries to $\texttt{Sign}_A$, $\mathcal{B}$ randomly picks $r, e, w, v \in \mathbb{Z}_N$ and computes the signature $\sigma = (\sigma_1, \sigma_2, e)$, where:

$$\sigma_1 = (h_1(u_0 u_1^{m_1} \cdots u_\ell^{m_\ell})^{-r})^{\frac{1}{\alpha - e}} X_3^w, \quad \sigma_2 = g_1^r X_3^v.$$

For the oracle queries to $\texttt{Sign}_B$, $\mathcal{B}$ randomly picks $r, e, w, v \in \mathbb{Z}_N$ and computes the signature $\sigma = (\sigma_1, \sigma_2, e)$, where:

$$\sigma_1 = (h_1(u_0 u_1^{m_1} \cdots u_\ell^{m_\ell})^{-r})^{\frac{1}{\alpha - e}} (Y_2 Y_3)^w, \quad \sigma_2 = g_1^r (Y_2 Y_3)^v.$$

Note that the value of v and w modulo p_2 and modulo p_3 are uncorrelated by the Chinese remainder theorem.

Finally, $\mathcal{A}$ will query $\mathcal{B}$ on some challenge message, $m_1, \ldots m_\ell$. $\mathcal{B}$ will choose some random $e^*, w, v \in \mathbb{Z}_N$ (with $\alpha \neq e^*$), and calculates the signature:

$$\sigma_1^* = h_1^{\frac{1}{\alpha - e^*}} \cdot T^{\frac{b_0 + \sum_{i=1}^{\ell} b_i m_i}{\alpha - e^*}} \cdot X_3^w, \quad K_2 = T \cdot X_3^v.$$

If $T = Z_1 Z_3$, it is a signature from $\texttt{Sign}_A$ (by considering $g^r = Z_1$). If $T = Z_1 Z_2 Z_3$, it is a signature from $\texttt{Sign}_B$, since $b_0, \ldots b_\ell$ modulo p_2 are not revealed at any point during the query phase. Hence the $\mathbb{G}_{p_2}$ part of σ_1^* is randomly distributed from the view of $\mathcal{A}$.

Once A returns the forgery, (σ^*, m^*), $\mathcal{B}$ must first check that $\mathcal{A}$ has not seen a signature for m^* before and that (σ^*, m^*) verifies. If either of these checks fail then$\mathcal{B}$ will guess randomly. If both of these are true, then $\mathcal{B}$ must determine what forgery class (σ^*, m^*) belongs to in order to determine what subgroup T is in. To distinguish between the forgery types, $\mathcal{B}$ must use a backdoor verification test similar to the one used in the previous proof. $\mathcal{B}$ sets:

$$C_0^* = \hat{e}(X_1 X_2, h_1), \quad C_1^* = (X_1 X_2)^\alpha, \quad C_2^* = X_1 X_2, \quad C_3^* = (X_1 X_2)^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}.$$

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_0^* = \hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*).$$

If this equality is false, then $\mathcal{B}$ will output 1. If the equality is true, then $\mathcal{B}$ will flip a coin $b \in \{0, 1\}$ and return b.

If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test since it passes the real verification test. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is not equal to zero modulo p_2. Then the backdoor verification equation proceeds as follows,

$$\begin{aligned} \hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*) &= C_0^* \cdot \hat{e}(\hat{g}_2^{\delta_1}, X_2^{\alpha - e^*}) \cdot \hat{e}(\hat{g}_2^{\delta_2}, X_2^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}) \\ &= C_0^* \cdot \hat{e}(\hat{g}_2, X_2)^{\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*)} \stackrel{?}{=} C_0^*. \end{aligned}$$

Thus, if the forgery fails the test, then with probability 1 it is a Type II forgery. If the forgery passes the test then it can be either Type I or Type II. We claim that a Type II forgery can also pass the additional verification test, but only with negligible probability.

For a Type II forgery, we have $\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*) = 0 \bmod p_2$. We have the following cases:

1. If $\delta_2 = 0 \bmod p_2$ and $\delta_1 \neq 0 \bmod p_2$, it implies $e^* = \alpha \bmod p_2$. Since $e^* \neq \alpha$ by $\mathcal{B}$ setting, it has negligible probability that they are equal modulo p_2.
2. If $\delta_2 \neq 0 \bmod p_2$, we rewrite the equation as $\delta(\alpha - e^*) + (b_0 + \sum_{i=1}^{\ell} b_i m_i^*) = 0 \bmod p_2$, where $\delta = \delta_1 / \delta_2$. In order to create a Type II forgery, an adversary must implicitly determine $(b_0 + \sum_{i=1}^{\ell} b_i m_i^*)/(\alpha - e^*)$ modulo p_2. The adversary only knows $(b_0 + \sum_{i=1}^{\ell} b_i m_i^*)/(\alpha - e)$ modulo p_2 from the challenge signature if $T = Z_1 Z_2 Z_3$. As long as $e \neq e^*$ modulo p_2, the adversary has no better than the negligible probability of achieving the correct value of δ modulo p_2.

We now consider the information obtained by the adversary. In the challenge signature, α, $b_0, \ldots b_\ell$ modulo p_2 are only included in the first element of the challenge signature. Thus the attacker can only derive the single value $\frac{b_0+\sum_{i=1}^{\ell} b_i m_i^*}{\alpha - e^*}$ modulo p_2. However, this single equation has unknowns α and $b_0, \ldots, b_\ell$ modulo p_2 and it is not possible to determine their unique values. Moreover, $\frac{b_0+\sum_{i=1}^{\ell} b_i m_i}{\alpha - e}$ is a pairwise independent function of $(m_1, \ldots, m_\ell)$ modulo p_2. Therefore, the attacker cannot achieve the correct value of $\frac{b_0+\sum_{i=1}^{\ell} b_i m_i^*}{\alpha - e}$ mod p_2 as long as $m_j \neq m_j^*$ mod p_2 for some $j \in [1, \ell]$, except with negligible probability. It is possible that m_i is equal to m_i^* modulo p_2 for all i, but m_j is not equal to m_j^* modulo N for some j. If this occurs with non-negligible probability, then $\mathcal{B}$ can extract a non-trivial factor of N by computing the greatest common divisor of N and $m_j - m_j^*$. This non-trivial factor can be used to break Assumption 2 with non-negligible advantage. Hence, if a forgery passes the additional verification test, then with high probability it is a Type I forgery. □

Lemma 9. *If Assumption 3 holds, then our signature scheme is B-II Matching.*

Proof. Suppose that there exists an adversary, $\mathcal{A}$, that can create a Type I forgery with non-negligible probability ϵ given access to an oracle for the $\texttt{Sign}_B$ algorithm. Then we can create an algorithm $\mathcal{B}$ that breaks Assumption 3 with non-negligible advantage.

Given $(g, X_3, g^a X_2, g^s Y_2, Z_2, T)$, $\mathcal{B}$ chooses random $b_0, \ldots b_\ell \in \mathbb{Z}_N$ and sets

$$g_1 = g, \quad u_i = g^{b_i} \quad \text{for } i \in [0, \ell], \quad \hat{e}(g_1, h_1) = \hat{e}(g, g^a X_2).$$

$\mathcal{B}$ implicitly sets $h_1 = g^a$. $\mathcal{B}$ sends the public key $\texttt{pk}$ to $\mathcal{A}$.

$\mathcal{B}$ can answer the $\texttt{Sign}_B$ oracle as follows. $\mathcal{B}$ randomly picks $r, e \in \mathbb{Z}_N$, $R_2, R_2' \in \mathbb{G}_{p_3}$ and $R_3, R_3' \in \mathbb{G}_{p_3}$ and calculates:

$$\sigma_1 = (g^a X_2 \cdot (u_0 u_1^{m_1} \cdots u_\ell^{m_\ell})^{-r})^{\frac{1}{\alpha - e}} \cdot R_2 \cdot R_3, \quad \sigma_2 = g^r \cdot R_2' \cdot R_3',$$

Therefore $\mathcal{B}$ can answer (σ_1, σ_2, e).

After the query phase, A will output some forgery, (σ^*, m^*). First, $\mathcal{B}$ will check that the forgery correctly verifies. If the forgery fails verification, then $\mathcal{B}$ will guess randomly. If the forgery verifies, then $\mathcal{B}$ can use this forgery to determine whether $T = \hat{e}(g, g)^{as}$. $\mathcal{B}$ will use a backdoor verification test similar to the previous proofs. First, B sets:

$$C_0^* = T, \quad C_1^* = (g^s Y_2)^\alpha, \quad C_2^* = g^s Y_2, \quad C_3^* = (g^s Y_2)^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}.$$

Since α and b_i are chosen randomly modulo N, there will be no correlation between the $\mathbb{G}_{p_1}$ and the $\mathbb{G}_{p_2}$ components of C_1^* and C_3^*. Finally, $\mathcal{B}$ will check that

Then $\mathcal{B}$ proceeds with a *backdoor verification* test as follows,

$$C_0^* = \hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*).$$

If this equality is true, then $\mathcal{B}$ will output 1. If the equality is false, then $\mathcal{B}$ will flip a coin $b \in \{0, 1\}$ and return b.

If T is a random group element in $\mathbb{G}_T$, we know that it will not pass this verification equation (no matter $\mathcal{A}$ returns Type I or Type II forgery) with all but with some negligible probability, δ''. In this case $\mathcal{B}$ will output 1 with probability 1/2.

Next we consider the case for $T = \hat{e}(g, g)^{as}$. If $\mathcal{A}$ returns a Type I forgery, it also passes the backdoor verification test. If $\mathcal{A}$ returns a Type II forgery, suppose the $\mathbb{G}_{p_2}$ part of σ_1^* and σ_2^* are $\hat{g}_2^{\delta_1}$ and $\hat{g}_2^{\delta_2}$ respectively, for some $\hat{g}_2 \in \mathbb{G}_{p_2}$, $\delta_1, \delta_2 \in \mathbb{Z}_N$ and either δ_1 or δ_2 is not equal to zero modulo p_2. Then the backdoor verification equation proceeds as follows,

$$\begin{aligned} \hat{e}(\sigma_1^*, C_1^* (C_2^*)^{-e^*}) \cdot \hat{e}(C_3^*, \sigma_2^*) &= C_0^* \cdot \hat{e}(\hat{g}_2^{\delta_1}, Y_2^{\alpha - e^*}) \cdot \hat{e}(\hat{g}_2^{\delta_2}, Y_2^{b_0 + \sum_{i=1}^{\ell} b_i m_i^*}) \\ &= C_0^* \cdot \hat{e}(\hat{g}_2, Y_2)^{\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*)} \stackrel{?}{=} C_0^*. \end{aligned}$$

If $\delta_1(\alpha - e^*) + \delta_2(b_0 + \sum_{i=1}^{\ell} b_i m_i^*) \neq 0$, then the test always fails and $\mathcal{B}$ will output 1 with probability 1/2. Else, $\mathcal{B}$ will output 1. However for an adversary to create a Type II forgery, it must find some δ_1, δ_2 such that $\delta_1(\alpha - e^*) = -\delta_2(b_0 - \sum_{i=1}^{\ell} b_i m_i^*)$. We have two possible cases: (1) $\delta_2 = 0 \bmod p_2$ and $\delta_1 \neq 0 \bmod p_2$. It implies finding $e^* = \alpha \bmod p_2$. However, α modulo p_2 is not revealed at any point during the query phase. (2) $\delta_2 \neq 0 \bmod p_2$. It implies finding $\delta = \delta_1/\delta_2$ such that $\delta(\alpha - e^*) = -(b_0 - \sum_{i=1}^{\ell} b_i m_i^*)$. However, $b_0, \ldots b_\ell$ modulo p_2 are not revealed at any point during the query phase. In both cases, there is a negligible chance, δ', of an attacker being able to create a Type II forgery that passes the backdoor verification test.

Thus, we can calculate the advantage of $\mathcal{B}$ against the Assumption 3 challenger

$$
\begin{aligned}
&\Big|\Pr[\mathcal{B}(D, T_0) = 1] - \Pr[\mathcal{B}(D, T_1) = 1]\Big| \\
=&\Big|\Pr[\mathcal{B}(D, T_0) = 1] - \big(\delta'' \cdot \Pr[\mathcal{B}(D, T_0) = 1] + (1 - \delta'') \cdot \frac{1}{2}\big)\Big| \\
=&\Big|(1 - \delta'')\big(\Pr[\mathcal{B}(D, T_0) = 1] - \frac{1}{2}\big)\Big| \\
=&(1 - \delta'')\big(\epsilon + (1 - \epsilon)(\delta' + (1 - \delta') \cdot \frac{1}{2}) - \frac{1}{2}\big) \\
=&(1 - \delta'')(\frac{\epsilon}{2} + \frac{\delta'}{2} - \frac{\epsilon\delta'}{2}).
\end{aligned}
$$

Thus, if ϵ is non-negligible, then $\mathcal{B}$ has non-negligible advantage against the Assumption 3 challenger. □

Combining the above three lemmata and Theorem 1, we have:

Theorem 6. *Our signature scheme is existentially unforgeable under an adaptive chosen message attack.*

E Review for Identity-Based Encryption

Pairing-based IBE that are fully secure in the standard model can be classified into two main families [12]: *commutative blinding* and *exponent inversion*. Roughly speaking, commutative blinding is to create blinding factors from two secret coefficients in a way that makes them "commute" (i.e. not depend on the application order) using pairings. An example is Waters IBE [30]. When compared with Boneh-Boyen's first IBE scheme [8], Waters IBE replaces its hash for identities with one relying on long parameters, to achieve adaptive security.

For exponent inversion IBE, the recipient's ID is embedded in the exponent via a secret function f, yet $g^{f(\mathsf{ID})}$ is publicly computable. Consider a ciphertext having $(g^{f(\mathsf{ID})})^s$, the inversion is done by pairing it with a private key of the form $\hat{g}^{1/f(\mathsf{ID})}$ to get a session key $\hat{e}(g, \hat{g})^s$. Sakai-Kasahara IBE [29] proposed an exponent inversion IBE (SK-IBE) secure in the random oracle model, under the q-SDH assumption [15]. The second IBE scheme of Boneh-Boyen [8] (BB_2-IBE) was selectively secure in the standard model under a q-type assumption.

E.1 Syntax

An IBE scheme consists of four probabilistic polynomial-time (PPT) algorithms:

1. `Setup`: On input a security parameter 1^λ, it generates a master public key mpk and a master secret key msk.
2. `Extract`: On input msk and an identity ID from an identity space $\mathcal{I}$, it outputs an identity-based secret key $\mathsf{sk}_{\mathsf{ID}}$.
3. `Enc`: On input mpk, ID and a message M from a message space $\mathcal{M}$, it outputs a ciphertext C.

4. $\mathtt{Dec}$: On input $\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}$ and C, it outputs a message m or $\perp$ symbolizing the failure of decryption.

Correctness. For all $M \in \mathcal{M}$ and $\mathsf{ID} \in \mathcal{I}$, $M = \mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M))$, where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^\lambda)$ and $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$.

E.2 Security Model for Confidentiality

We consider the following indistinguishability based game against adaptive chosen identity and chosen plaintext attacks (IND-ID-CPA) for confidentiality [11].

1. Setup. The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^k)$ and gives mpk to the adversary $\mathcal{A}$.
2. Query 1. The following oracles can be queried by $\mathcal{A}$:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID})$: On input an identity $\mathsf{ID} \in \mathcal{I}$, it returns the identity-based secret key $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$.
3. Challenge. $\mathcal{A}$ sends two messages $M_0^*, M_1^* \in \mathcal{M}$ and an identity $\mathsf{ID}^* \in \mathcal{I}$ to the challenger. The challenger picks a random bit b' and computes $C^* \leftarrow \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}^*, M_{b'}^*)$. The challenger sends C^* to $\mathcal{A}$.
4. Query 2. $\mathcal{A}$ is allowed to query the above oracles adaptively.
5. Output. $\mathcal{A}$ returns a guess b^* of b'.

$\mathcal{A}$ wins the game if $b' = b^*$. We require that there was no query with $\mathcal{KEO}(\mathsf{ID}^*)$. The advantage of $\mathcal{A}$ is the probability of winning the game minus $1/2$. An IBE scheme is IND-ID-CPA secure if there is no PPT $\mathcal{A}$ with non-negligible advantage in the game above.

F Security Proof of Our IBE Construction

Under the dual system encryption paradigm [31], we define the following semi-functional (SF) structures which are used in the security proofs only. These SF structures are just like their normal version in the actual scheme, but "perturbed" by a $\mathbb{G}_{p_2}$ generator, denoted by either $\bar{g}_2$ or $\hat{g}_2$ below.

An *SF secret key* (or just *SF key*) is in the form of

$$K_1' = K_1 \cdot \bar{g}_2^{\gamma}, \quad K_2' = K_2 \cdot \bar{g}_2,$$

where $\gamma \in \mathbb{Z}_N$, and (K_1, K_2) is a normal secret key.

An *SF ciphertext* is in the form of

$$(C_0' = C_0, \quad C_1' = C_1 \cdot \hat{g}_2, \quad C_2' = C_2 \cdot \hat{g}_2^{\delta}),$$

where $\delta \in \mathbb{Z}_N$ and (C_0, C_1, C_2) is a normal ciphertext.

Regarding decryption involving SF structures, it will succeed if an SF key is used to decrypt a normal ciphertext, or a normal key is used to decrypt an SF ciphertext. However, decrypting an SF ciphertext using an SF secret key will result in a message "blinded" by

$$\hat{e}(\bar{g}_2, \hat{g}_2)^{\gamma+\delta}.$$

In case that the exponents in these extra blinding factors are zeros, decryption still works and this leads us to the notion of *nominally semi-functional (NSF) secret keys*. An NSF secret key is a special kind of SF key which can be used to decrypt SF ciphertext, that means $\gamma + \delta = 0$. If an SF secret key is not nominally semi-functional, then it is *truly semi-functional*.

Theorem 7. *Our IBE scheme is IND-ID-CPA secure under Assumptions 1, 2 and 3.*

Proof. We prove by a hybrid argument using a sequence of games. The first game Game_{real} is the IND-ID-CPA game. Let the challenge identity be ID^*.

The second game Game_{res} is the same as Game_{real} except that the adversary cannot ask for the secret key of identity as $\mathsf{ID} = \mathsf{ID}^* \bmod p_2$. This restriction will be retained throughout the subsequent games. After that, we denote q as the number of extract oracle queries. For $k = 0$ to q, we define Game_k as:

Game$_k$: It is the same as Game_{real}, except that the challenge ciphertext is semi-functional and the keys used to answer first k^{th} oracle queries are semi-functional. The keys for the rest of the queries are normal.

As a result, all keys are normal and the challenge ciphertext is semi-functional in Game_0. In Game_q, all keys and the challenge ciphertext are semi-functional.

The last game is Game_{final}, which is the same as Game_q except that the challenge ciphertext is a semi-functional encryption of a random message, instead of one of the two challenge messages.

We will prove the indistinguishability between these games.

Lemma 10. *We can construct an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 1 or Assumption 2 if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_{real}$) - $Adv_{\mathcal{A}}$($Game_{res}$) is non-negligible*

The proofs of lemma 10 is easy and is omitted.

Lemma 11. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 1, if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_{res}$) - $Adv_{\mathcal{A}}$($Game_0$) = ϵ.*

Proof. Given (g, X_3, T) from Assumption 1, $\mathcal{B}$ can simulate Game_{res} or Game_0 with $\mathcal{A}$. $\mathcal{B}$ chooses random $\beta \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_1 = g^\beta$, $g_3 = X_3$. $\mathcal{B}$ generates the rest of mpk and $\mathsf{msk} = (h_1, \alpha, g_3)$ according to $\mathtt{Setup}$.

For the extract oracle queries, $\mathcal{B}$ answers by calculating the $\mathsf{sk}_{\mathsf{ID}}$ using msk.

When $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0^*, M_1^*, and an identity ID^* to $\mathcal{B}$, $\mathcal{B}$ randomly picks a bit $b' \in \{0, 1\}$ and $s \in \mathbb{Z}_N$. $\mathcal{B}$ calculates the challenge ciphertext as:

$$C_0^* = M_{b'}^* \cdot \hat{e}(T, h_1), \quad C_1^* = T^{\alpha - \mathsf{ID}^*}, \quad C_2^* = T^\beta.$$

If $T = g^s$, this is a normal ciphertext and hence $\mathcal{B}$ simulates Game_{res}. If $T = g^s Y_2$, this is an SF ciphertext with $\hat{g}_2 = Y_2^{\alpha - \mathsf{ID}^*}, \hat{g}_2^\delta = Y_2^\beta$; and hence $\mathcal{B}$ simulates Game_0 with $\delta = \beta/(\alpha - \mathsf{ID}^*)$. By the Chinese remainder theorem, the values of $\alpha, \beta \bmod p_2$ are not correlated with the values of α and β modulo p_1. If $\mathcal{A}$ can distinguish between Game_{res} and Game_0, $\mathcal{B}$ can then break Assumption 1. □

Lemma 12. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 2, if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_{\ell-1}$) - $Adv_{\mathcal{A}}$($Game_\ell$) = ϵ.*

Proof. Given $(g, X_1X_2, X_3, Y_2Y_3, T)$ from Assumption 2, $\mathcal{B}$ can simulate $\text{Game}_{\ell-1}$ or Game_ℓ with $\mathcal{A}$. $\mathcal{B}$ picks random $\beta, \alpha \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $g_1 = g$, $u_1 = g^\beta$ and $g_3 = X_3$. $\mathcal{B}$ generates the rest of mpk and $\mathsf{msk} = (h_1, \alpha, g_3)$ as $\mathtt{Setup}$.

For the k^{th} distinct extraction oracle query on ID_k:

- if $k < \ell$, $\mathcal{B}$ calculates the normal key $\mathsf{sk}_{\mathsf{ID}_k}$ using msk.
- if $k > \ell$, $\mathcal{B}$ calculates the normal key $\mathsf{sk}_{\mathsf{ID}_k} = (K_1, K_2)$ using msk. $\mathcal{B}$ randomly picks $\gamma_1, \gamma_2 \in \mathbb{Z}_N$ and calculates the SF key:

$$K_1' = K_1 \cdot (Y_2Y_3)^{\gamma_1}, \quad K_2' = K_2 \cdot (Y_2Y_3)^{\gamma_2}.$$

This is a semi-functional key. By the Chinese remainder theorem, the values of γ_1, γ_2 modulo p_2 and those modulo p_3 are not correlated.

– if $k = \ell$, $\mathcal{B}$ chooses random $X_3', X_3'' \in \mathbb{G}_{p_3}$ and calculates the key $\mathsf{sk}_{\mathsf{ID}_\ell}$:

$$K_1 = h_1^{\frac{1}{\alpha - \mathsf{ID}_\ell}} \cdot T^{\frac{\beta}{\alpha - \mathsf{ID}_\ell}} \cdot X_3', \quad K_2 = T \cdot X_3''.$$

If $T = Z_1 Z_3$, it is a normal key (by considering $g^r = Z_1$). Hence $\mathcal{B}$ simulates $\mathrm{Game}_{\ell-1}$. If $T = Z_1 Z_2 Z_3$, it is an SF key with $\bar{g}_2^\gamma = Z_2^{\frac{\beta}{\alpha - \mathsf{ID}_\ell}}$ and $\bar{g}_2 = Z_2$. Hence $\mathcal{B}$ simulates Game_ℓ. Again, note that the value of $\gamma \bmod p_2$ is not correlated with the values of α and β modulo p_1.

$\mathcal{B}$ returns the above $\mathsf{sk}_{\mathsf{ID}}$ as the response of the query.

In the challenge phase, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0^*, M_1^*, and an identity ID^*. $\mathcal{B}$ chooses a random bit $b' \in \{0, 1\}$ and calculates the challenge ciphertext:

$$C_0^* = M_{b'}^* \cdot \hat{e}(X_1 X_2, h_1), \quad C_1^* = (X_1 X_2)^{\alpha - \mathsf{ID}^*}, \quad C_2^* = (X_1 X_2)^\beta.$$

It is an SF ciphertext with $\hat{g}_2 = X_2^{\alpha - \mathsf{ID}^*}$ and $\hat{g}_2^\delta = X_2^\beta$. Note that $f(\mathsf{ID}) = \beta/(\alpha - \mathsf{ID})$ is a pairwise independent function modulo p_2. Therefore as long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell$, δ and $\gamma = \beta/(\alpha - \mathsf{ID}_\ell) \bmod p_2$ will seem randomly distributed to $\mathcal{A}$ (again, we note that the values of α and β modulo p_2 are uncorrelated with their values modulo p_1 by the Chinese remainder theorem). If $\mathsf{ID}^* = \mathsf{ID}_\ell \bmod p_2$, $\mathcal{A}$ has made an invalid key request. This is where we use our additional restriction.

Hence, $\mathcal{B}$ can break Assumption 2 if $\mathcal{A}$ can distinguish $\mathrm{Game}_{\ell-1}$ and Game_ℓ. □

Lemma 13. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 3, if there exists $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_q$) - $Adv_{\mathcal{A}}(Game_{final}) = \epsilon$.*

Proof. Given $(g, g^a X_2, g^s Y_2, Z_2, X_3, T)$ from Assumption 3, $\mathcal{B}$ chooses random $\beta, \alpha \in \mathbb{Z}_N$ and sets

$$g_1 = g, \quad u_1 = g^\beta, \quad \hat{e}(g_1, h_1) = \hat{e}(g, g^a X_2).$$

$\mathcal{B}$ implicitly sets $h_1 = g^a$. $\mathcal{B}$ sends the master public key mpk to $\mathcal{A}$.

$\mathcal{B}$ can calculate the semi-functional secret key for ID as follows. $\mathcal{B}$ randomly picks $r \in \mathbb{Z}_N$, $R_2, R_2' \in \mathbb{G}_{p_3}$and $R_3, R_3' \in \mathbb{G}_{p_3}$ and calculates:

$$K_1' = (g^a X_2 \cdot u_1^{-r})^{\frac{1}{\alpha - \mathsf{ID}}} \cdot R_2 \cdot R_3, \quad K_2' = g^r \cdot R_2' \cdot R_3',$$

Therefore $\mathcal{B}$ can answer all extraction oracle queries.

In the challenge phase, $\mathcal{B}$ randomly chooses $b' \in \{0, 1\}$ and calculates the SF ciphertext:

$$C_0' = M_{b'}^* \cdot T, \quad C_1' = (g^s Y_2)^{\alpha - \mathsf{ID}^*}, \quad C_2' = (g^s Y_2)^\beta.$$

If $T = \hat{e}(g, g)^{as}$, then $\mathcal{B}$ simulates Game_q. Otherwise, then $\mathcal{B}$ simulates Game_{final}. If $\mathcal{A}$ can distinguish between these two games, $\mathcal{B}$ can break Assumption 3. Therefore no PPT adversary $\mathcal{A}$ can distinguish between Game_q and Game_{final}. □

Back to the proof of our theorem. Finally in Game_{final}, the value of b' is information-theoretically hidden from $\mathcal{A}$. Hence $\mathcal{A}$ has no advantage in winning Game_{final}. If Assumptions 1, 2 and 3 hold, then Game_{real} is indistinguishable from Game_{final}. Hence the attacker has negligible advantage in winning Game_{real}. Therefore, our scheme is IND-ID-CPA secure. □

F.1 Extension to CCA Security

We can extend our IBE scheme to achieve CCA security, by extending the scheme to 2-level HIBE and applying strong one-time signatures [14]. We sketch the changes as follows:

$\mathsf{Setup}(1^\lambda)$: The PKG additionally picks generators w_1, v_1 of the subgroups $\mathbb{G}_{p_1}$ as part of the master public key. Denote $(\mathsf{KGen}, \mathsf{Sign}, \mathsf{Verify})$ as a strong one-time signature scheme.

$\mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$: The PKG randomly picks $r, t \in \mathbb{Z}_N$, $X_3, X_3', X_3'', X_3''' \in \mathbb{G}_{p_3}$ and computes:

$$K_1 = (h_1 u_1^{-r})^{\frac{1}{\alpha - \mathsf{ID}}} w_1^t X_3, \quad K_2 = g_1^r X_3', \quad K_3 = g_1^{t(\alpha - \mathsf{ID})} X_3'', \quad K_4 = v_1^t X_3'''.$$

$\mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M)$: To encrypt a message M for ID, the sender randomly picks $s \in \mathbb{Z}_N$, runs $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$ and outputs $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ where

$$C_0 = M \cdot \hat{e}(g_1, h_1)^s, \quad C_1 = g_1^{s(\alpha - \mathsf{ID})}, \quad C_2 = u_1^s, \quad C_3 = (v_1^{\mathsf{vk}} w_1)^s,$$
$$\sigma = \mathsf{Sign}(\mathsf{sk}, C_0 || C_1 || C_2 || C_3).$$

$\mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathfrak{C})$: Given a ciphertext $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ and a secret key $\mathsf{sk}_{\mathsf{ID}} = (K_1, K_2, K_3, K_4)$, it first checks if $\mathsf{Verify}(\mathsf{vk}, C_0 || C_1 || C_2 || C_3, \sigma) = 1$. If not, it outputs $\perp$. Otherwise, the recipient calculates:

$$M = C_0 \cdot \hat{e}(C_3, K_3) / \hat{e}(C_1, K_1 K_4^{\mathsf{vk}}) \cdot \hat{e}(C_2, K_2).$$

Theorem 8. *The above IBE scheme is IND-ID-CCA secure under Assumptions 1, 2 and 3 and the unforgeability of the strong one-time signature scheme.*

The proof is similar to the CPA security. We only sketch the differences below. The semi-functional $\mathbb{G}_{p_2}$ part of the key appears in K_1, K_2, K_3, K_4. The semi-functional $\mathbb{G}_{p_2}$ part of the ciphertext appears in C_1, C_2, C_3. The decryption oracle is simulated by either using the normal or semi-functional key to decrypt normal ciphertext. The semi-functional challenge ciphertext cannot be asked to the decryption oracle, and modifying the challenge ciphertext will not give a valid ciphertext (by the unforgeability of the strong one-time signature scheme). The proof is analogous to the CCA security of the accountable authority IBE.

F.2 Extension to Prime Order Group

Our IBE construction is modified to one in prime order group as follows.

$\mathtt{Setup}(1^\lambda)$: The PKG runs the bilinear group generator $\mathcal{G}(1^\lambda)$ to get $(p, \mathbb{G}, \mathbb{G}_T, \hat{e})$ where $\mathbb{G}$ has prime order p and g is a generator of $\mathbb{G}$. It samples random dual orthonormal bases $(\mathbb{D}, \mathbb{D}^*) \leftarrow Dual(\mathbb{Z}_p^6)$ as defined in [24]. We let $\overrightarrow{d_1}, \ldots, \overrightarrow{d_6}$ denote the elements of $\mathbb{D}$ and $\overrightarrow{d_1^*}, \ldots, \overrightarrow{d_6^*}$ denote the elements of $\mathbb{D}^*$. The PKG randomly picks $\alpha, \theta, \delta \in \mathbb{Z}_p$. The master public key is

$$(p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g^{\overrightarrow{d_1}}, \ldots, g^{\overrightarrow{d_4}}, g^{\alpha \overrightarrow{d_1}}, g^{\alpha \overrightarrow{d_3}}, \hat{e}(g, g)^{\beta \theta \overrightarrow{d_1} \cdot \overrightarrow{d_1^*}}).$$

The message space $\mathcal{M}$ is $\mathbb{G}_T$ and the identity space $\mathcal{I}$ is $\mathbb{Z}_p$. The master secret key is

$$(g^{\theta \overrightarrow{d_1^*}}, g^{\beta \theta \overrightarrow{d_1^*}}, g^{\theta \overrightarrow{d_2^*}}, g^{\delta \overrightarrow{d_3^*}}, g^{\delta \overrightarrow{d_4^*}}, \alpha).$$

$\mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$: The PKG randomly picks $r_1, r_2 \in \mathbb{Z}_p$, and computes

$$\overrightarrow{K} = g^{(\frac{\beta - r}{\alpha - \mathsf{ID}}) \theta \overrightarrow{d_1^*} + r \theta \overrightarrow{d_2^*} + (\frac{-r}{\alpha - \mathsf{ID}}) \delta \overrightarrow{d_3^*} + r \delta \overrightarrow{d_4^*}}.$$

$\mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M)$: To encrypt a message M for ID, the sender randomly picks $s_1, s_2 \in \mathbb{Z}_p$ and outputs $\mathfrak{C} = (C_0, C_1)$ where

$$C_0 = M \cdot \left(\hat{e}(g, g)^{\beta \theta \overrightarrow{d_1} \cdot \overrightarrow{d_1^*}}\right)^{s_1}, \quad \overrightarrow{C_1} = g_1^{s_1(\alpha - \mathsf{ID}) \overrightarrow{d_1} + s_1 \overrightarrow{d_2} + s_2(\alpha - \mathsf{ID}) \overrightarrow{d_3} + s_2 \overrightarrow{d_4}}.$$

$\mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathfrak{C})$: Given a ciphertext $\mathfrak{C} = (C_0, \overrightarrow{C_1})$ and a secret key $\mathsf{sk}_{\mathsf{ID}} = \overrightarrow{K}$, the recipient calculates:

$$M = C_0 / \hat{e}_6(\overrightarrow{C_1}, \overrightarrow{K}).$$

The proof is similar to the one in [24] and the proof of our signature scheme in prime order groups, under the decisional linear assumption.

G Security of Our Accountable-Authority IBE

We now recall the security model for A-IBE. It consists of five PPT algorithms:

1. $\mathtt{Setup}$: On input a security parameter 1^λ, it generates a system parameter param, a master public key mpk and a master secret key msk. The param is treated as the input of all other algorithms, and it is omitted from the writing for simplicity.
2. $\mathtt{Extract}$: On input msk and an identity ID from an identity space $\mathcal{I}$, it engages in an interactive protocol with the user. At the end, the user receives an identity-based secret key $\mathsf{sk}_{\mathsf{ID}}$. Note that the algorithm may not know the exact key that the user obtains.
3. $\mathtt{Enc}$: On input mpk, ID and a message M from a message space $\mathcal{M}$, it outputs a ciphertext C.
4. $\mathtt{Dec}$: On input $\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}$ and C, it outputs a message M or $\perp$ symbolizing the failure of decryption.
5. $\mathtt{Trace}^{\mathbb{D}}$: On input $\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}$ and a black-box access to an ϵ-useful decoder box $\mathbb{D}$(defined below) for an identity ID, the algorithm will decide if $\mathbb{D}$ was created by the PKG of the user ID.

Definition 1. *For non-negligible ϵ, a PPT algorithm $\mathbb{D}$ is an ϵ-useful decoder box for an identity ID if*

$$\Pr[M \leftarrow \mathcal{M} : \mathbb{D}(\mathtt{Enc}(\mathit{mpk}, \mathsf{ID}, M)) = M] \geq \epsilon.$$

Correctness. For all $M \in \mathcal{M}$ and $\mathsf{ID} \in \mathcal{I}$, $M = \mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}, M))$, where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^\lambda)$ and $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$. For some A-IBE schemes, they only require the decryption succeeds with overwhelming probability (e.g. [28]).

G.1 Security Models

Confidentiality. We consider the following indistinguishability based game against adaptive chosen identity and chosen ciphertext attacks (IND-ID-CCA) for confidentiality.

1. Setup. The challenger runs $(\mathsf{param}, \mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^k)$, withhold msk, and gives $(\mathsf{param}, \mathsf{mpk})$ to the adversary $\mathcal{A}$.
2. Query 1. The following oracles can be queried adaptively by $\mathcal{A}$:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID})$: On input an identity $\mathsf{ID} \in \mathcal{I}$, it returns the identity-based secret key $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$.
   - Decryption Oracle $\mathcal{DO}(\mathsf{ID}, C)$: On input an identity $\mathsf{ID} \in \mathcal{I}$ and a ciphertext C, it returns the decryption result $\mathtt{Dec}(\mathsf{mpk}, \mathtt{Extract}(\mathsf{msk}, \mathsf{ID}), C)$.
3. Challenge. $\mathcal{A}$ sends two messages $M_0^*, M_1^* \in \mathcal{M}$ and an identity $\mathsf{ID}^* \in \mathcal{I}$ to the challenger. The challenger picks a random bit b' and computes $C^* \leftarrow \mathtt{Enc}(\mathsf{mpk}, \mathsf{ID}^*, M_{b'}^*)$. The challenger sends C^* to $\mathcal{A}$.
4. Query 2. $\mathcal{A}$ is allowed to query the above oracles adaptively.
5. Output. $\mathcal{A}$ returns a guess b^* of b'.

$\mathcal{A}$ wins the game if $b' = b^*$. We require that there was no query with $\mathcal{KEO}(\mathsf{ID}^*)$ or $\mathcal{DO}(\mathsf{ID}^*, C^*)$. The advantage of $\mathcal{A}$ is the probability of winning the game minus 1/2. An A-IBE scheme is IND-ID-CCA secure if there is no PPT $\mathcal{A}$ with non-negligible advantage in the game above.

Dishonest User Security. We consider the following ComputeNewKey game against adaptive chosen identity and chosen ciphertext attacks (ComputeNewKey-CCA) for dishonest user security.

1. Setup. The challenger runs $(\mathsf{param}, \mathsf{mpk}, \mathsf{msk}) \leftarrow \mathtt{Setup}(1^k)$, withhold msk, and gives $(\mathsf{param}, \mathsf{mpk})$ to the adversary $\mathcal{A}$.
2. Query. The following oracles can be queried adaptively by $\mathcal{A}$:
   - Extraction Oracle $\mathcal{KEO}(\mathsf{ID})$: On input an identity $\mathsf{ID} \in \mathcal{I}$, it returns the identity-based secret key $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathtt{Extract}(\mathsf{msk}, \mathsf{ID})$.

- Decryption Oracle $\mathcal{DO}(\mathsf{ID}, C)$: On input an identity $\mathsf{ID} \in \mathcal{I}$ and a ciphertext C, it returns the decryption result $\mathtt{Dec}(\mathsf{mpk}, \mathtt{Extract}(\mathsf{msk}, \mathsf{ID}), C)$.

3. Output. $\mathcal{A}$ outputs an ϵ-useful decoder box $\mathbb{D}$ and a key $\mathsf{sk}_{\mathsf{ID}^*}$ for some identity ID^*.

$\mathcal{A}$ wins the game if $\mathtt{Trace}^{\mathbb{D}}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}^*}, \epsilon) = \text{PKG}$. An A-IBE scheme is said to be ComputeNewKey-CCA secure if there is no PPT $\mathcal{A}$ with non-negligible advantage in winning the game above.

We can also require that ID^* was queried once in the Extraction Oracle. Since the A-IBE scheme is required to satisfy the IND-ID-CCA security as well, outputting a key for an identity which has not been queried would contradict the IND-ID-CCA security. Hence, adding this additional requirement does not weaken the security.

Note that the extra decryption oracle given to $\mathcal{A}$ may help $\mathcal{A}$ to create a decoder box $\mathbb{D}$, since $\mathbb{D}$ mimics the function of a decryption oracle. Therefore, the ComputeNewKey-CCA model is stronger than its CPA variant.

Dishonest PKG Security. We consider the following FindNewKey game against adaptive chosen ciphertext attacks (FindNewKey-CCA) for dishonest PKG security.

1. Initialize. The challenger gives param to the adversary $\mathcal{A}$.
2. Setup. $\mathcal{A}$ gives the master public key mpk and an identity $\mathsf{ID}^* \in \mathcal{I}$ to the challenger. The challenger aborts if they are not well-formed.
3. Extract. The challenger and $\mathcal{A}$ engage in the extract protocol for ID^*. If neither party aborts, the challenger receives $\mathsf{sk}_{\mathsf{ID}^*}$ as output.
4. Query. The following oracle can be queried adaptively by $\mathcal{A}$:
   - Decryption Oracle $\mathcal{DO}(C)$: On input a ciphertext C, it returns the decryption result $M/\perp \leftarrow \mathtt{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}^*}, C)$.
5. Output. $\mathcal{A}$ outputs an ϵ-useful decoder box $\mathbb{D}$.

$\mathcal{A}$ wins the game if $\mathtt{Trace}^{\mathbb{D}}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}^*}, \epsilon) = \text{User}$. An A-IBE scheme is said to be FindNewKey-CCA secure if there is no PPT $\mathcal{A}$ with non-negligible advantage in winning the game above.

G.2 Assumptions

We need the modified Assumption 1', 2' and 3' for the composite order group of 4 primes. We use an extra Assumption 4' for the Dishonest User security of the A-IBE scheme. These assumptions and their generic security can be found in [16]. (Our Assumption 4' is slightly weaker than the one in [16] by giving less parameters in the problem.) We also use an extra Assumption 5' for the Dishonest PKG security of the A-IBE scheme. It is very similar to the Assumption 3'.

Assumption 1'. Given a group generator $\mathcal{G}$, we define the following distribution:

$$(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda), \quad g, X_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_3 \xleftarrow{R} \mathbb{G}_{p_3}, \quad g_4 \xleftarrow{R} \mathbb{G}_{p_4},$$
$$T_0 \xleftarrow{R} \mathbb{G}_{p_1p_2}, \quad T_1 \xleftarrow{R} \mathbb{G}_{p_1}. \quad D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_3, g_4).$$

Assume that for any PPT algorithm $\mathcal{A}_1$ with output in $\{0, 1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_1} := |\Pr[(D, T_0) = 1] - \Pr[(D, T_1) = 1]| = neg(\lambda).$$

Assumption 2'. Given a group generator $\mathcal{G}$, we define the following distribution:

$$(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda),$$
$$g, X_1, Z_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_i, Y_i, Z_i \xleftarrow{R} \mathbb{G}_{p_i} (i = 2, 3), \quad g_4 \xleftarrow{R} \mathbb{G}_{p_4},$$
$$T_0 = Z_1Z_3, \quad T_1 = Z_1Z_2Z_3. \quad D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1X_2, X_3, Y_2Y_3, g_4).$$

Assume that for any PPT algorithm $\mathcal{A}_2$ with output in $\{0,1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_2} := |\Pr[(D,T_0)=1] - \Pr[(D,T_1)=1]| = neg(\lambda).$$

Assumption 3'. Given a group generator $\mathcal{G}$, we define the following distribution:

$$\begin{aligned}
&(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda),\\
&\alpha, s \xleftarrow{R} \mathbb{Z}_N, \quad g \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_2, Y_2, Z_2 \xleftarrow{R} \mathbb{G}_{p_2}, \quad X_3 \xleftarrow{R} \mathbb{G}_{p_3}, \quad g_4 \xleftarrow{R} \mathbb{G}_{p_4},\\
&T_0 = \hat{e}(g,g)^{\alpha s}, \quad T_1 \xleftarrow{R} \mathbb{G}_T. \quad D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^\alpha X_2, g^s Y_2, Z_2, X_3, g_4).
\end{aligned}$$

Assume that for any PPT algorithm $\mathcal{A}_3$ with output in $\{0,1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_3} := |\Pr[(D,T_0)=1] - \Pr[(D,T_1)=1]| = neg(\lambda).$$

Assumption 4'. Given a group generator $\mathcal{G}$, we define the following distribution:

$$\begin{aligned}
&(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda),\\
&r, s \xleftarrow{R} \mathbb{Z}_N, \quad g_1, u_1, y_1 \xleftarrow{R} \mathbb{G}_{p_1}, \quad g_2, Y_2, Z_2, W_2, V_2 \xleftarrow{R} \mathbb{G}_{p_2},\\
&g_3 \xleftarrow{R} \mathbb{G}_{p_3}, \quad g_4, y_4, W_4, V_4 \xleftarrow{R} \mathbb{G}_{p_4}, \quad T_0 = y_1^s V_2 V_4, \quad T_1 \xleftarrow{R} \mathbb{G}_{p_1p_2p_4}.\\
&D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, g_2, g_3, g_4, y_1y_4, y_1^r Y_2, g_1^r Z_2, g_1^s W_2 W_4).
\end{aligned}$$

Assume that for any PPT algorithm $\mathcal{A}_4$ with output in $\{0,1\}$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_4} := |\Pr[(D,T_0)=1] - \Pr[(D,T_1)=1]| = neg(\lambda).$$

Assumption 5'. Given a group generator $\mathcal{G}$, we define the following distribution:

$$\begin{aligned}
&(N = p_1p_2p_3p_4, \mathbb{G}, \mathbb{G}_T, \hat{e}) \xleftarrow{R} \mathcal{G}(1^\lambda),\\
&r, s \xleftarrow{R} \mathbb{Z}_N, \quad g_1, y_1, \xleftarrow{R} \mathbb{G}_{p_1}, \quad X_2, Y_2 \xleftarrow{R} \mathbb{G}_{p_2}, \quad g_3 \xleftarrow{R} \mathbb{G}_{p_3}, \quad g_4 \xleftarrow{R} \mathbb{G}_{p_4}.\\
&D := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, y_1, g_3, g_4, y_1^r, g_1^r X_2, y_1^s Y_2).
\end{aligned}$$

Assume that for any PPT algorithm $\mathcal{A}_5(D)$ with output T in $\mathbb{G}_T$, the advantage

$$Adv_{\mathcal{G},\mathcal{A}_5} := \Pr[T = \hat{e}(y_1,g_1)^{rs}] = neg(\lambda).$$

G.3 Security Proof

Under the dual system encryption paradigm [31], we define the following semi-functional (SF) structures which are used in the security proofs only. These SF structures just like their normal version in the actual scheme, but "perturbed" by a $\mathbb{G}_{p_2}$ generator, denoted by either $\bar{g}_2$ or $\hat{g}_2$ below.

An *SF secret key* (or just *SF key*) is in the form of

$$K_1' = K_1 \cdot \bar{g}_2^{\gamma_1}, \quad K_2' = K_2 \cdot \bar{g}_2, \quad K_3' = K_3 \cdot \bar{g}_2^{\gamma_2}, \quad K_4' = K_4 \cdot \bar{g}_2^{\gamma_3}.$$

where $\gamma_1, \gamma_2, \gamma_3 \in \mathbb{Z}_N$, and (K_1, K_2, K_3, K_4) is a normal secret key.

An *SF ciphertext* is in the form of

$$(C_0' = C_0, \quad C_1' = C_1 \cdot \hat{g}_2, \quad C_2' = C_2 \cdot \hat{g}_2^{\delta_1}, \quad C_3' = C_3 \cdot \hat{g}_2^{\delta_2}, \quad \sigma', \quad \mathsf{vk}),$$

where $\delta_1, \delta_2 \in \mathbb{Z}_N$ and $(C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ is a normal ciphertext. Note that $\sigma' = \mathsf{Sign}(\mathsf{sk}, C_0||C_1||C_2||C_3)$ can only be computed with the knowledge of sk.

Regarding decryption involving SF structures, it will succeed if an SF key is used to decrypt a normal ciphertext, or a normal key is used to decrypt an SF ciphertext. However, decrypting an SF ciphertext using an SF secret key will result in a message "blinded" by

$$\hat{e}(\bar{g}_2, \hat{g}_2)^{\gamma_1 + \mathsf{vk} \cdot \gamma_3 + \delta_1 - \delta_2 \gamma_2}.$$

In case that the exponents in these extra blinding factors are zeros, decryption still works and this leads us to the notion of *nominally semi-functional (NSF) secret keys*. An NSF secret key is a special kind of SF key which can be used to decrypt SF ciphertext, that means $\gamma_1 + \mathsf{vk} \cdot \gamma_3 + \delta_1 - \delta_2 \gamma_2 = 0$. If an SF secret key is not nominally semi-functional, then it is *truly semi-functional*.

Confidentiality. The IND-ID-CCA security is similar to the IND-ID-CCA security of our dual system Gentry-IBE. The definition of semi-functional keys and ciphertext are analogous to that in our dual system Gentry-IBE, except the addition of group $\mathbb{G}_{p_4}$ elements in the ciphertext.

Theorem 9. *Our A-IBE is IND-ID-CCA secure if Assumptions 1', 2' and 3' hold.*

Proof. The proof is very similar to the IND-ID-CPA game of our dual system Gentry-IBE, except how the interactive extraction oracle is simulated. We also prove by a hybrid argument using a sequence of games. The games are the same Game_{real}, Game_{res}, $\text{Game}_1, \ldots, \text{Game}_q$ and Game_{final}, except that (1) q is the number of distinct ID asked to the key extraction oracle and the decryption oracle, (2) for Game_{res}, the adversary cannot ask for the key of $\mathsf{ID} = \mathsf{ID}^* \bmod p_2$, and cannot ask for the decryption of ciphertext with the one-time verification key part $\mathsf{vk} = \mathsf{vk}^* \bmod p_2$. We will prove the indistinguishability between these games.

Lemma 14. *We can construct an algorithm $\mathcal{B}$ with non-negligible advantage in breaking Assumption 1' or Assumption 2', if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_{real}$) - $Adv_{\mathcal{A}}$($Game_{res}$) is non-negligible.*

The proof of lemma 14 is easy and is omitted.

Lemma 15. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 1', if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_{res}$) - $Adv_{\mathcal{A}}$($Game_0$) = ϵ.*

Proof. It is the same as the proof of Lemma 11, since $\mathcal{B}$ can use msk to simulate the extract oracle queries directly. Given (g, X_3, g_4, T) from Assumption 1', $\mathcal{B}$ can simulate Game_{res} or Game_0 with $\mathcal{A}$. $\mathcal{B}$ chooses random $\kappa, \nu, \phi, \varphi \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $\beta = \kappa\nu, g_1 = g, u_1 = g^\beta, g_3 = X_3, v_1 = g^\phi, w_1 = g^\varphi, y_1 = g^\kappa$. $\mathcal{B}$ generates the rest of $\mathsf{param}, \mathsf{mpk}$ and the master secret key $\mathsf{msk} = (h_1, \alpha, \nu)$ according to `Setup`.

For the extract oracle and the decryption oracle queries, $\mathcal{B}$ calculates the $\mathsf{sk}_{\mathsf{ID}}$ using msk and answers the query.

In the challenge phase, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0^*, M_1^*, and an identity ID^*. $\mathcal{B}$ randomly picks a bit $b' \in \{0, 1\}$, $X_4, X_4' \in \mathbb{G}_{p_4}$ and $s \in \mathbb{Z}_N$. $\mathcal{B}$ runs $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}(1^\lambda)$ and calculates the challenge ciphertext as:

$$C_0^* = M_{b'}^* \cdot \hat{e}(T, h_1), \quad C_1^* = T^{\alpha - \mathsf{ID}^*} X_4, \quad C_2^* = T^\beta X_4', \quad C_3^* = T^{\phi \cdot \mathsf{vk} + \varphi} X_4'',$$

where $\sigma^* \leftarrow \mathsf{Sign}(\mathsf{sk}^*, C_0^* || C_1^* || C_2^* || C_3^*)$. If $T = g^s$, this is a normal ciphertext and hence $\mathcal{B}$ simulates Game_{res}. If $T = g^s Y_2$, this is an SF ciphertext with $\hat{g}_2 = Y_2^{\alpha - \mathsf{ID}^*}, \hat{g}_2^{\delta_1} = Y_2^\beta, \hat{g}_2^{\delta_2} = Y_2^{\phi \cdot \mathsf{vk} + \varphi}$; and hence $\mathcal{B}$ simulates Game_0 with $\delta_1 = \frac{\beta}{\alpha - \mathsf{ID}^*}, \delta_2 = \frac{\phi \cdot \mathsf{vk} + \varphi}{\alpha - \mathsf{ID}^*}$. Note that the value of $\alpha, \beta, \mu_1, \phi, \varphi \bmod p_2$ is not correlated with the values of $\alpha, \beta, \mu_1, \phi, \varphi$ modulo p_1 by the Chinese remainder theorem.

For further decryption oracle query with $\mathsf{ID} = \mathsf{ID}^*$ and $\mathsf{vk} = \mathsf{vk}^*$, $\mathcal{B}$ returns $\perp$ for invalid ciphertext (by the security of the strong one-time signature).

Therefore if $\mathcal{A}$ can distinguish between Game_{res} and Game_0, then $\mathcal{B}$ can break Assumption 1'. □

Lemma 16. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 2', if there exists an adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ (Game$_{\ell-1}$) - $Adv_{\mathcal{A}}$(Game$_\ell$) $= \epsilon$.*

Proof. Given $(g, X_1X_2, X_3, Y_2Y_3, g_4, T)$, $\mathcal{B}$ can simulate Game$_{\ell-1}$ or Game$_\ell$ with $\mathcal{A}$. $\mathcal{B}$ chooses random $\kappa, \alpha, \phi, \varphi, \nu \in \mathbb{Z}_N$, $h_1 \in \mathbb{G}_{p_1}$. $\mathcal{B}$ sets $\beta = \kappa\nu, g_1 = g, u_1 = g^\beta, v_1 = g^\phi, w_1 = g^\varphi, y_1 = g^\kappa$ and $g_3 = X_3$. $\mathcal{B}$ generates the rest of $\mathsf{param}, \mathsf{mpk}$ and the master secret key $\mathsf{msk} = (h_1, \alpha, \nu)$ according to `Setup`.

For the k^{th} distinct oracle query on ID_k is the extraction oracle:

- If $k < \ell$, $\mathcal{B}$ calculates the normal key using msk.
- if $k > \ell$, $\mathcal{B}$ calculates the normal key $\mathsf{sk}_{\mathsf{ID}_k} = (K'_1, K'_2, K'_3, K'_4)$ using msk. $\mathcal{B}$ randomly picks $\gamma_1, \gamma_2, \gamma_3, \gamma_4 \in \mathbb{Z}_N$ and calculates the SF key:

$$\bar{K}'_1 = K'_1 \cdot (Y_2Y_3)^{\gamma_1},\ \bar{K}'_2 = K'_2 \cdot (Y_2Y_3)^{\gamma_2},\ \bar{K}'_3 = K'_3 \cdot (Y_2Y_3)^{\gamma_3},\ \bar{K}'_4 = K'_4 \cdot (Y_2Y_3)^{\gamma_4}.$$

  This is a semi-functional key and $\gamma_1, \ldots, \gamma_4$ modulo p_2 and modulo p_3 are not correlated by the Chinese remainder theorem.
- if $k = \ell$, $\mathcal{B}$ computes (A_1, A_2) using ν. The adversary sends the commitment R_1 to $\mathcal{B}$. $\mathcal{B}$ chooses random $t'_0 \in \mathbb{Z}_N$, $X'_3, X''_3, X'''_3, X''''_3 \in \mathbb{G}_{p_3}$ and calculates the key $\mathsf{sk}_{\mathsf{ID}_\ell}$:

$$K'_1 = (h_1 \cdot R_1)^{\frac{1}{\alpha-\mathsf{ID}_\ell}} \cdot T^{\frac{-\beta}{\alpha-\mathsf{ID}_\ell}} \cdot T^{t'_0\varphi} \cdot X'_3, \quad K'_2 = T \cdot X''_3,$$
$$K'_3 = T^{t'_0(\alpha-\mathsf{ID})} X'''_3, \quad K'_4 = T^{t'_0\phi} X''''_3.$$

  If $T = Z_1Z_3$, it is a normal key (by considering $g_1^r = Z_1, g_1^{t_0} = Z_1^{t'_0}$). Hence $\mathcal{B}$ simulates Game$_{\ell-1}$. If $T = Z_1Z_2Z_3$, it is an SF key with $\bar{g}_2 = Z_2$, $\bar{g}_2^{\gamma_1} = Z_2^{\frac{-\beta}{\alpha-\mathsf{ID}_\ell}+t'_0\varphi}$, $\bar{g}_2^{\gamma_2} = Z_2^{t'_0(\alpha-\mathsf{ID}_\ell)}$ and $\bar{g}_2^{\gamma_3} = Z_2^{t'_0\phi}$. Hence $\mathcal{B}$ simulates Game$_\ell$. Again, note that the value of $\gamma_1, \gamma_2, \gamma_3 \bmod p_2$ are not correlated with the values of $\alpha, \beta, \phi, \varphi, t'_0$ modulo p_1.

$\mathcal{B}$ returns the above $\mathsf{sk}_{\mathsf{ID}}$ as the response of the query.

For the decryption oracle query, $\mathcal{B}$ calculates the identity-based secret key as the key extraction first (and picks R by itself) and then outputs the decryption result.

In the challenge phase, $\mathcal{A}$ sends $\mathcal{B}$ two messages M_0^*, M_1^*, and an identity ID^*. $\mathcal{B}$ chooses a random bit $b' \in \{0,1\}$, $X_4, X'_4, X''_4 \in \mathbb{G}_{p_4}$ and runs $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}(1^\lambda)$. $\mathcal{B}$ calculates the challenge ciphertext $\mathfrak{C} = (C_0^*, C_1^*, C_2^*, C_3^*, \sigma^*, \mathsf{vk}^*)$, where:

$$C_0^* = M_{b'}^* \cdot \hat{e}(X_1X_2, h_1), \quad C_1^* = (X_1X_2)^{\alpha-\mathsf{ID}^*} X_4,$$
$$C_2^* = (X_1X_2)^\beta X'_4 \quad C_3^* = (X_1X_2)^{\phi\cdot\mathsf{vk}^*+\varphi} X''_4,$$

and $\sigma^* \leftarrow \mathsf{Sign}(\mathsf{sk}^*, C_0^*||C_1^*||C_2^*||C_3^*)$. It is an SF ciphertext with $\hat{g}_2 = X_2^{\alpha-\mathsf{ID}^*}$, $\hat{g}_2^{\delta_1} = X_2^\beta$ and $\hat{g}_2^{\delta_2} = X_2^{\phi\cdot\mathsf{vk}+\varphi}$.

For further decryption oracle query with $\mathsf{vk} = \mathsf{vk}^*$, $\mathcal{B}$ returns $\perp$ for invalid ciphertext (by the security of the strong one-time signature).

Finally consider the view of adversary. If the ℓ^{th} oracle query is the extraction oracle query for ID_ℓ, $\mathcal{A}$ sees $\delta_1 = \frac{\beta}{\alpha-\mathsf{ID}^*}$ and $\delta_2 = \frac{\phi\cdot\mathsf{vk}^*+\varphi}{\alpha-\mathsf{ID}^*}$ from the challenge ciphertext, and $\gamma_1 = \frac{-\beta}{\alpha-\mathsf{ID}_\ell} + t'_0\varphi \bmod p_2$, $\gamma_2 = t'_0(\alpha - \mathsf{ID}_\ell)$ and $\gamma_3 = t'_0\phi$ from the ℓ^{th} key extraction oracle query.

Lemma 17. *The values $\delta_1, \delta_2, \gamma_1, \gamma_2$ and γ_3 modulo p_2 are randomly distributed from the view of $\mathcal{A}$.*

Proof. As we state, we want to show the five exponents on $\mathbb{G}_{p_2}$ are 5-pairs independent in adversary's view. Then we have

$$\delta_1 = \frac{\beta}{\alpha-\mathsf{ID}^*},\ \delta_2 = \frac{\mathsf{vk}^*\phi+\varphi}{\alpha-\mathsf{ID}^*},\ \gamma_1 = \frac{-\beta}{\alpha-\mathsf{ID}_\ell} + t'_0\varphi,\ \gamma_2 = t'_0(\alpha-\mathsf{ID}_\ell), \quad \gamma_3 = t'_0\phi,$$

and the value $\alpha, \beta, t'_0, \phi, \varphi$ modulo p_2 is unknown by the adversary, but the value $\mathsf{vk}^*, \mathsf{ID}_\ell, \mathsf{ID}^*$ is known by $\mathcal{A}$. Following the definition of 5-pairs independent, we need to prove that the distribution of any one value is uniform, even if the other four values are fixed. There are five cases we need to show (the possibility that more than one of these fixed value is equal to 0 modulo p_2 is negligible, so we can just consider the case that all of these four values are non-zero):

1. $(\delta_1, \delta_2, \gamma_1, \gamma_2) \to \gamma_3$. When $\delta_1, \delta_2, \gamma_1, \gamma_2$ is fixed, and α is a variable, we show the relationship between α and γ_3:

$$\begin{aligned}
&\beta = \delta_1(\alpha - \mathsf{ID}^*), \quad t'_0 = \frac{\gamma_2}{\alpha - \mathsf{ID}_\ell}, \quad \varphi = \frac{\gamma_1}{\gamma_2}(\alpha - \mathsf{ID}_\ell) + \frac{\delta_1}{\gamma_2}(\alpha - \mathsf{ID}^*),\\
&\mathsf{vk}^*\phi = (\delta_2 + \frac{\delta_1}{\gamma_2})(\alpha - \mathsf{ID}^*) - \frac{\gamma_1}{\gamma_2}(\alpha - \mathsf{ID}_\ell),\\
&\gamma_3 = \frac{1}{\mathsf{vk}^*}[(\delta_2\gamma_2 + \delta_1)(1 + \frac{\mathsf{ID}_\ell - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}) - \gamma_1].
\end{aligned}$$

   As long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell \bmod p_2$, we can see that γ_3's distribution is uniform if α is uniformly distributed.
2. $(\delta_1, \delta_2, \gamma_1, \gamma_3) \to \gamma_2$. The value $\delta_1, \delta_2, \gamma_1, \gamma_3$ is fixed, and α is a variable, we show the relationship between α and γ_2:

$$\begin{aligned}
&\beta = \delta_1(\alpha - \mathsf{ID}^*),\\
&\gamma_1 + \mathsf{vk}^*\gamma_3 = -\delta_1(\frac{\alpha - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}) + t'_0(\mathsf{vk}^*\phi + \varphi) = -\delta_1(\frac{\alpha - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}) + t'_0\delta_2(\alpha - \mathsf{ID}^*),\\
&t'_0 = \frac{\gamma_1 + \mathsf{vk}^*\gamma_3}{\delta_2(\alpha - \mathsf{ID}^*)} + \frac{\delta_1}{\delta_2(\alpha - \mathsf{ID}_\ell)},\\
&\gamma_2 = \frac{\gamma_1 + \mathsf{vk}^*\gamma_3}{\delta_2} \cdot (1 + \frac{\mathsf{ID}^* - \mathsf{ID}_\ell}{\alpha - \mathsf{ID}^*}) + \frac{\delta_1}{\delta_2}.
\end{aligned}$$

   As long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell \bmod p_2$, we can see that γ_2's distribution is uniform if α is uniformly distributed.
3. $(\delta_1, \delta_2, \gamma_2, \gamma_3) \to \gamma_1$. The value $\delta_1, \delta_2, \gamma_2, \gamma_3$ is fixed, and α is a variable, we show the relationship between α and r_1:

$$\begin{aligned}
&\beta = \delta_1(\alpha - \mathsf{ID}^*), \quad t'_0 = \frac{\gamma_2}{\alpha - \mathsf{ID}_\ell}, \quad \phi = \frac{\gamma_3}{\gamma_2}(\alpha - \mathsf{ID}_\ell),\\
&\varphi = (\alpha - \mathsf{ID}^*)\delta_2 - \frac{\mathsf{vk}^*\gamma_3}{\gamma_2}(\alpha - \mathsf{ID}_\ell), \quad t'_0\varphi = \gamma_2\delta_2(1 + \frac{\mathsf{ID}_\ell - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}) - \mathsf{vk}^*\gamma_3,\\
&\gamma_1 = (\gamma_2\delta_2 - \delta_1)(1 + \frac{\mathsf{ID}_\ell - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}) - \mathsf{vk}^*\gamma_3.
\end{aligned}$$

   As long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell \bmod p_2$, we can see that γ_1's distribution is uniform if α is uniformly distributed.
4. $(\delta_1, \gamma_1, \gamma_2, \gamma_3) \to \delta_2$. The value $\delta_1, \gamma_1, \gamma_2, \gamma_3$ is fixed, and α is a variable, we show the relationship between α and δ_2:

$$\begin{aligned}
&\beta = \delta_1(\alpha - \mathsf{ID}^*), \quad t'_0 = \frac{\gamma_2}{\alpha - \mathsf{ID}_\ell}, \quad t'_0(\mathsf{vk}^*\phi + \varphi) = \gamma_1 + \mathsf{vk}^*\gamma_3 + \delta_1(\frac{\alpha - \mathsf{ID}^*}{\alpha - \mathsf{ID}_\ell}),\\
&(\mathsf{vk}^*\phi + \varphi) = \frac{\gamma_1 + \mathsf{vk}^*\gamma_3}{\gamma_2}(\alpha - \mathsf{ID}_\ell) + \frac{\delta_1}{\gamma_2}(\alpha - \mathsf{ID}^*),\\
&\delta_2 = \frac{\gamma_1 + \mathsf{vk}^*\gamma_3}{\gamma_2}(1 + \frac{\mathsf{ID}^* - \mathsf{ID}_\ell}{\alpha - \mathsf{ID}^*}) + \frac{\delta_1}{\gamma_2}.
\end{aligned}$$

   As long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell \bmod p_2$, we can see that δ_2's distribution is uniform if α is uniformly distributed.

5. $(\delta_2, \gamma_1, \gamma_2, \gamma_3) \to \delta_1$. The value $\delta_2, \gamma_1, \gamma_2, \gamma_3$ is fixed, and α is a variable, we show the relationship between α and δ_1:

$$t_0' = \frac{\gamma_2}{\alpha - \mathsf{ID}_\ell}, \quad \gamma_1 + \mathsf{vk}^*\gamma_3 = \frac{-\beta}{\alpha - \mathsf{ID}_\ell} + \frac{\gamma_2}{\alpha - \mathsf{ID}_\ell}(\varphi + \mathsf{vk}^*\phi),$$
$$(\gamma_1 + \mathsf{vk}^*\gamma_3)(\alpha - \mathsf{ID}_\ell) = -\beta + \gamma_2\delta_2(\alpha - \mathsf{ID}^*),$$
$$\delta_1 = \frac{-\beta}{\alpha - \mathsf{ID}^*} = (\gamma_1 + \mathsf{vk}^*\gamma_3)(1 + \frac{\mathsf{ID}^* - \mathsf{ID}_\ell}{\alpha - \mathsf{ID}^*}) - \gamma_2\delta_2.$$

As long as $\mathsf{ID}^* \neq \mathsf{ID}_\ell \bmod p_2$, we can see that δ_1's distribution is uniform if α is uniformly distributed.

Again, we note that the values of $\alpha, \beta, t_0', \phi$ and φ modulo p_2 are uncorrelated with their values modulo p_1 by the Chinese remainder theorem.

By the $Game_{res}$, $\mathcal{A}$ cannot ask for extraction query on $\mathsf{ID}_\ell = \mathsf{ID}^* \bmod p_2$. Hence, the values $\delta_1, \delta_2, \gamma_1, \gamma_2$ and γ_3 modulo p_2 are randomly distributed from the view of $\mathcal{A}$. □

Now, we consider the view of $\mathcal{A}$ if the ℓ^{th} query is the decryption oracle on $C' = (C_0', C_1', C_2', C_3', \mathsf{vk}', \sigma')$ for ID_ℓ. If $\mathsf{ID}_\ell \neq \mathsf{ID}^* \bmod p_2$, then the output of the decryption oracle is determined by the identity-based secret key of ID_ℓ in our simulation. It is already analyzed in Lemma 17. Next, we consider for the case of $\mathsf{ID}_\ell = \mathsf{ID}^* \bmod p_2$, but $\mathsf{vk}' \neq \mathsf{vk}^*$ for the decryption oracle query.

Recall that $\mathcal{B}$ uses an SF key to run the decryption algorithm. If C_1', C_2', C_3' has a $\mathbb{G}_{p_2}$ part of $\hat{g}_2, \hat{g}_2^{\bar{\delta}_1}, \hat{g}_2^{\bar{\delta}_2}$ respectively, then the decryption will result in a message "blinded" by $\hat{e}(\bar{g}_2, \hat{g}_2)^{\gamma_1 + \mathsf{vk}' \cdot \gamma_3 + \bar{\delta}_1 - \bar{\delta}_2\gamma_2}$.

By putting the values of γ_1, γ_2 and γ_3 of the SF key into the equation, we have the value of the exponent ξ as:

$$\begin{aligned}\xi &= \frac{\beta}{\alpha - \mathsf{ID}^*} + t_0'\varphi + \mathsf{vk}' \cdot t_0'\phi + \bar{\delta}_1 - \bar{\delta}_2 t_0'(\alpha - \mathsf{ID}^*) \\ &= \frac{\beta}{\alpha - \mathsf{ID}^*} + \bar{\delta}_1 + t_0'(\varphi + \mathsf{vk}' \cdot \phi - \bar{\delta}_2(\alpha - \mathsf{ID}^*)).\end{aligned}$$

Therefore, $\mathcal{A}$ has the view of ξ, δ_1, δ_2 modulo p_2 from $\mathcal{B}$'s answer. Observe that δ_1 and δ_2 do not contain the value of $t_0' \bmod p_2$. Since the value of $t_0' \bmod p_2$ does not appear elsewhere during the simulation, $\xi \bmod p_2$ appears to be randomly distributed if $\varphi + \mathsf{vk}' \cdot \phi - \bar{\delta}_2(\alpha - \mathsf{ID}^*) \neq 0 \bmod p_2$. It implies that $\mathcal{A}$ may notice that ξ is not randomly distributed if $\bar{\delta}_2 = \frac{\varphi + \mathsf{vk}' \cdot \phi}{\alpha - \mathsf{ID}^*}$. However, observe that δ_1 has no information about φ and ϕ modulo p_2, and $\delta_2 = \frac{\varphi + \mathsf{vk}^* \cdot \phi}{\alpha - \mathsf{ID}^*}$. If $\mathsf{vk}' \neq \mathsf{vk}^* \bmod p_2$, then $\mathcal{A}$ can only compute such $\bar{\delta}_2$ value with negligible probability for two unknowns φ and ϕ modulo p_2. The case of $\mathsf{vk}' = \mathsf{vk}^* \bmod p_2$ is excluded by the $Game_{res}$.

Hence, $\mathcal{B}$ can break Assumption 2' if $\mathcal{A}$ can distinguish $\text{Game}_{\ell-1}$ and Game_ℓ. □

Lemma 18. *We can construct an algorithm $\mathcal{B}$ with advantage ϵ in breaking Assumption 3', if there exists $\mathcal{A}$ such that $Adv_{\mathcal{A}}$ ($Game_q$) - $Adv_{\mathcal{A}}(Game_{final}) = \epsilon$.*

Proof. Given $(g, X_3, g^a X_2, g^s Y_2, Z_2, g_4, T)$ from Assumption 3', $\mathcal{B}$ chooses random $\beta, \phi, \varphi, \nu \in \mathbb{Z}_N$ and sets

$$g_1 = g, \quad u_1 = g^\beta, \quad \hat{e}(g_1, h_1) = \hat{e}(g, g^a X_2), \quad v_1 = g^\phi, \quad w_1 = g^\varphi, \quad y_1 = u_1^{1/\nu}.$$

$\mathcal{B}$ implicitly sets $h_1 = g^a$. $\mathcal{B}$ sends the $\mathsf{param}, \mathsf{mpk}$ to $\mathcal{A}$.

$\mathcal{B}$ can calculate the semi-functional secret key for ID as follows. $\mathcal{B}$ computes (A_1, A_2) and its proof π using ν. The adversary sends the commitment R_1 to $\mathcal{B}$. $\mathcal{B}$ randomly picks $r, t_0 \in \mathbb{Z}_N$, $W_2, W_2', W_2'', W_2''' \in \mathbb{G}_{p_2}$ and $W_3, W_3', W_3'', W_3''' \in \mathbb{G}_{p_3}$ and calculates:

$$K_1' = (g^a X_2 \cdot R_1 \cdot u_1^{-r})^{\frac{1}{\alpha - \mathsf{ID}}} \cdot w_1^{t_0} \cdot W_2 W_3,$$
$$K_2' = g^r \cdot W_2' W_3', \quad K_3' = g^{t_0(\alpha - \mathsf{ID})} \cdot W_2'' W_3'', \quad K_4' = v_1^{t_0} \cdot W_2''' W_3''',$$

Therefore $\mathcal{B}$ can answer all extraction oracle and decryption oracle queries.

For the decryption oracle query, $\mathcal{B}$ calculates the identity-based secret key as the key extraction first and then outputs the decryption result.

In the challenge phase, $\mathcal{B}$ randomly chooses $b' \in \{0,1\}$, $X_4, X_4', X_4'' \in \mathbb{G}_{p_4}$ and runs $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}(1^\lambda)$. It calculates the SF ciphertext:

$$C_0' = M_{b'}^* \cdot T, \quad C_1' = (g^s Y_2)^{\alpha - \mathsf{ID}^*} X_4, \quad C_2' = (g^s Y_2)^{\beta} X_4', \quad C_3' = (g^s Y_2)^{\phi \cdot \mathsf{vk}^* + \varphi} X_4'',$$

where $\sigma^* \leftarrow \mathsf{Sign}(\mathsf{sk}^*, C_0' || C_1' || C_2' || C_3')$. If $T = \hat{e}(g,g)^{as}$, then $\mathcal{B}$ simulates Game_q. Otherwise, then $\mathcal{B}$ simulates Game_{final}. If $\mathcal{A}$ can distinguish between these two games, $\mathcal{B}$ can break Assumption 3'.

For further decryption oracle query with $\mathsf{vk} = \mathsf{vk}^*$, $\mathcal{B}$ returns $\perp$ for invalid ciphertext (by the security of the strong one-time signature).

Therefore no PPT adversary $\mathcal{A}$ can distinguish between Game_q and Game_{final}. □

Back to the proof of our theorem. Finally in Game_{final}, the value of b' is information-theoretically hidden from $\mathcal{A}$. Hence $\mathcal{A}$ has no advantage in winning Game_{final}. If Assumption 1', 2' and 3' hold, then Game_{real} is indistinguishable from Game_{final}. Hence the attacker has negligible advantage in winning Game_{real}. Therefore, our scheme is IND-ID-CPA secure. □

Dishonest User Security. We now prove the black-box traceability.

Theorem 10. *If Assumptions 1',2' and 4' hold, then no PPT adversary has non-negligible advantage in the adaptive-ID ComputeNewKey-CCA game.*

Proof. We first consider the probability that an iteration of the tracing algorithm increases the value ctr.

Lemma 19. *In the adaptive-ID ComputeNewKey-CCA game, if $\mathbb{D}^*$ correctly opens well-formed ciphertexts with probability ϵ, then the probability that an iteration of the tracing algorithm increases ctr is at least $p_1 \geq \epsilon - \delta'$, where δ' is the probability of solving Assumption 1', 2' or 4'.*

Proof. We consider a sequence of Games. Let Game Real be the original adaptive-ID ComputeNewKey-CCA game. Let Game 0 be the same as Game Real except that the adversary cannot ask for the secret key of identity as $\mathsf{ID} = \mathsf{ID}^* \bmod p_2$. Denote q as the maximum number of extraction oracle and decryption oracle query. For $i \in [1,q]$, denote Game i as the same as Game 0 except the keys and the values (A_1, A_2) used to answer the first i^{th} distinct ID oracle query are semi-functional. The keys for the remaining queries are normal.

Similar to the IND-ID-CPA proof, Game Real and Game 0 are indistinguishable if Assumption 1' and 2' hold. Similar to Lemma 16, we can show that Game i and Game $i-1$ are indistinguishable if Assumption 2' holds. The only difference is to simulate a "SF" version of (A_1, A_2), which means that they have a $\mathbb{G}_{p_2}$ part. It cam be simulated using T from Assumption 2' for the ℓ^{th} distinct ID query. The algorithm $\mathcal{B}$ picks a random $r_2' \in \mathbb{Z}_N$ and uses $T^{\nu r_2'}$ and $T^{r_2'}$ to simulate A_1 and A_2 respectively. $\mathcal{B}$ uses the simulator of the concurrent zero knowledge proof of knowledge to generate a correct π. The rest of the proof follows the proof of Lemma 16.

Finally, consider the following simulation of Game q. From Assumption 4', we have $(g_1, g_2, g_3, g_4, u_1 u_4, u_1^r Y_2, g_1^r Z_2, g_1^s W_2 W_4)$ $\mathcal{B}$ chooses random $\alpha, \phi, \varphi \in \mathbb{Z}_N$, $y_1, h_1 \in \mathbb{G}_{p_1}$ and sets $w_1 = g_1^{\varphi}, v_1 = g_1^{\phi}$. $\mathcal{B}$ implicitly sets msk as $(h_1, \alpha, \nu = \log_{y_1} u_1)$. $\mathcal{B}$ honestly sets the rest of param and mpk. $\mathcal{B}$ sends the $\mathsf{param}, \mathsf{mpk}$ to $\mathcal{A}$.

$\mathcal{B}$ can calculate the semi-functional secret key for ID as follows. $\mathcal{B}$ picks some random $r_2' \in \mathbb{Z}_N$, $X_2, X_2' \in \mathbb{G}_{p_2}$ and computes:

$$A_1 = (u_1^r Y_2)^{r_2'} X_2, \quad A_2 = (g_1^r Z_2)^{r_2'} X_2'.$$

It implicitly sets $r_2 = r \cdot r'_2$. $\mathcal{B}$ uses the simulator of the zero knowledge proof of knowledge to generate a correct π, without knowing ν. The adversary sends the commitment R_1 to $\mathcal{B}$. $\mathcal{B}$ randomly picks $r'_1, t \in \mathbb{Z}_N$, $\bar{R}_2, \bar{R}'_2, \bar{R}''_2, \bar{R}'''_2 \in \mathbb{G}_{p_2}$ and $\bar{R}_3, \bar{R}'_3, \bar{R}''_3, \bar{R}'''_3 \in \mathbb{G}_{p_3}$ and calculates:

$$K'_1 = (h_1 \cdot R_1 \cdot (u_1^r Y_2)^{-r'_1})^{\frac{1}{\alpha - \mathsf{ID}}} \cdot w_1^t \cdot \bar{R}_2 \cdot \bar{R}_3,$$
$$K'_2 = (g_1^r Z_2)^{r'_1} \cdot \bar{R}'_2 \cdot \bar{R}'_3, \quad K'_3 = g_1^{t(\alpha - \mathsf{ID})} \cdot \bar{R}''_2 \cdot \bar{R}''_3, \quad K'_4 = v_1^t \cdot \bar{R}'''_2 \cdot \bar{R}'''_3.$$

It implicitly sets $r_1 = r \cdot r'_1$. Therefore $\mathcal{B}$ can answer all extraction oracle and decryption oracle queries.

Finally, $\mathcal{A}$ returns a challenge identity ID^*, an identity-based secret key $\mathsf{sk}^*_{\mathsf{ID}^*} = (K^*_1, K^*_2, K^*_3, K^*_4)$ and a decoder box $\mathbb{D}^*$ that correctly decrypts a fraction ϵ of ciphertexts. In the tracing stage, $\mathcal{B}$ randomly chooses $M^* \in \mathbb{G}_T, Q_2, Q'_2 \in \mathbb{G}_{p_2}, Q_4, Q'_4 \in \mathbb{G}_{p_4}$, runs $(\mathsf{vk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KGen}(1^\lambda)$ and calculates the (SF) ciphertext:

$$C'_1 = (g_1^s W_2 W_4)^{\alpha - \mathsf{ID}^*} \cdot Q_2 Q_4, \quad C'_2 = T, C'_3 = (g_1^s W_2 W_4)^{\varphi + \mathsf{vk}^* \phi} \cdot Q'_2 Q'_4,$$
$$C'_0 = M^* \cdot \hat{e}(C'_1, K^*_1 K^{*\mathsf{vk}^*}_4) \cdot \hat{e}(C'_2, K^*_2) / \hat{e}(C'_3, K^*_3),$$

and $\sigma^* = \mathsf{Sign}(\mathsf{sk}^*, C'_0 || C'_1 || C'_2 || C'_3)$. $\mathcal{B}$ provides $\mathfrak{C}^* = (C'_0, C'_1, C'_2, C'_3, \sigma^*, \mathsf{vk}^*)$ to $\mathbb{D}^*$ as a single iteration of the tracing algorithm. If $T = u_1^s V_2 V_4$, then $\mathcal{B}$ provides a properly formed encryption of M^* to $\mathbb{D}^*$. Otherwise, $\mathcal{B}$ provides a malformed ciphertext as used in the tracing algorithm.

$\mathcal{B}$ runs a single iteration of the tracing algorithm to $\mathbb{D}^*$ using $\mathfrak{C}^*$. If $\mathcal{A}$ wins, the *ctr* should only increase by some negligible probability δ'. When $T = u_1^s V_2 V_4$, the *ctr* always increases by 1. When T is randomly chosen from $\mathbb{G}_{p_1 p_2 p_4}$, $\mathfrak{C}^*$ is a "malformed" ciphertext used for the tracing algorithm (i.e. the $\mathbb{G}_{p_1}$ part of $T \neq u_1^s$) with overwhelming probability.

If Assumption 4' holds, no PPT algorithm can distinguish between these two cases. Therefore, $\mathcal{B}$ should obtain the same result if $\mathfrak{C}^*$ is feed into $\mathbb{D}^*$. However, for the case of $T = u_1^s V_2 V_4$, the *ctr* always increases by 1. It contradicts that the *ctr* should only increase by some negligible probability δ'. Hence, if $\mathcal{A}$ wins, $\mathcal{B}$ can break Assumption 4'. □

Now, observe that the tracing algorithm points to the PKG if $ctr = 0$ at the end. The variable *ctr* can be seen as the sum of $L = 16\lambda/\epsilon$ independent random variables $X_i \in \{0, 1\}$ having the same expected value p_1. We have $\mu = E[ctr] = Lp_1$. The Chernoff bound shows that, for any real number ω such that $0 \leq \omega \leq 1$, $\Pr[ctr < (1 - \omega)\mu] < \exp(-\mu\omega^2/2)$. Under Assumption 1', 2' and 4', we have $\delta' \leq \epsilon/2$ (since we assume $\epsilon/2$ is non-negligible). From Lemma 19, we have

$$\mu = Lp_1 \geq L(\epsilon - \delta') \geq \frac{L\epsilon}{2} = 8\lambda.$$

With $\omega = 1/2$, the Chernoff bound guarantees that

$$\Pr[ctr < 1] < \Pr[ctr < 4\lambda] = \Pr[ctr < \mu/2] < \exp(-\mu/8) = \exp(-\lambda).$$

□

Dishonest PKG Security. Finally, we prove the security against dishonest PKG.

Theorem 11. *If Assumptions 1' and 5' hold, then no PPT adversary has non-negligible advantage in the black-box FindKey-CCA game.*

Proof. Firstly, we show that the probability of increasing *ctr* in each iteration is low by the following lemma.

Lemma 20. *In the black-box FindKey-CCA game, one iteration of the tracing algorithm increases ctr with probability at most δ', if δ' is the probability of solving Assumption 1' or 5'.*

Proof. It can be proved by two games. In Game_0, the single iteration of the tracing algorithm uses a ciphertext $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ as described in the tracing algorithm. In Game_1, the single iteration of the tracing algorithm uses a ciphertext $\mathfrak{C} = (C_0, C_1, C_2', C_3, \sigma, \mathsf{vk})$ with C_2' has a random $\mathbb{G}_{p_2}$ element. It can be seen as a "semi-functional" ciphertext at the C_2 part only.

The indistinguishability of Game_0 and Game_1 can be proved as the proof of Lemma 15. We only need to change the C_2 part from normal to semi-functional. Therefore these two games are indistinguishable if Assumption 1' holds.

Next, we consider the case of Game_1. Given $(g_1, y_1, g_3, g_4, y_1^r, g_1^r X_2, y_1^s Y_2, T)$ from Assumption 5', $\mathcal{B}$ sets $g_1, y_1, g_3, g_4, N, \hat{e}, \mathbb{G}, \mathbb{G}_T$ as part of the public parameters param. $\mathcal{B}$ generates the crs of the concurrent zero knowledge proof of knowledge with a knowledge extractor. $\mathcal{B}$ sends the param to $\mathcal{A}$ in the Initialize phase. $\mathcal{A}$ chooses the master secret keys and sends the related master public key $\mathsf{mpk} = (g_1^\alpha, \hat{e}(g_1, h_1), u_{14})$ and a challenge identity ID^* to $\mathcal{B}$.

During the Extract phase for ID, $\mathcal{A}$ sends (A_1, A_2) and a proof π to $\mathcal{B}$. $\mathcal{B}$ uses the knowledge extractor to get the values (ν, r_2) such that the $\mathbb{G}_{p_1}$ part of A_1 and A_2 are $y_1^{\nu r_2}$ and $g_1^{r_2}$ respectively. $\mathcal{B}$ picks some random $\theta \in \mathbb{Z}_N$ and sends to the PKG

$$R_1 = (y_1^r)^{-\nu r_2} (g_1^{\alpha - \mathsf{ID}})^\theta,$$

which implicitly sets $r_0 = r$. $\mathcal{A}$ returns $(\hat{K}_1, \hat{K}_2, \hat{K}_3, \hat{K}_4)$ to $\mathcal{B}$. $\mathcal{B}$ randomly picks $Y_3, Y_3', Y_3'', Y_3''' \in \mathbb{G}_{p_3}$ and computes

$$K_1 = \hat{K}_1 Y_3 / g_1^\theta, \quad \tilde{K}_2 = \hat{K}_2 (g_1^r X_2)^{r_2} Y_3', \quad K_3 = \hat{K}_3 Y_3'', \quad K_4 = \hat{K}_4 Y_3'''.$$

Then $\mathcal{B}$ checks if

$$\hat{e}(K_1, g_1^{\alpha - \mathsf{ID}}) = \hat{e}(g_1, h_1) \cdot \hat{e}(u_{14}, \tilde{K}_2) \cdot \hat{e}(w_1, K_3) \quad \wedge \quad \hat{e}(K_3, v_1) = \hat{e}(g_1^{\alpha - \mathsf{ID}}, K_4).$$

Note that the correct value of $K_2 = \hat{K}_2 A_2^r Y_3'$ is not known by $\mathcal{B}$. Therefore, $\mathcal{B}$ only has the "semi-functional" key of $(K_1, \tilde{K}_2, K_3, K_4)$ instead of the actual key $\mathsf{sk}_{\mathsf{ID}^*} = (K_1, K_2, K_3, K_4)$. However, $\mathcal{B}$ can still answer all decryption oracle query using the "semi-functional" key. It is because $\mathcal{A}$ have not seen any $\mathbb{G}_{p_2}$ element so far and cannot produce a "semi-functional" ciphertext. For all normal ciphertext queried in the decryption oracle, the $\mathbb{G}_{p_2}$ part of $\tilde{K}_2$ will disappear after the pairing operation with C_2.

Finally, $\mathcal{A}$ outputs an ϵ-useful decoder box $\mathbb{D}$. If the tracing algorithm increases ctr, $\mathbb{D}$ must decrypt any ciphertext as if using the real $\mathsf{sk}_{\mathsf{ID}^*}$. $\mathcal{B}$ randomly picks $s' \in \mathbb{Z}_N$, $X_4, X_4', X_4'' \in \mathbb{G}_{p_4}$, $M \in \mathbb{G}_T$ and runs $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KGen}(1^\lambda)$. $\mathcal{B}$ computes

$$C_0 = M \cdot \hat{e}(g_1, h_1)^{s'}, \; C_1 = g_1^{s'(\alpha - \mathsf{ID}^*)} X_4, \; C_2 = (y_1^s Y_2)^\nu \cdot X_4', \; C_3 = (y_{14} v_1^{\mathsf{vk}} u_1)^{s'} X_4'',$$

and $\sigma = \mathsf{Sign}(\mathsf{sk}, C_0 || C_1 || C_2 || C_3)$. $\mathcal{B}$ submits $\mathfrak{C} = (C_0, C_1, C_2, C_3, \sigma, \mathsf{vk})$ to the decoder box $\mathbb{D}$. Note that $\mathbb{D}$ (generated by the dishonest PKG) is able to recognize invalid ciphertexts in the tracing stage (by using α, ν and the pairing). However, as long as $\mathbb{D}$ is assumed stateless, it cannot shutdown or self-destruct when detecting a tracing attempt. $\mathbb{D}$ tries to decrypt such invalid ciphertexts in the same way as the owner of the identity-based secret key $\mathsf{sk}_{\mathsf{ID}^*}$, and outputs a decrypted message M^*. Observe that $\mathbb{D}$ should not output $\perp$ since σ is a valid strong one-time signature. If this iteration of the tracing algorithm increases ctr with probability ϵ, then it means that with the same probability,

$$M^* = \frac{C_0 \cdot \hat{e}(C_3, K_3)}{\hat{e}(C_1, K_1 K_4^{\mathsf{vk}}) \cdot \hat{e}(C_2, K_2)},$$

Then $\mathcal{B}$ calculates

$$\begin{aligned} T &= \frac{C_0 \cdot \hat{e}(C_3, K_3)}{M^* \cdot \hat{e}(C_1, K_1 K_4^{\mathsf{vk}}) \cdot \hat{e}(C_2, \hat{K}_2)} \\ &= \hat{e}(C_2, K_2 \cdot \hat{K}_2^{-1}) = \hat{e}((y_1^s Y_2)^\nu X_4', A_2^r Y_3') = \hat{e}(y_1^{\nu s}, g_1^{r_2 r}), \end{aligned}$$

and uses $T^{1/r_2 \nu}$ as the solution of breaking Assumption 5'. $\square$

Finally, the dishonest PKG is not detected if it outputs $\mathbb{D}$ and the tracing stage ends with a non-zero value of ctr. From Lemma 20, we have:

$$\Pr[ctr \neq 0] = \Pr[ctr \geq 1] = 1 - (1 - \delta')^L \leq L\delta' = \frac{16\lambda\delta'}{\epsilon} \leq \frac{16\lambda}{\epsilon \cdot p(\lambda)},$$

for any positive polynomial $p(\cdot)$, if Assumption 5' holds. □

H Comparison with Exponent-Inversion Signatures, IBE, and More

H.1 Discussion on Linear IBE

Boyen [12] proposed a framework of *Linear IBE* which was an abstraction of IBE that captures the properties of the exponent inversion paradigm (including the SK-IBE and the BB_2-IBE). Linear IBE can be used to construct hierarchical IBE, fuzzy IBE, and attribute-based encryption [12]. However, their construction uses the fact that the session key is of the form $\hat{e}(g, \hat{g})^s$, where $g, \hat{g}$ are public parameters that are independent of the master secret key. In our construction, the session key is of the form $\hat{e}(g_1, h_1)^s$ where h_1 is part of the master secret key. Therefore our dual system Gentry-IBE construction does not belong to the linear IBE family. In fact, our session keys are more similar to the commutative blinding family (which is a function of the master secret key and s). Therefore, our construction has a key structure of the exponent inversion family and the session key of the commutative blinding family.

H.2 Discussion on Boneh-Boyen Signatures

The Boneh-Boyen signature scheme is provably secure in the standard model under the q-SDH assumption [9]. It is shown that the converse of this statement is also true [23], and hence forging Boneh-Boyen signatures is equivalent to solving the q-SDH problem. However, we cannot reduce the security of our dual form Boneh-Boyen signatures to the original scheme in [9] nor to the q-SDH problem. It is because the our scheme includes g^r as part of the signature, while the original scheme in [9] only outputs r. In addition, the secrecy of the values (h_1, α) is essential in our security proof, while the q-SDH problem only guarantees the secrecy of α.

H.3 Discussion on Accumulator and Verifiable Random Function

Changing to the subgroup-type assumption requires some randomness in the scheme. It is non-trivial that how this conversion can be done for cryptosystems that outputs a deterministic value. An important future work is how to remove the q-type assumption for such kind of cryptosystems like verifiable random function and accumulator.