J Cryptol (2017) 30:735–759
DOI: 10.1007/s00145-016-9234-8

Journal of
CRYPTOLOGY

Short Signatures from Diffie–Hellman: Realizing Almost Compact Public Key

Jae Hong Seo
Department of Mathematics, Myongji University, Yongin, Republic of Korea
jaehongseo@mju.ac.kr

Communicated by Hugo Krawczyk.

Received 14 December 2014 / Revised 16 June 2016
Online publication 10 August 2016

Abstract. In this paper, we present a new digital signature scheme based on the computational Diffie–Hellman (CDH) assumption in the standard model. The proposed signature scheme is not only asymptotically almost compact but also practical for concrete parameters in the sense that the public key has 29 group elements, and the signature consists of two group elements and two exponents for 80-bit security. Note that the Waters signature scheme, which is the previous best digital signature scheme in the same category (CDH assumption, standard model), requires linear-sized public keys in the security parameter, particularly those with 164 group elements for 80-bit security. To achieve our goal, we revisited the CDH-based signature scheme proposed by Hohenberger and Waters (EUROCRYPT 2009), which is a stateful signature scheme but achieves asymptotically compact parameters in the sense that its public key and signature consist of constant group elements. We modify the Hohenberger–Waters signature scheme to remove the state information from the signatures. More precisely, we use programmable hashes and random tags, instead of counters which is the state information maintained by a signer. To prove the security of the proposed signature scheme, we developed prefix-guessing technique for *random tags*. Note that the prefix-guessing technique was first introduced by Hohenberger and Waters (CRYPTO 2009) and was originally used for *message queries*.

Keywords. Digital Signature, Standard Model, Computational Diffie-Hellman.

1. Introduction

Digital signature schemes are one of the fundamental primitives of modern cryptography and are used in many cryptographic protocols as an important building block. From both practical and theoretical standpoints, it is important to design efficient signature schemes whose security is proven under reliable assumptions. Most efficient signature schemes follow a hash-and-sign paradigm (rather than a tree-based approach [19,22]) to obtain efficient signatures, particularly short signatures. Collision-resistant hash functions mapping from an arbitrarily long message to a short bit-string is first used, and then, the hashed message is signed onto. Each of the hash-and-sign signature schemes requires relatively

strong assumptions (e.g., random oracles [2,11,23,26,27,41,43], strong RSA assumptions [20,24,25], q-strong type assumptions [8,30,32,42], or LRSW assumptions [13]), inefficient signing/verification processes [30,34], long public parameters [46], or maintenance of the state by the signer [33]. However, there is an approach to prove the full security of signature schemes in the standard model using a decisional assumption. More precisely, there are several identity-based and functional encryption schemes that can be transformed into digital signature schemes via the Naor transformation [10], which have been proven secure under decisional Diffie–Hellman assumptions or decisional linear assumptions by using the standard model. However, all these decisional assumptions are stronger than the CDH assumption. In fact, there is a series of works on the construction of an efficient signature scheme based on standard assumptions such as RSA and CDH (rather than decisional assumptions) in the standard model; these studies were mainly those conducted by Hohenberger and Waters [33,34], Hofheinz et al. [30], and Yamada et al. [48]. In this study, we aim to develop new techniques for constructing efficient CDH-based signature schemes.

Our Result. The Waters signature scheme [46] was the previous best construction in the category of hash-and-sign, CDH-based, and standard model signature schemes.[1] It has a constant signature size (two group elements) and linear public keys in the security parameter. Achieving sublinear public keys (in the category of hash-and-sign, CDH-based, and standard model signatures) has been an important open problem. We provide an answer by proposing a new signature scheme with almost compact public keys. In fact, our signature scheme achieves *asymptotically almost compact* parameters in the sense that the signature size is constant (two group elements and two exponents)[2] in the security parameter, and the public keys can be of the size $\omega(1)$ (e.g., $\log\log\lambda$); that is, any function in $\omega(1)$ is possible.[3]

Furthermore, our security analysis can yield not only asymptotically almost compact parameters but also reasonably short parameters to serve as concrete security parameters. For instance, in the case of 80-bit security, a signature consists of two group elements and two exponents and a public key consists of 29 group elements and a description of a collision-resistant hash function, which is considerably shorter than the public key size (164 group elements and a description of a collision-resistant hash function) of the Waters signature scheme [46] for the same security parameter; however, the reduction losses of both schemes are the same $O(\lambda q)$.[4]

Our Strategy. We first revisit the Hohenberger–Waters signatures (EUROCRYPT 2009 [33]), which is a CDH-based, standard model, hash-and-sign signature scheme and has compact parameters (in the sense that public key size and the signature size are

[1] There is a subsequent work, which is independent of our work and has better asymptotic performance than the Waters signatures. We will compare it at the end of this section.

[2] We can reduce the signature size to two group elements and one exponent under an additional standard assumption of the existence of pseudorandom function; this will be discussed in Sect. 5.1.

[3] The public key size is not constant, but any strictly increasing function is possible, and hence, we say that our scheme achieves "*asymptotically almost compact*" parameters.

[4] There are some variants of the Waters signatures [31,34,39]. However, the basic construction framework is essentially the same as that of the Waters signatures.

constant in the security parameter). However, their scheme is stateful; that is, it forces a signer to maintain the state information. This property is not desirable in general but is acceptable in some applications and meaningful for various reasons. In fact, Hohenberger and Waters viewed their result as a step toward realizing practical, standard model signatures under standard assumptions in a *stateless* manner, and expected similar progress outcomes to the advance of the early tree-based signature schemes such that first, a stateful signature scheme is proposed (e.g., GMR signatures [29]), and then, the method to remove the state is developed (e.g., Goldreich [28]). We revisit the Hohenberger–Waters signature scheme and construct a stateless, CDH-based, standard model signature scheme on the basis of the Hohenberger–Waters signature scheme. That is, one can consider our proposal as a stateless version of the Hohenberger–Waters signatures.

The main idea behind Hohenberger–Waters signatures is the use of counters, which is the state information maintained by a signer. Since all counters are distinct and the number of counters used in practice is bounded by a polynomial, in the security proof, the simulator can restrict the adversaries to use one counter in the forgery and can guess such the counter with non-negligible probability. If the simulator's guess is correct, we can apply a selectively secure signature scheme's proof technique (e.g., [7]). Since our aim is a stateless signature scheme, we cannot use this technique directly. Instead of a counter, we arrange a random tag chosen from $[1, Q] \cap \mathbb{Z}$ for each signature, where Q is an appropriately large integer and will be specified later. When using random tags, we have to be careful since the same tags can be used several times in contrast to ordered counters, and this could be troublesome in the security proof. To handle tag duplications, we use a different technique called programmable hash functions [32]. A naive combination of these techniques allows us to obtain short and stateless CDH-based signatures with a *somewhat* short public key of $\Theta(\frac{\lambda}{\log \lambda})$ group elements, but we cannot achieve the standard security notion of *existential unforgeability against the chosen-message attack (EUF-CMA)*. Instead, we can achieve only a rather weaker *bounded CMA* security notion, which guarantees that the signature scheme is only secure against a fixed number (polynomial but predetermined at the time of parameter generation) of signing queries. In the security proof, the simulator has to guess in advance the random tag used for the forgery. To this end, the domain $[1, Q]$ for random tags should be a polynomial in the security parameter, which is larger than q, where q denotes the maximum number of signing queries. That is, if Q is small, there could be a very large number of duplications among the random tags, and thus, we cannot control such the case even by using the programmable hashes. If Q is large (e.g., a superpolynomial), then we cannot attain polynomial-time reduction. More precisely, $Q > q$ is required to apply the birthday lemma so that q is bounded above at the time of parameter generation.

Our main scheme is a generalization of the previous naive combination of Hohenberger–Waters signatures and programmable hashes; a signature has a random tag *vector* instead of a random tag. Surprisingly, our main scheme not only achieves shorter public keys but is also EUF-CMA secure. Let us briefly explain why our generalization, which is seemingly very natural but non-trivial to analyze, can lead improvements in both efficiency and security. If the overall length of a tag vector is sufficiently large, which should be a superpolynomial, we can show that there are no $(m + 1)$ collisions in the set of q random tag vectors, with a $1 - neg(\lambda)$ probability, where m denotes some fixed public parameter and $neg(\lambda)$ represents a negligible function in the security

parameter λ. This probability can be calculated from the generalized birthday lemma. For the domain of tag vectors, we use $[1, \lambda]^k$, where k is a function in $\omega(1)$, and we can show that it satisfies the above probability. Therefore, we do not require a parameter associating the maximum number of signing queries q for the domain of tags, and thus, we can avoid the bounded CMA security notion. Furthermore, the public key size is bounded from the above-mentioned $O(m + k)$, and our parameter selection enables $m = k$ so that we can attain asymptotically almost compact public keys; that is, the public key size could be any (fixed) function in $\omega(1)$ (e.g., $\log \log \lambda$). There is one more important issue related to the security proof. Although we increase the overall length of the tag vectors, a trivial simulation will end up with an exponential-time reduction since it cannot guess the target tag vectors with a non-negligible probability. To resolve this, we extend the prefix-guessing technique, which was first introduced by Hohenberger and Waters (CRYPTO 2009 [34]). Hohenberger and Waters used the prefix-guessing technique for the target message in the simulation; the simulator guesses the shortest prefix of the target message, which is not a prefix of any of the messages queried by the adversary, before the simulator generating the parameters.[5] We apply the prefix-guessing technique to the random tag vectors instead of the messages. In [34], each message bit is separately handled to enable prefix guessing with a high probability, although several bits (at most $\log \lambda$) can be handled via a trivial trade-off between efficiency and reduction loss [39]. For random tag vectors, we have to consider the case wherein several tags have the same prefix. We extend the Hohenberger–Waters prefix-guessing technique to handle duplications among random tag vectors; this allows the simulator to guess a prefix of the target tag vector with a high probability. Therefore, our prefix-guessing technique is of independent interest.

Related Works (Hash-and-Sign Signatures). By relying on the random oracle heuristic, many constructions for efficient signature schemes in several settings have been proposed (e.g., DL setting [11,23,27,41,43], RSA setting [2], and Lattice setting [26]). However, there have been some studies showing the limitations of the random oracles [14,21,37].

In the DL setting, Boneh and Boyen [8] proposed the first signature scheme in the standard model where the signature size is comparable with that of the BLS signature scheme [11] in the random oracle model. Okamoto [42] proposed a signature scheme that is more effective in many applications such as blind signatures, group signatures, and anonymous credentials. Recently, Hofheinz et al. [30,32] proposed short signatures using programmable hash functions where the signature size is a bit shorter than that in the previous schemes. However, the security of all these signature schemes are proven under non-static q-type assumptions. The size of the problem instance of q-type assumptions is (linearly) increased according to the number of signing queries. There are analyses for the q-type assumptions [12,18]. Camenisch and Lysyanskaya [13] proposed a signature scheme that can be used for constructing efficient group signatures and identifying escrow schemes and anonymous credential systems. They proved the

[5] Therefore, the prefix-guessing technique is suitable for weak CMA security, where the adversary issues all message queries before receiving the parameters. Note that there is a general transformation from EUF-wCMA secure signatures to EUF-CMA secure signatures via the Chameleon hashes. We also use the same strategy using the Chameleon hashes based on the discrete logarithm assumption [36].

security of their signature scheme under an interactive assumption called the LRSW assumption [38]. Interactive assumptions are non-falsifiable assumptions, and there is a considerable amount of criticism for non-falsifiable assumptions [40].

To the best of our knowledge, there were only two signature schemes based on the standard CDH assumption in the standard model [33,46]. However, the signer of the signature scheme in [33] needs to maintain certain states, i.e., a stateful signature scheme. Therefore, there is only one construction for stateless signatures [46] that has been proven secure under the standard CDH assumption in the standard model. However, the signature scheme proposed in [46] has a large public key $\Theta(\lambda)$ as compared to all the aforementioned signature schemes based on the strong assumptions.

Note that there is a generic transformation from the IBE scheme to the signature scheme, called the Naor transformation, and there are efficient DH-based IBE constructions in the standard model [3,16,17,35,47]. However, signature schemes transformed from these IBE constructions inherently require decisional (as opposed to search) assumptions such as the decisional linear assumption and the decisional DH assumption.

In the RSA setting, the first standard model construction was developed by Gennaro et al. [25]. Subsequently, Cramer and Shoup [20] and Fischlin [24] proposed more efficient signature schemes. These schemes were proven secure under the strong RSA assumption. Hohenberger and Waters proposed the first hash-and-sign signatures from the RSA assumption [33]. However, their scheme requires the signer to maintain states, i.e., a stateful signature scheme. In the same year, they proposed the first-ever stateless RSA-based signatures [34]. Subsequently, Hofheinz et al. [30] and Yamada et al. [48] improved its efficiency. However, all these signature schemes based on the RSA assumption require a large number of primality tests at the signing and verifying stages.

Independent Work. Independently of our work, Böhl et al. [5,6] proposed almost the same CDH-based signature scheme but with a completely different security analysis. Both schemes are the same except the length of each component in a tag vector. More precisely, all components of a tag vector are of equal length in our construction, but of various lengths in [5,6]. These minor differences in their scheme are crucial for their new proof strategy, called confined guessing; most notably, the reduction algorithm behaves differently according to the adversarial success probability, and various tag lengths are essential for realizing this proof strategy.[6] Confined guessing approach can be applied to several settings such as RSA and SIS, and the CDH-based signature scheme is one instantiation of their strategy.

Comparison. We give a brief comparison with Waters scheme [46] and BHJKS scheme [5,6] in Table 1.[7] BHJKS scheme achieves sublinear public keys ($O(\log \lambda)$ group elements), which is smaller than the Waters scheme but larger than ours (e.g.,

[6] According to adversarial success probability, the simulator in the confined guessing strategy chooses a tag component with appropriate size so that we say that various tag component lengths are essential in Böhl et al. scheme. In contrast, our prefix-guessing proof approach requires that components in a tag vector are of equal length.

[7] Note that Naccache [39] proposed a variant of Waters signatures that offers a trade-off between the public key size and the tightness of the security reduction, but the asymptotic behavior is the same as the Waters signatures.

Table 1. EUF-CMA secure CDH-based signature schemes.

Scheme	PK size	Sig. size	Reduction loss	Sec. model
Waters [46]	$O(\lambda)\tau_{\mathbb{G}}$	$2\tau_{\mathbb{G}}$	$O(\lambda q)$	EUF-CMA
BHJKS [5,6]	$O(\log_d \lambda)\tau_{\mathbb{G}}$	$2\tau_{\mathbb{G}} + 1\tau_{\mathbb{F}_p}$	$O\left(\frac{2^{2+\frac{d}{m'}} q^{\frac{d}{m'}+d}}{\varepsilon^{\frac{d}{m'}}}\right)$	EUF-CMA
This paper (tag-free)*	$O(f(\lambda))\tau_{\mathbb{G}}^\dagger$	$2\tau_{\mathbb{G}} + 2\tau_{\mathbb{F}_p}$ $(2\tau_{\mathbb{G}} + 1\tau_{\mathbb{F}_p})$	$O(\lambda q)$	EUF-CMA

$\tau_{\mathbb{G}}$ is the size of group element, $\tau_{\mathbb{F}_p}$ is the size of the exponent
In [5,6], m' and $d > 1$ are constants and ε is the adversarial success probability
*A variant scheme under an additional assumption of the existence of pseudorandom functions
† $f(\lambda)$ can be any strictly increasing function, that is, $f \in \omega(1)$; e.g., $\log\log\lambda$

$O(\log\log\lambda)$ group elements). Contrary to the other schemes, the reduction loss $O\left(\frac{2^{2+\frac{d}{m'}} q^{\frac{d}{m'}+d}}{\varepsilon^{\frac{d}{m'}}}\right)$ of the BHJKS scheme depends on the adversarial success probability ε. Since d and m' are constants defined in the scheme and q are polynomial in the security parameter λ (if we assume polynomial-time adversaries), the numerator is polynomial in λ. If ε is constant, the quantity in the big-O notation is comparable to the reduction loss λq of the other schemes. However, for adversaries with small (but still non-negligible) success probability ε, the reduction loss becomes much larger than that of the other schemes; for instance, if $\varepsilon = 1/(\lambda q)^{2m'/d}$ (polynomial in λ), then the reduction loss is larger than $(\lambda q)^2$. On the other hand, the reduction loss of Waters scheme and ours is $O(\lambda q)$, regardless of the adversarial success probability.

Journal Version Versus Conference Version. The current paper is a journal version of the first part of the conference paper (EUROCRYPT 2013) [4][8]. The security analysis given in the conference version guarantees only the bounded CMA security notion. More precisely, the prefix-guessing technique used in [4,44] inherently requires the domain of random tag vectors to be of the form $[1, Q]^k$ satisfying $Q > q$, where q is the maximum of the allowable signing queries in the proof so that q should be bounded by Q at the parameter generation time. In this journal version, we endeavor to remove the undesirable restriction $Q > q$ of the conference version [4,44].

Outline. In the next section, we give useful preliminaries for explaining our results. In Sect. 3, the first approach toward stateless CDH-based signatures and its limitations are presented. Our main CDH-based signature scheme and its security analysis are presented in Sect. 4. In particular, we give practical parameters for concrete security parameters $\lambda \in \{80, 128, 192, 256\}$ in Sect. 4.1. In Sect. 5, we give two extensions for shorter signatures using a pseudorandom function and using asymmetric pairings.

[8] Böhl et al. [4] is the merged paper of [5,44], and the full version of the second part of [4] is published in [6].

2. Preliminaries

In this section, we describe the notation used in this paper, the standard definition of a digital signature scheme and its security, and a background on the underlying bilinear group structure and complexity assumption.

Notation. Throughout the paper, λ means the security parameter. For $a, b \in \mathbb{Z}$, $[a, b]$ means $\{a, a+1, \ldots, b\}$. For vectors $\overrightarrow{v}$ and $\overrightarrow{w}$, $\langle \overrightarrow{v}, \overrightarrow{w} \rangle$ means the standard dot product between $\overrightarrow{v}$ and $\overrightarrow{w}$. For a probabilistic algorithm Alg, $Alg(x) \rightarrow a$ means that Alg assigns the result to a on input x with uniformly chosen random coins. If we emphasize the random coins r of Alg, then we use $Alg(x; r)$. If the input of Alg is clear from the context, we sometimes omit it and simply write $Alg \rightarrow a$. If Alg terminates in polynomial time in λ, then Alg is called a probabilistic polynomial-time (PPT) algorithm. For a set S, $s \stackrel{\$}{\leftarrow} S$ denotes that the element s is uniformly chosen from S. A negligible function is a function $\mu(\lambda) : \mathbb{N} \rightarrow \mathbb{R}$ such that for every positive polynomial $poly(\cdot)$, there exists a positive integer N_{poly} such that for all $\lambda > N_{\text{poly}}$, $|\mu(\lambda)| < \frac{1}{poly(\lambda)}$. For two real-valued functions a and b in λ, $a \sim b$ means that $|a - b|$ is a negligible function in λ.

2.1. *Syntax and Security of Signature Scheme*

Signature Scheme. A signature scheme with message space $\mathcal{M}$ consists of three PPT algorithms, KeyGen, Sign, and Verify.

KeyGen(λ): It takes the security parameter λ and outputs a pair of public key PK and secret key SK.

Sign(PK, M, SK): It takes PK, SK, and a message $M \in \mathcal{M}$ and outputs a signature σ.

Verify(PK, M, σ): It takes PK, $M \in \mathcal{M}$, and σ, and returns $b \in \{0, 1\}$, where the case $b = 1$ means that σ is a valid signature on M and the case $b = 0$ means that σ is invalid.

Correctness: KeyGen, Sign, and Verify must satisfy the following correctness condition.

$$\Pr[\mathsf{KeyGen}(\lambda) \rightarrow (PK, SK), \mathsf{Verify}(PK, M, \mathsf{Sign}(PK, M, SK)) \rightarrow 1] = 1.$$

Existential UnForgeability. The standard security notion for signature schemes, called *Existential UnForgeability with respect to Chosen-Message Attacks* (EUF-CMA), was formalized by Goldwasser et al. [29]. There is a slightly weaker model called Existential UnForgeability with respect to *weak* Chosen-Message Attacks (EUF-wCMA). The adversaries in both security models are given the public key and access to a signing oracle, and win if she can produce a valid pair of a signature and a message on which the adversary did not query to the signing oracle. In the EUF-CMA security model, the adversary is allowed to query any time before she outputs a forgery. However, the adversary in the EUF-wCMA model has to send the challenger the entire list of messages that she wants to query before receiving the public key; thus, we sometimes call

the adversary in the EUF-wCMA model, a non-adaptive adversary. Next, we provide the formal definition of the EUF-CMA secure signature scheme and the EUF-wCMA secure signature scheme. Let $\mathsf{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be a signature scheme. We now consider the two following experiments:

$\mathbf{Exp}^{\text{EUF-CMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda)$
$(PK, SK) \leftarrow \mathsf{KeyGen}(\lambda)$;
$(M, \sigma) \leftarrow \mathcal{A}^{\mathsf{Sign}(\cdot)}(PK)$;
Define L as the set of
all messages queried
by the adversary;
Return
$\begin{cases} 1 \text{ if } M \notin L \\ \quad \text{and } \mathsf{Verify}(PK, M, \sigma) = 1, \\ 0 \text{ otherwise.} \end{cases}$

$\mathbf{Exp}^{\text{EUF-wCMA}}_{\mathsf{SIG},\mathcal{A}=(\mathcal{A}_1,\mathcal{A}_2)}(\lambda)$
$(M_1, \ldots, M_q, st) \leftarrow \mathcal{A}_1(st)$;
$(PK, SK) \leftarrow \mathsf{KeyGen}(\lambda)$;
For $\forall i \in [1, q]$, $\sigma_i \leftarrow \mathsf{Sign}(PK, M_i, SK)$;
$(M, \sigma) \leftarrow \mathcal{A}_2(PK, \sigma_1, \ldots, \sigma_q, st)$;
Return
$\begin{cases} 1 \text{ if for } \forall i \in [1, q], \ M \neq M_i \\ \quad \text{and } \mathsf{Verify}(PK, M, \sigma) = 1, \\ 0 \text{ otherwise.} \end{cases}$

We define adversarial advantages of the above experimentations as follows:

$$\mathbf{Adv}^{\text{EUF-CMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda) = \Pr\left[\mathbf{Exp}^{\text{EUF-CMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda) = 1\right]$$
$$\text{and } \mathbf{Adv}^{\text{EUF-wCMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda) = \Pr\left[\mathbf{Exp}^{\text{EUF-wCMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda) = 1\right].$$

Definition 1. Let SIG be a signature scheme. If for any PPT adversary $\mathcal{A}$, $\mathbf{Adv}^{\text{EUF-CMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda)$ ($\mathbf{Adv}^{\text{EUF-wCMA}}_{\mathsf{SIG},\mathcal{A}}(\lambda)$, respectively) is a negligible function in λ, we say that the signature scheme SIG is EUF-CMA secure (EUF-wCMA secure, respectively).

2.2. *Chameleon Hash Functions and Generic Transformation*

Krawczyk and Rabin [36] formalized the notion of a Chameleon hash function and provided a simple construction based on the DL assumption in the standard model. A Chameleon hash function H takes two inputs m (message) and r (randomness) and outputs a hash value $H(m; r)$. It satisfies three properties, namely collision resistance, trapdoor collisions, and uniformity. The collision-resistant property states that it is infeasible (for any PPT adversary) to find two distinct messages m and m' and randomness r and r' such that $H(m; r) = H(m'; r')$. The uniformity property states that for each message m, $H(m; r)$ has the same probability distribution, where r is chosen uniformly at random. The trapdoor collisions property states that given some trapdoor information, any pair of m, r, and any additional message m', it is possible to efficiently find a randomness r' such that $H(m; r) = H(m'; r')$.

We review the Chameleon hashes based on the DL assumption.[9] Let $\mathcal{G}_{\text{ch}}$ be a group generator that takes the security parameter λ as an input and outputs a cyclic group of

[9] In [36], the Chameleon hash function is constructed over a multiplicative subgroup of a finite field. We can easily generalize it to a Chameleon hash function over any cyclic group in which the DL assumption holds.

prime order p' of 2λ bits (e.g., an elliptic curve group generator).

$\mathsf{CHSetup}(\lambda)$: $\mathcal{G}_{\mathrm{ch}}(\lambda) \to \mathbb{G}_{\mathrm{ch}}$.
Choose $g_{\mathrm{ch}} \xleftarrow{\$} \mathbb{G}_{\mathrm{ch}}$ and $\beta \xleftarrow{\$} \mathbb{Z}^*_{p'}$ and compute $h_{\mathrm{ch}} = g_{\mathrm{ch}}^{\beta}$.
Output $\{g_{\mathrm{ch}}, h_{\mathrm{ch}}\}$, as the description of $H(\cdot;\cdot)$, and trapdoor $tr = \beta$,
where $H(\cdot;\cdot) : \mathbb{Z}_{p'} \times \mathbb{Z}_{p'} \to \mathbb{G}_{\mathrm{ch}}$ is defined by $(x, r) \mapsto g_{\mathrm{ch}}^{x} h_{\mathrm{ch}}^{r}$.

$\mathsf{Trapdoor\ collision}(\beta, x, r, x')$: Output $r' = r + (x - x')/\beta$.

We can easily check that the above-mentioned scheme satisfies the three properties of Chameleon hashes. In particular, the collision resistance of the abovementioned scheme tightly comes from the DL assumption on $\mathbb{G}_{\mathrm{ch}}$; that is, if there exists an adversary finding collisions of the above Chameleon hashes with $\varepsilon_{\mathrm{ch}}$ probability in time T_{ch}, then we can construct an algorithm solving the DL problem with $\varepsilon_{\mathrm{ch}}$ probability in time T'_{ch} such that $T'_{\mathrm{ch}} \approx T_{\mathrm{ch}}$.

The generic transformation from EUF-wCMA secure signatures to EUF-CMA secure signatures has been used in many previously proposed signature schemes (e.g., [8,33,34,36,45]). Suppose that $(\mathsf{G}, \mathsf{S}, \mathsf{V})$ is an EUF-wCMA secure signature scheme for arbitrary-length messages and $\mathsf{CHSetup}$ is a generator for the description of a Chameleon hash based on the DL assumption. Then, the following scheme is an EUF-CMA secure signature scheme for fixed-length messages.[10]

$\mathsf{KeyGen}(\lambda)$: Run $\mathsf{CHSetup}(\lambda) \to (H(\cdot;\cdot), tr)$ and $\mathsf{G}(\lambda) \to (pk, sk)$, publish $PK = (pk, H)$, and then keep $SK = \{sk\}$.
$\mathsf{Sign}(PK, M, SK)$: Pick a random $r \in \mathbb{Z}_p$, compute $y = H(M; r)$, run $\mathsf{S}(pk, y, sk) \to \sigma'$, and then output the signature $\sigma = (\sigma', r)$.
$\mathsf{Verify}(PK, M, \sigma)$: Parse σ as (σ', r), compute $y = H(M; r)$, and then output $\mathsf{V}(pk, y, \sigma')$.

Lemma 1. *([34]) If* $(\mathsf{G}, \mathsf{S}, \mathsf{V})$ *is EUF-wCMA secure and* $\mathsf{CHSetup}$ *is a generator for secure Chameleon hashes, then the abovementioned scheme is an EUF-CMA secure signature scheme.*[11]

2.3. *Background on Group and Assumption*

In this paper, we use groups with bilinear pairings and the computational Diffie–Hellman (CDH) assumption in the bilinear group setting.

[10] For signing arbitrary-length messages, the signer can first apply collision-resistant hash functions.

[11] In [34], a generic transformation is given for an EUF-wCMA secure signature scheme for fixed-length messages. Note that we can easily generalize it and prove an analogous lemma for arbitrary-length messages. In fact, if the message size of an EUF-wCMA secure signature scheme is larger than or equal to the size of the representation of the Chameleon hash values, then the generic transformation derives the EUF-CMA secure signature scheme.

We define bilinear groups of prime order. The proposed signature scheme essentially uses a bilinear map.

Definition 2. We say that $\mathcal{G}$ is a bilinear group generator, if on input the security parameter λ, it outputs a tuple $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$, where p is a prime, $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_t$ are finite abelian groups of order p, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_t$ is a non-degenerate bilinear map, that is (bilinearity), for all $a, b \in \mathbb{Z}_p$ and $g \in \mathbb{G}_1, g' \in \mathbb{G}_2, e(g^a, g'^b) = e(g, g')^{ab}$ and (non-degeneracy) for generators $g \in \mathbb{G}_1$ and $g' \in \mathbb{G}_2$, $e(g, g') \neq 1$.

If $\mathbb{G}_1 = \mathbb{G}_2$, then we denote $\mathbb{G}_1$ and $\mathbb{G}_2$ by $\mathbb{G}$, and we say that e is a type-1 pairing. If $\mathbb{G}_1 \neq \mathbb{G}_2$ but there is an efficiently computable homomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$, then we say that e is a type-2 pairing. Otherwise (that is, $\mathbb{G}_1 \neq \mathbb{G}_2$ and there are no efficiently computable homomorphisms between $\mathbb{G}_1$ and $\mathbb{G}_2$), we say that e is a type-3 pairing.

We define the CDH assumption in the bilinear group setting.

Definition 3. *(Computational Diffie–Hellman Assumption)* Let $\mathcal{G}$ be a (symmetric) bilinear group generator. We say that $\mathcal{G}$ satisfies the CDH assumption if for any polynomial-time probabilistic algorithm $\mathcal{A}$ the following advantage $\mathbf{Adv}^{\mathbf{CDH}}_{\mathcal{A}}$ is negligible function in the security parameter λ.

$$\begin{aligned}&\mathbf{Adv}^{\mathbf{CDH}}_{\mathcal{G},\mathcal{A}}(\lambda)\\ &= \Pr\Big[\mathcal{A}(p, \mathbb{G}, \mathbb{G}_t, e, g, g^a, g^b) \to g^{ab}\\ &\qquad \Big| \mathcal{G}(\lambda) \to (p, \mathbb{G}, \mathbb{G}_t, e), a, b \xleftarrow{\$} \mathbb{Z}_p, g \xleftarrow{\$} \mathbb{G}\Big].\end{aligned}$$

3. Bounded CMA Secure Signatures with Sublinear Public Key

In this section, we present our first trial for almost compact signatures. The scheme proposed in this section will be extended to achieve the security in the weak CMA model in the next section.

We begin with reviewing the Hohenberger–Waters signature scheme HWSig [33], which is given in Fig. 1.[12] HWSig has compact parameters in the sense that a signature and a public key consist of constant group elements and exponents but requires a signer to maintain some state information ct, which is the number of signatures used until now. From the viewpoint of security proof, the advantage of using the state information ct is that each signature has distinct ct so that we can apply the proof technique used for selectively secure signatures (e.g., Boneh–Boyen Signatures [7]). More precisely, the idea of this approach is to enable a simulator, which reduces the security of signature scheme to the CDH problem, to restrict the adversary to attack one of the polynomially many indexes so that the simulator can guess the index associated with the target forgery

[12] In [33], Hohenberger and Waters present the signature scheme in Fig. 1 as a secondary scheme. The original CDH-based Hohenberger–Waters signature scheme has a merged form of the scheme in Fig. 1 and the Chameleon hash function. To make the effect of the Chameleon hash function and the other techniques clear, we use the scheme in Fig. 1, instead of the original scheme in [33].

$\mathsf{KeyGen}(\lambda)$: Run $\mathcal{G} \to (p, \mathbb{G}, \mathbb{G}_t, e)$ and choose $w, v, u, g_1, h, g \xleftarrow{\$} \mathbb{G}, \alpha \xleftarrow{\$} \mathbb{Z}_p$.
Output $PK = \{w, v, u, g_1, h, g, g^\alpha\}$ and $SK = \{\alpha\}$. Set $\mathsf{ct} = 0$

$\mathsf{Sign}(PK, M, SK)$: Choose $t \xleftarrow{\$} \mathbb{Z}_p$ and increase ct by 1. // ct is a state information.
Compute $\sigma_1 = (vu^M)^\alpha (w^{\lceil log_2(\mathsf{ct}) \rceil} h g_1^{\mathsf{ct}})^t$ and $\sigma_2 = g^{-t}$.
Output $\sigma = (\sigma_1, \sigma_2, \mathsf{ct})$.

$\mathsf{Verify}(PK, M, \sigma)$: Parse σ to $(\sigma_1, \sigma_2, \mathsf{ct})$.
Check whether $e(\sigma_1, g) e(\sigma_2, w^{\lceil log_2(\mathsf{ct}) \rceil} h g_1^{\mathsf{ct}}) \stackrel{?}{=} e(g^\alpha, vu^M)$.
Output 1 if the equation holds; otherwise, 0.

Fig. 1. The Hohenberger–Waters signature scheme [33].

in advance. If the simulator's guess is correct, the simulator embeds the CDH problem so that the solution of the CDH problem is extractable from any signature associated with the target index.

One naive trial to modify HWSig into a stateless signature scheme is to change the state information ct with a random tag. However, we have to be open to the possibility of there being two (or more) signatures with the same tags. To handle several signature queries using the same random tag, we apply programmable hashes [32]. In particular, we use weak programmable hash functions. For a 2λ-bit message M, we consider M as an element of $\mathbb{Z}_p$ and $(v \prod_{i=1}^{m} u_i^{M^i})$ is used instead of (vu^M) in HWSig.[13] Here, $(v \prod_{i=1}^{m} u_i^{M^i})$ is a weak programmable hash function on input M and enables the simulator to sign on at most m messages (with the same tag). Next, we use a random tag tag instead of a counter ct.[14] In addition, we remove $w^{\lceil log_2(\mathsf{ct}) \rceil}$ part from HWSig since the role of this part is ambiguous for our purpose.[15] As a result, we obtain a stateless short signature scheme with sublinear public keys, as shown in Fig. 2.

We set Q to be a polynomial in λ. The simulation strategy is as follows: the simulator guesses tag^*, the tag of the forgery (with a non-negligible $\frac{1}{Q}$ probability) and uses the technique for the selectively secure signature scheme of the Boneh–Boyen signatures. For each signature, the tag is randomly chosen so that there may exist several signatures containing the same tag as tag^* among the resulting signatures of singing queries. Under normal circumstances, the simulator cannot produce signatures with tag^* (since we use technique for a selectively secure scheme). We can resolve this by using the weak programmable hash functions. If we uniformly choose a tag from $[1, Q]$ at most q times for polynomial $Q \geq q$, there are at most $m = O(\frac{\lambda}{\log \lambda})$ same tags as the tag of the forgery with an overwhelming probability. (Here, we need to restrict $Q \geq q$ for meaningful parameter.) Therefore, the simulator can create m signatures, which have

[13] M^i is not the i-th bit of M, but M raised to the power i.

[14] Note that the idea of using random tags in signature schemes is already used in previous works [30,33].

[15] In the security proof of HWSig, $w^{\lceil log_2(\mathsf{ct}) \rceil}$ part is used to prevent forgeries associated with indices that are larger than the number of signing queries q. More precisely, the domain of indices $[1, 2^\lambda]$ is divided by two parts $[1, 2^{\lceil log_2 q \rceil}]$ and $(2^{\lceil log_2 q \rceil}, 2^\lambda]$. The former is polynomial size so that the simulator can deal with in polynomial time, but the latter is too large for the simulation to guess one index contained in the latter one. Instead, the logarithm of the index is used like $w^{\lceil log_2(\mathsf{ct}) \rceil}$. However, this separation is successful only when there is no signing query with index contained in the latter part and the *stateful* signature scheme is exactly the case. Since we will not use such the separation in the security proof for *stateless* signature scheme, $w^{\lceil log_2(\mathsf{ct}) \rceil}$ is not necessary.

$\mathsf{KeyGen}(\lambda)$: Run $\mathcal{G} \to (p, \mathbb{G}, \mathbb{G}_t, e)$ and choose $v, u_1, \ldots, u_m, g_1, h, g \xleftarrow{\$} \mathbb{G}, \alpha \xleftarrow{\$} \mathbb{Z}_p$.
Choose an integer Q.
Output $PK = \{Q, v, u_1, \ldots, u_m, g_1, h, g, g^\alpha\}$ and $SK = \{\alpha\}$.

$\mathsf{Sign}(PK, M, SK)$: Choose $t \xleftarrow{\$} \mathbb{Z}_p$ and $tag \xleftarrow{\$} [1, Q]$.
Compute $\sigma_1 = (v \prod_{i=1}^m u_i^{M^i})^\alpha (h g_1^{tag})^t$ and $\sigma_2 = g^{-t}$.
Output $\sigma = (\sigma_1, \sigma_2, tag)$.

$\mathsf{Verify}(PK, M, \sigma)$: Parse σ to $(\sigma_1, \sigma_2, tag)$.
Check whether $tag \in [1, Q]$. If not, abort and output 0.
Check whether $e(\sigma_1, g) e(\sigma_2, h g_1^{tag}) \stackrel{?}{=} e(g^\alpha, v \prod_{i=1}^m u_i^{M^i})$.
Output 1 if the equation holds; otherwise, 0.

Fig. 2. Naive approach for somewhat short public key.

the same tag as that of the forgery. For other queries related to different tags, we can apply the technique of the selectively secure signature scheme.

Unfortunately, the above-mentioned proof strategy works only against a rather weak attack called a *bounded chosen-message-attack*. Informally, the proposed signature scheme determines the number of allowable signing queries q at the parameter generation time since $Q \geq q$. This could be any polynomial in the security parameter, and hence, the security proof cannot guarantee any security against more than q signing queries. According to the applications, the number of signature generations could be bounded above at the parameter generation time, and the proposed scheme could be useful for such applications. However, usually, it is not a desirable property in practice.

We omit the proof of the bounded security of the scheme in Fig. 2 since the scheme in the next section achieves better performance and security statement and its security proof contains all ideas to prove the scheme in Fig. 2.

Remark 1. The proposed signature scheme is for fixed-length messages, but we note that we can easily modify it for arbitrary-length messages by using collision-resistant hash functions; to do so, first, we need to compute the hash value of a long message and then use it as a message for the signature scheme.

Remark 2. We used a combination of two techniques in this section for signature schemes based on the CDH assumption. This approach is similar to that used for signature schemes based on the RSA assumption and q-DH assumption [31]. Note that our main contribution is explained in the next section.

4. Almost Compact Signature Scheme

We describe our main signature scheme, which is a generalization of the previous scheme shown in Fig. 3. In contrast to the scheme discussed in the previous section, we use a tag vector $\overrightarrow{tag}$ of length k instead of an integer tag in the signatures. Note that there are two variables k and m, which are not exactly specified in the description. We determine k and m in Sect. 4.1

$\mathsf{KeyGen}(\lambda)$: Run $\mathcal{G} \to (p, \mathbb{G}, \mathbb{G}_t, e)$ and choose $v, u_1, \ldots, u_m, g_1, \ldots, g_k, h, g \xleftarrow{\$} \mathbb{G}, \alpha \xleftarrow{\$} \mathbb{Z}_p$.
Choose an integer Q. // We will specify Q later.
Output $PK = \{Q, v, u_1, \ldots, u_m, g_1, \ldots, g_k, h, g, g^\alpha\}$ and $SK = \{\alpha\}$.

$\mathsf{Sign}(PK, M, SK)$: Choose $r \xleftarrow{\$} \mathbb{Z}_p$ and $\overrightarrow{tag} \xleftarrow{\$} (tag_1, \ldots, tag_k) \xleftarrow{\$} [1, Q]^k$ (k times canonical product set).
Compute $\sigma_1 = (v \prod_{i=1}^m u_i^{M^i})^\alpha (h \prod_{i=1}^k g_i^{tag_i})^r$ and $\sigma_2 = g^{-r}$.
//Here M^i means M to the power of i (mod p)
Output $\sigma = (\sigma_1, \sigma_2, \overrightarrow{tag})$.

$\mathsf{Verify}(PK, M, \sigma)$: Parse σ to $(\sigma_1, \sigma_2, \overrightarrow{tag})$.
Check whether $\overrightarrow{tag} \in [1, Q]^k$. If not, abort and output 0.
Check whether $e(\sigma_1, g) e(\sigma_2, h \prod_{i=1}^k g_i^{tag_i}) \stackrel{?}{=} e(g^\alpha, v \prod_{i=1}^m u_i^{M^i})$.
Output 1 if the equation holds; otherwise, 0.

Fig. 3. Almost compact Diffie–Hellman based (EUF-wCMA secure) signature scheme.

The public key size is $O(m + k)$. When Q^k is smaller than p, we can consider a tag vector to be an element of $\mathbb{Z}_p$, and thus, the signature size is two group elements and one exponent. Our analysis shows that the public key is asymptotically almost compact in the sense that it could be any function in $\omega(1)$. In Sect. 5.1, we will explain the tag-free variants; by adding a constant factor in the public keys, we can remove the tag vectors from the signatures and obtain shorter signatures.

Even if the proposed signature scheme is a simple generalization of the previous scheme obtained via a combination of two techniques, the security analysis is more challenging than the construct itself. The basic proof strategy of the previous scheme (Fig. 2) is to guess the tag tag^* of the forgery and then use the programmability of the weak programmable hash function $(v \prod_{i=1}^m u_i^{M^i})$ to sign for the signature with the same tag. We cannot naively apply this proof strategy to the generalized construction due to the following reason. If k is small, then m should be large since there will be many signing queries associated with the same tag vector as that of the forgery $\overrightarrow{tag}^* \in [1, Q]^k$. Recall that the public key size is $O(m + k)$ so that large m leads large public keys. If k is large, the simulator cannot naively guess $\overrightarrow{tag}^*$, with a non-negligible probability. That is, we would fail to construct a polynomial-time reduction. We developed a proof technique to resolve this problem. Surprisingly, we can remove the restriction $Q \geq q$ so that we achieve the EUF-wCMA security for the main scheme.[16]

Basically, we use the *prefix-guessing* strategy [34] for polynomial-time reduction; just guess a prefix of the target tag vector such that it is different from that of all other tag vectors used in signing queries and it has the smallest length satisfying this property. If the guess is correct, we can apply the technique of selectively secure schemes. In fact, the prefix-guessing technique in [34] is used to guess a prefix of the target *message*, and it is non-trivial to directly apply the same approach for *random tag vectors*. All message queries and target messages are distinct in the EUF-CMA security model; therefore, it is quite easy to define and guess the prefix of the target message that is different from the message queries' prefix of the same length and has the smallest length satisfying

[16] The security proof in the conference version [4,44] still has the same restriction $Q \geq q$, and we explain the reason of such the restriction in Sect. 4.2.

this property. However, we cannot expect the random tag vectors to satisfy the same property since these vectors are chosen at random and possibly have duplications. To handle random tag vectors for the prefix guessing, we carefully separate the adversarial types according to the relation between the prefix of the target tag vector and the tag vectors used in the signing queries.

Theorem 1. *If there is an adversary breaking the EUF-wCMA security of the proposed signature scheme with ε success probability and T running time, then we can construct a CDH problem solver $\mathcal{B}$ with ε' success probability and T' running time, where*

$$\varepsilon' \geq \frac{m+1}{kqQ}\left(\varepsilon - \frac{q^{m+1}}{(m+1)!Q^{km}} - \frac{q}{p}\right) \text{ and } T \approx T'.$$

Proof. The goal of the proof is to construct a simulator $\mathcal{B}$ that solves the CDH problem with running an EUF-wCMA attacker $\mathcal{A}$ of the proposed signature scheme.

Simulation Description. The simulator $\mathcal{B}$ first takes an uniform instance of the CDH problem, $(g, g^a, g^b) \in \mathbb{G}^3$, and the bilinear group description $(\mathbb{G}, \mathbb{G}_t, e)$ over which the CDH instance is defined. For the sake of simplicity, let $A = g^a$ and $B = g^b$. $\mathcal{B}$ receives a list L of q messages $M_1, \ldots, M_q$ from $\mathcal{A}$. For $i \in [1, q]$, $\mathcal{B}$ uniformly generates random tag vectors $\overrightarrow{tag}_i$ in advance that will be used in the i-th signing query on M_i.

Adversarial Types: Next, $\mathcal{B}$ define the adversarial types according to the relation between the target tag vector and the set $\{\overrightarrow{tag}_i\}$. To this end, we need some preliminaries. We begin with defining useful notation. Let $\mathcal{T}$ and $\mathcal{T}^i$ be sets $[1, Q]$ and $[1, Q]^i$ (i times canonical product set), respectively. For $j \in [1, q]$, let $\overrightarrow{tag}_j \in \mathcal{T}^k$ be the tag vector (randomly chosen by the simulator) of the signature on the jth message (queried by the adversary). Let $\overrightarrow{tag}^* = (s_1^*, \ldots, s_k^*) \in \mathcal{T}^k$ be the tag vector of the forgery output by the adversary. For $\overrightarrow{tag} \in \mathcal{T}^k$ and $i \leq k$, let $\overrightarrow{tag}^{(i)} \in \mathcal{T}^i$ be the first i entries of $\overrightarrow{tag}$ (e.g., $\overrightarrow{tag} = (tag_1, \ldots, tag_k)$ and $\overrightarrow{tag}^{(i)} = (tag_1, \ldots, tag_i)$). For fixed $\{\overrightarrow{tag}_i\}_{i\in[1,q]}$, S_i is defined as

$$\left\{ \hat{tag} \in \mathcal{T}^i \mid \exists \text{ at least } (m+1) \text{ distinct } j_1, \ldots, j_{m+1} \in [1, q] \right.$$
$$\left. \text{such that } \hat{tag} = \overrightarrow{tag}_{j_1}^{(i)} = \ldots = \overrightarrow{tag}_{j_{m+1}}^{(i)} \right\}.$$

Let us consider an example to help the readers understand the definition of S_i.
Example. Suppose that

$$\overrightarrow{tag}_1^{(i)} = \ldots = \overrightarrow{tag}_{m+2}^{(i)} \neq \overrightarrow{tag}_j^{(i)} \text{ for } j \in [m+3, q],$$
$$\overrightarrow{tag}_1^{(i+1)} = \ldots = \overrightarrow{tag}_{m+1}^{(i+1)} \neq \overrightarrow{tag}_j^{(i+1)} \text{ for } j \in [m+2, q],$$

and $\overrightarrow{tag}_{m+3}^{(i)}, \ldots, \overrightarrow{tag}_{q}^{(i)}$ are distinct. Then,

$$\begin{cases} \overrightarrow{tag}_j^{(i)} \in S_i \text{ for } j \in [1, m+2] \\ \overrightarrow{tag}_j^{(i)} \notin S_i \text{ for } j \in [m+3, q], \end{cases}, \quad \begin{cases} \overrightarrow{tag}_j^{(i+1)} \in S_{i+1} \text{ for } j \in [1, m+1] \\ \overrightarrow{tag}_j^{(i+1)} \notin S_{i+1} \text{ for } j \in [m+2, q], \end{cases}$$

and $|S_i| = |S_{i+1}| = 1$.
We can easily see that $|S_{i+1}| \leq |S_i|$. Let n be the largest integer in $[1, k]$ such that $S_n \neq \emptyset$. If we choose m, k, and Q appropriately, we can show that $n < k$ with overwhelming probability, where the probability is taken over the choice of $\{\overrightarrow{tag}_i\}_{i\in[1,q]}$. More precisely, we can prove the following generalized birthday lemma.

Lemma 2. (Generalized Birthday Lemma) $\Pr_{\overrightarrow{tag}_1,\ldots,\overrightarrow{tag}_q \xleftarrow{\$} \mathcal{T}^k}[|S_i| \geq \ell] < \left(\frac{q^{m+1}}{(m+1)!Q^{im}}\right)^{\ell}$.[17]

If we set $i = k$ and $\ell = 1$, Lemma 2 implies that $|S_k| = 0$ (that is, $n < k$) with at least $1 - \frac{q^{m+1}}{(m+1)!Q^{km}}$ probability. We provide the proof of Lemma 2 in "Appendix."

Then, we are ready to define the adversarial type, which follows.

$$\begin{aligned} \text{Type-1} &: \overrightarrow{tag}^{*(1)} \notin S_1. \\ \text{Type-2} &: \overrightarrow{tag}^{*(1)} \in S_1, \text{ and } \overrightarrow{tag}^{*(2)} \notin S_2. \\ &\vdots \\ \text{Type-}i &: \overrightarrow{tag}^{*(i-1)} \in S_{i-1}, \text{ and } \overrightarrow{tag}^{*(i)} \notin S_i. \\ &\vdots \\ \text{Type-}k &: \overrightarrow{tag}^{*(k-1)} \in S_{k-1}, \text{ and } \overrightarrow{tag}^{*(k)} \notin S_k. \\ \text{Type-}(k+1) &: \overrightarrow{tag}^{*(k)} \in S_k. \end{aligned}$$

If $|S_k| \geq 1$, then the simulator aborts. For this case, we say that an event E_1 occurs. Otherwise (that is, $|S_k| < 1$), we know that there is an integer $n < k$ such that every adversary should be only one of $n + 1$ types. Note that we do not require the condition on the public parameter $Q \geq q$.

Prefix Guessing: Then, the simulator guesses a prefix of the target tag vector as follows: it guesses adversary type, say type-i, with at least $\frac{1}{k}$. Next step of the simulator is to guess $\overrightarrow{tag}^{*(i)}$. To this end, the simulator guesses $\overrightarrow{tag}^{*(i-1)}$ (if $i > 1$ only) and tag_i^*, respectively, where $\overrightarrow{tag}^{*(i-1)}$ is a prefix vector consisting of the first $(i-1)$ entries of the target tag vector $\overrightarrow{tag}^*$ and tag_i^* is the i-th entry of $\overrightarrow{tag}^*$. For $\overrightarrow{tag}^{*(i-1)}$, the simulator randomly chooses a tag vector in the set $\{\overrightarrow{tag}_j\}_{j\in[1,q]}$ and then sets its $(i-1)$-th prefix as the simulator's guess $\overrightarrow{tag}^{*(i-1)}$; that is, the simulator guesses $\overrightarrow{tag}^{*(i-1)}$ as a randomly chosen vector from $\{\overrightarrow{tag}_j^{(i-1)}\}_{j\in[1,q]}$. For tag_i^*, the simulator uniformly guesses it from

[17] In the conference version [4], we fully used the advantage of Lemma 2. However, in this version, the special case ($i = k$ and $\ell = 1$) of Lemma 2 is sufficient for our new security proof.

its domain $[1, Q]$. We will argue that the simulator's guess of $\overrightarrow{tag}^{*(i)}$ is correct with at least $\frac{m+1}{qQ}$ later. In the following description, let us assume that $\mathcal{B}$'s guesses for the adversarial type i and $\overrightarrow{tag}^{*(i)}$ are correct, so that the variable i denotes adversarial type.

Let us briefly explain the remaining part of the simulation. The important property of the prefix guessing $\mathcal{B}$ did above is that there are at most m tag vectors $\overrightarrow{tag}_j$'s such that their i-th prefixes are equal to the i-th prefix of the target tag vector, that is, $\overrightarrow{tag}_j^{(i)} = \overrightarrow{tag}^{*(i)}$ since for the type-i adversary, $\overrightarrow{tag}^{*(i)} \notin S_i$. (For the type-$(n+1)$ adversary, $S_{n+1} = \emptyset$.) By using this property, we can simulate public key and all signatures on M_i's; for signatures with tag vectors having prefixes different from $\overrightarrow{tag}^{*(i)}$, we will use the technique for selectively secure signature scheme (e.g., Boneh–Boyen signatures [9]), and for signatures with tag vectors (at most m) having the same prefix as $\overrightarrow{tag}^{*(i)}$, we will use the (weak) programmable hashes [32] in simulation.

KeyGen: We know that the i-prefix of the target tag vector, $\overrightarrow{tag}^{*(i)}$ is not contained in S_i, and so there exist at most m distinct tag vectors whose i-prefix is equal to $\overrightarrow{tag}^{*(i)}$ among q tag vectors for signing queries. Let I be the set of indices for such tag vectors. Then, $|I| \leq m$. We first define a polynomial $f(X)$ having as roots messages M_j for $j \in I$; that is, $f(X) := \prod_{j \in I}(X - M_j)$. If I is an empty set, we just define $f(X) = 1$. We can rewrite $f(X)$ by $\sum_{j=0}^{m} x_j X^j$ for some coefficients $x_0, \ldots, x_m \in \mathbb{Z}_p$. Note that $x_j = 0$ for $j > |I|$. Next, $\mathcal{B}$ uniformly chooses integers $y_0, \ldots, y_m, z_1, \ldots, z_i, w_1, \ldots, w_k \xleftarrow{\$} \mathbb{Z}_p$. We define another polynomial $Y(X) := \sum_{j=0}^{m} y_j X^j$ and vectors $\overrightarrow{z} = (z_1, \ldots, z_i)$ and $\overrightarrow{w} = (w_1, \ldots, w_k)$. Lastly, $\mathcal{B}$ generates a public key $PK = \{v, u_1, \ldots, u_m, g_1, \ldots, g_k, h, g, g^\alpha\}$ as in Fig. 4. Then, the unknown $b = \log_g B$ is implicitly set as the corresponding secret key.

Sign: $\mathcal{B}$ generates signatures on $M_1, \ldots, M_q$ as follows. For the j-th signing query, $\mathcal{B}$ first checks whether $j \in I$ and then $\mathcal{B}$ separately behaves as follows:
If $j \in [1, q] \setminus I$, then $\mathcal{B}$ checks whether the equality $\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z} \rangle = 0$ holds. (Recall that $\mathcal{B}$ already chose all tag vectors $\overrightarrow{tag}_1, \ldots, \overrightarrow{tag}_q$ in advance before the *prefix-guessing* phase.) If the equality holds, then $\mathcal{B}$ aborts the simulation and outputs a random element. For this case, we say that an event E_2 occurs. Otherwise, $\mathcal{B}$ chooses a random integer $r' \xleftarrow{\$} \mathbb{Z}_p$ and computes a signature as follows.

$$\sigma_{j1} = B^{Y(M_j) - (w_0 + \langle \overrightarrow{tag}_j, \overrightarrow{w} \rangle) \frac{Y(M_j)}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z} \rangle}} \cdot (A^{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z} \rangle} g^{w_0 + \langle \overrightarrow{tag}_j, \overrightarrow{w} \rangle})^{r'} \quad (1)$$

$$\text{and } \sigma_{j2} = B^{\frac{f(M_j)}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z} \rangle}} \cdot g^{-r'}. \quad (2)$$

$v = A^{x_0} g^{y_0}$,	$u_j = A^{x_j} g^{y_j}$ for $j \in [1, m]$,	$h = A^{-\langle \overrightarrow{tag}^{*(i)}, \overrightarrow{z} \rangle} g^{\overrightarrow{w}_0}$
$g_j = \begin{cases} A^{z_j} g^{w_j} & \text{for } j \in [1, i] \\ g^{w_j} & \text{for } j \in [i+1, k] \end{cases}$,	$g = g$,	$g^\alpha = B$

Fig. 4. Public key simulation.

If $j \in I$, $\mathcal{B}$ chooses $r \xleftarrow{\$} \mathbb{Z}_p$ and computes a signature as follows.

$$\sigma_{j1} = B^{Y(M_j)}(g^{w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle})^r \text{ and } \sigma_{j2} = g^{-r}. \tag{3}$$

Lastly, $\mathcal{B}$ defines the j-th signature σ_j on M_j by $(\sigma_{j1}, \sigma_{j2}, \overrightarrow{tag}_j)$.
Response: $\mathcal{B}$ sends $\mathcal{A}$ the public key $PK = (v, u_1, \ldots, u_m, h, g_1, \ldots, g_k, g, g^\alpha)$ along with signatures $\sigma_1, \ldots, \sigma_q$.

Extraction from Forgery At the end of interaction, $\mathcal{B}$ receives a message M^* along with a forgery $\sigma^* = (\sigma_1^*, \sigma_2^*, \overrightarrow{tag}^*)$ on M^* from $\mathcal{A}$ such that $M^* \notin L$. If $\mathsf{Verify}(PK, M^*, \sigma^*) = 0$, $\mathcal{B}$ aborts. Otherwise, $f(M^*) = \sum_{i=0}^m x_i (M^*)^i \neq 0$ since $M^* \notin L$ and all f's roots are contained in L. Finally, $\mathcal{B}$ outputs

$$(\sigma_1^* \cdot B^{-Y(M^*)} \cdot (\sigma_2^*)^{w_0+\langle \overrightarrow{tag}^*, \overrightarrow{w}\rangle})^{\frac{1}{f(M^*)}} \tag{4}$$

as the solution of the CDH instance (g, g^a, g^b).

Analysis of the Reduction Algorithm $\mathcal{B}$

Distribution of simulation. We show that the simulated transcript (public key and signing queries) between $\mathcal{A}$ and $\mathcal{B}$ is indistinguishable from the real transcript on the condition that the simulator does not abort. Since $y_0, \ldots, y_m$, and $w_0, \ldots, w_k$ are uniformly chosen from $\mathbb{Z}_p$ and the CDH instance is also uniformly generated, the public key simulated by $\mathcal{B}$ is identical to those of the output of the KeyGen algorithm. Next, we consider the distribution of simulated signatures for signing queries. All the tag vectors are uniformly and independently chosen in advance at the beginning of the simulation, and we can assume that each randomly chosen tag vector is assigned to the corresponding message at the generating time. Therefore, that the distribution of the tag vectors is identical to the real transcript, except for the case of the simulation fail events E_1 and E_2. Although the simulator used the tag vectors before the signing phase (that is, during the prefix-guessing phase), it does not matter; the tag vectors are used only for guessing the prefix of the target tag vector, which is embedded into public key and signatures, and we can show that, regardless of the correctness of the prefix guessing, the distribution of signatures is identical to the real one. Let us focus on the other parts in signatures except for tag vectors. For $j \in [1, q] \setminus I$, we argue that the randomness r used in the j-th signature query is distributed as if $r = -\frac{f(M_j)b}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle} + r'$;

$$\begin{aligned}
\sigma_{j1} &= B^{\frac{Y(M_j)-(w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle)\frac{Y(M_j)}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle}}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle}} \cdot \left(A^{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle} g^{w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle}\right)^{r'} \\
&= B^{Y(M_j)} \cdot \left(g^{ab}\right)^{f(M_j)} \cdot \left(A^{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle} g^{w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle}\right)^{\frac{-f(M_j)b}{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle}} \\
&\quad \cdot \left(A^{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle} g^{w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle}\right)^{r'} \\
&= \left(\prod_{t=0}^m (A^{x_t} g^{y_t})^{M_j^t}\right)^b \cdot \left(A^{\langle (\overrightarrow{tag}_j^{(i)} - \overrightarrow{tag}^{*(i)}), \overrightarrow{z}\rangle z_t} g^{w_0+\langle \overrightarrow{tag}_j, \overrightarrow{w}\rangle}\right)^r
\end{aligned}$$

$$= \left(\prod_{t=0}^{m}(A^{x_t}g^{y_t})^{M_j^t}\right)^b \cdot \left(A^{-\langle\overrightarrow{tag}^{*(i)},\overrightarrow{z}\rangle}g^{w_0}\cdot\prod_{t=1}^{i}(A^{z_t}g^{w_t})^{tag_{jt}}\cdot\prod_{t=i+1}^{k}(g^{w_t})^{tag_{jt}}\right)^r$$

$$= \left(v\prod_{t=1}^{m}u_t^{M_j^t}\right)^b \cdot \left(h\prod_{t=1}^{k}g_t^{tag_{jt}}\right)^r$$

and $\sigma_{j2} = B^{\frac{f(M_j)}{\langle(\overrightarrow{tag}_j^{(i)}-\overrightarrow{tag}^{*(i)}),\overrightarrow{z}\rangle}}\cdot g^{-r'} = g^{-r}$.

We can see that r is uniformly distributed according to r' since r' is uniformly and independently chosen from $\mathbb{Z}_p$. Consequently, we showed that the simulated distribution of σ_{j1} and σ_{j2} is identical to that of the output of Sign algorithm.

For $j \in I$, we argue that the simulated signature is distributed with the randomness r;

$$\begin{aligned}\sigma_{j1} &= B^{Y(M_j)}\cdot\left(g^{w_0+\langle\overrightarrow{tag}_j,\overrightarrow{w}\rangle}\right)^r\\ &= B^{Y(M_j)}\cdot\left(g^{ab}\right)^{f(M_j)}\cdot\left(A^{\langle(\overrightarrow{tag}_j^{(i)}-\overrightarrow{tag}^{*(i)}),\overrightarrow{z}\rangle}g^{w_0+\langle\overrightarrow{tag}_j,\overrightarrow{w}\rangle}\right)^r\\ &= \left(\prod_{t=0}^{m}(A^{x_t}g^{y_t})^{M_j^t}\right)^b\cdot\left(A^{-\langle\overrightarrow{tag}^{*(i)},\overrightarrow{z}\rangle}g^{w_0}\prod_{t=1}^{i}(A^{z_t}g^{w_t})^{tag_{jt}}\prod_{i=i+1}^{k}(g^{w_t})^{tag_{jt}}\right)^r\\ &= \left(v\prod_{t=1}^{m}u_t^{M_j^t}\right)^b\left(h\prod_{t=1}^{k}g_t^{tag_{jt}}\right)^r.\end{aligned}$$

In the second equality, we used the fact that $f(M_j) = 0$ and $\overrightarrow{tag}_j^{(i)} = \overrightarrow{tag}^{*(i)}$ for $j \in I$. We know that $\sigma_{j2} = g^{-r}$. Since r is an uniformly chosen integer, the simulated distribution of σ_{j1} and σ_{j2} is identical to that of the output of Sign algorithm.

We note that the generation of public key and signatures is based on the simulator's guessing $\overrightarrow{tag}^{*(i)}$ for the prefix of the target vector, and the above argument is still valid even when the simulator's guessing is wrong. Therefore, the simulated transcript is identical to the real transcript when neither E_1 nor E_2 occurs.

CDH Solution Extraction. We show that the CDH solution $\mathcal{B}$ outputs is valid on the condition that two events E_1 and E_2 do not occur and the simulator's guessing $\overrightarrow{tag}^{*(i)}$ is correct. If $\mathcal{A}$ outputs a valid forgery $(\sigma_1^*, \sigma_2^*, \overrightarrow{tag}^*)$, then it satisfies the verification equation so that it is of the form $\sigma_1^* = (v\prod_{t=1}^{m}u_t^{M^{*t}})^{\alpha}(h\prod_{t=1}^{k}g_t^{tag_t^*})^r$ and $\sigma_2^* = g^{-r}$ for some r. From the simulator's public key setting, we know that σ_1^* is equal to $(g^{ab})^{f(M^*)}(g^b)^{Y(M^*)}(g^r)^{w_0+\langle\overrightarrow{tag}^*,\overrightarrow{w}\rangle}$ so that $\mathcal{B}$ outputs

$$\left(\sigma_1^*\cdot B^{-Y(M^*)}\cdot(\sigma_2^*)^{w_0+\langle\overrightarrow{tag}^*,\overrightarrow{w}\rangle}\right)^{\frac{1}{f(M^*)}} = g^{ab}.$$

Simulation Halt. From Lemma 2, we obtain that the probability $\Pr[E_1] = \Pr_{\overrightarrow{tag}_1,\ldots,\overrightarrow{tag}_q\xleftarrow{\$}\mathcal{T}^k}[|S_k| \geq 1]$ is less than $\frac{q^{m+1}}{(m+1)!Q^{km}}$. For $\Pr[E_2]$, we obtain that $\Pr[E_2] =$

$\Pr_{z_1,\ldots,z_i}\left[\exists j \in [1,q] \setminus I \text{ such that } \Sigma_{t=1}^{i}(tag_{jt} - tag_t^*)z_t = 0\right] < \frac{q}{p}$ from the union bound. We argue that $\Pr[E_2]$ is independent from the adversarial behaviors; all z_t's are completely hidden from the adversarial view since those are masked by w_t's in public key.

Simulator's guess. We show that the simulator's guess is correct with at least $\frac{m+1}{kqQ}$ probability. The simulator's prefix guessing consists of three steps. First, it guesses the adversarial types with $\frac{1}{k}$ probability; since this guess is completely independent from all other process of the simulator and is also hidden from the adversarial view, it is correct with $\frac{1}{k}$ probability. Next, we consider the conditional probability that the simulator's guess of $\overrightarrow{tag}^{*(i-1)}$ is correct once its guess of the adversarial type is correct as i. If $i > 1$, then $\overrightarrow{tag}^{*(i-1)} \in S_{i-1}$ so that there are at least $m+1$ tag vectors in $\{\overrightarrow{tag}_j\}_{j\in[1,q]}$ such that $\overrightarrow{tag}_j^{(i-1)} = \overrightarrow{tag}^{*(i-1)}$. Hence, the probability that the simulator chooses such a tag vector $\overrightarrow{tag}_j$ satisfying the equality $\overrightarrow{tag}_j^{(i-1)} = \overrightarrow{tag}^{*(i-1)}$ is more than $\frac{m+1}{q}$. Finally, the simulator can guess tag_i^* with $\frac{1}{Q}$. Since all probabilities are independent, the overall probability to correctly guess the prefix of the target tag vector is at least $\frac{m+1}{kqQ}$.

Success Probability. For the success probability $\Pr[S_{\mathcal{A}}] = \varepsilon$, we can bound $\mathcal{B}$'s success probability $\Pr[S_{\mathcal{B}}]$ as follows.

$$\begin{aligned}\Pr[S_{\mathcal{B}}] &= \frac{m+1}{kqQ}\Pr[S_{\mathcal{A}} \wedge \neg E_1 \wedge \neg E_2] \\ &\geq \frac{m+1}{kqQ}(\Pr[S_{\mathcal{A}}] - \Pr[E_1 \vee E_2]) \\ &> \frac{m+1}{kqQ}\left(\varepsilon - \frac{q^{m+1}}{(m+1)!Q^{km}} - \frac{q}{p}\right)\end{aligned}$$

□

4.1. *Parameter Selection*

If we choose $Q = poly(\lambda)$ and $m = k = \omega(1)$, where $poly$ is an arbitrary polynomial, then we can show that the simulator's success probability

$$\frac{m+1}{kqQ}\left(\varepsilon - \frac{q^{m+1}}{(m+1)!Q^{km}} - \frac{q}{p}\right)$$

is non-negligible, where ε is non-negligible and q denotes the maximum number of allowed signing queries, which is a polynomial in λ. For example, if $Q = \lambda$, $m = k = \log\log\lambda$, then $\frac{q^{m+1}}{Q^{km}} = \frac{q^{1+\log\log\lambda}}{\lambda^{(\log\log\lambda)^2}}$ is clearly a negligible function in λ, under the condition that q is a polynomial in λ. Therefore, we obtain the asymptotic result $\Pr[S_{\mathcal{B}}] \sim \frac{1}{q\lambda}\varepsilon$, with the parameter selection $m = k = \omega(1)$ and $Q = \lambda$.

Although, for any polynomial q, $\frac{q^{m+1}}{(m+1)!Q^{km}}$ can be negligible with $m = k = \omega(1)$ and $Q = poly(\lambda)$, it is not exponentially small. In practice, one may want to consider

Table 2. Concrete PK size of EUF-wCMA secure signature scheme in Fig. 3.

λ	m	k	PK size
80	5	18	$27\tau_{\mathbb{G}}$
128	6	25	$35\tau_{\mathbb{G}}$
192	7	33	$44\tau_{\mathbb{G}}$
256	8	40	$52\tau_{\mathbb{G}}$

$\tau_{\mathbb{G}}$ is the size of group element

sub-exponential or exponential-time adversaries. To this end, we can choose $m = k = \Theta(\sqrt{\frac{\lambda}{\log\lambda}})$ and $Q = \lambda$ so that $\frac{q^{m+1}}{(m+1)!Q^{km}}$ is exponentially small in λ.[18] In this case, we still have the sublinear public key $\Theta(\sqrt{\frac{\lambda}{\log\lambda}})$ while preserving the reduction loss $O(\lambda q)$.

In particular, for a concrete security parameter, we can yield reasonably short public keys. More precisely, for concrete security parameter $\lambda \in \{80, 128, 192, 256\}$, we can minimize $m+k$ with constraint $\frac{q^{m+1}}{(m+1)!\lambda^{km}} \leq \frac{1}{2^{\lambda}}$ and $q \leq 2^{\lambda}$. We give example parameters satisfying these conditions in Table 2. Finally, after transforming via Lemma 1, we have an EUF-CMA secure signature scheme with a public key size of 29 group elements (including two additional group elements for a Chameleon hash function) for 80-bit security, which is much shorter than the public key size (164 group elements) of the Waters signature scheme for the same security parameter; note that the reduction loss of both schemes is the same $O(\lambda q)$. Further, the transformed EUF-CMA secure signatures are still short (two group elements and two exponents).

4.2. *Note on the Security Proof in [4,44]*

Contrary to the security analysis in this paper, the security proof in [44] and the first part of [4] prevents only the adversaries with bounded signing queries. We explain the proof strategy in [4,44] and the reason why it inherently requires the restriction of the adversarial model.

Overall proof strategy is almost the same as that in this paper. The main difference is the prefix-guessing phase for $\overrightarrow{tag}^{*(i)}$ satisfying $\overrightarrow{tag}^{*(i-1)} \in S_{i-1}$ and $\overrightarrow{tag}^{*(i)} \notin S_i$ when assuming the type-i adversary. In [4,44], for type-i adversary, the simulator naively guesses $\overrightarrow{tag}^{*(i)}$ by uniformly choosing $\overrightarrow{tag}^{*(i-1)}$ from S_{i-1} and tag_i^* from $\mathcal{T}$, where $\overrightarrow{tag}^{*(i)} = (\overrightarrow{tag}^{*(i-1)}, tag_i^*) \in \mathcal{T}^i$. Then, the simulator's guess is correct at least the probability $\frac{1}{|S_{i-1}|\cdot|\mathcal{T}|}$. To bound the size of S_{i-1}, we use Lemma 2, so that we obtain $\Pr[|S_{i-1}| > \lambda] < \left(\frac{q^{m+1}}{(m+1)!Q^{(i-1)m}}\right)^{\lambda}$. To make the right-hand side of the inequality negligible for arbitrary adversarial type i, it is necessary to set $Q > q$. If this probability becomes negligible by setting parameters Q and m appropriately and so we ignore it, then the simulator succeeds in guessing $\overrightarrow{tag}^{*(i)}$ with high probability $\frac{1}{|S_{i-1}|\cdot|\mathcal{T}|} \geq \frac{1}{\lambda Q}$. The remaining part of the proof is essentially the same as given in this paper. If $Q \leq q$, one cannot bound the size of S_{i-1} sufficiently small so that this proof approach cannot

[18] The other term $\frac{q}{p}$ is already exponentially small in λ.

guarantee non-negligible success probability in the simulator's guess. Therefore, this approach necessarily requires $Q > q$ and loses $\left(\frac{q^{m+1}}{(m+1)!Q^{(i-1)m}}\right)^{\lambda}$ probability, compared to the proof given in this paper; that is, the maximum allowable signing queries q should be bounded by the size of tag Q at the parameter generation time.

In a nutshell, our prefix guessing for $\overrightarrow{tag}^{*(i-1)} \in S_{i-1}$ is done by randomly choosing from $\{\overrightarrow{tag}_j^{(i-1)}\}_{j\in[1,q]}$ but that in [4,44] is done by randomly choosing from S_{i-1}. Therefore, in our security proof, we do not need to bound $|S_{i-1}|$ so that we can remove the undesirable requirement $Q > q$.

5. Extensions

In this section, we provide two extensions of the main construction. We give an EUF-wCMA secure variant of the main scheme that has shorter signatures (two group elements). Next, an instantiation using asymmetric pairings (type-2 pairings or type-3 pairings) instead of symmetric pairings (type-1 pairings) is considered.

5.1. *Tag Compression Using Pseudorandom Functions*

In this subsection, we introduce a trick for tag compression using (non-adaptive) pseudorandom functions (PRF). Note that similar techniques are used in the RSA-based signatures [30,33,34,48] to compress random prime numbers used in each signature. If we use this trick, we can reduce the signature size of EUF-wCMA secure scheme to two group elements by augmenting signing/verification costs and adding constant factor in public key size.

Each signature has a tag vector that is uniformly chosen from its domain. Thus, a signer can use PRF mapping from messages to tag vectors and publishes the PRF the signer used along with its key. Even though the signer publishes the PRF key (in the weak security model), we can use the fact that the distribution of tag vectors is indistinguishable from the uniform distribution. In some application, short signatures are important even though public key size and signing/verifying costs increase. Then, the signature scheme with tag compression technique is appropriate in such applications.

Let $PRF : \{0,1\}^* \to \{0,1\}^{k\lceil \log Q\rceil}$ be a pseudorandom function family. The signer randomly chooses a PRF key K, uses $PRF_K(M)$ as the tag vector associated with the signature of M, and publishes the description of PRF and the key K of PRF as a part of PK. Then, we can remove tag vectors from signatures since everyone who knows a message and PK can compute the corresponding tag vector. Hence, the resulting EUF-wCMA secure signature scheme has shorter signatures (two group elements).

5.2. *Instantiation Using Asymmetric Pairings*

Although we described our construction using type-1 pairings in Sect. 3, we can easily modify our construction to be instantiated using type-2 pairings or type-3 parings as in Fig. 5. The scheme using type-1 parings and its security proof does not use pairing's symmetry property. Our main idea to achieve short public key is prefix guessing

$\mathsf{KeyGen}(\lambda)$: Run $\mathcal{G} \to (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_t, e)$.
Choose $\mathsf{v}, \mathsf{u}_1, \ldots, \mathsf{u}_m, \mathsf{g}_1, \ldots, \mathsf{g}_k, \mathsf{h}, \alpha \xleftarrow{\$} \mathbb{Z}_p, g \xleftarrow{\$} \mathbb{G}_2, g' \xleftarrow{\$} \mathbb{G}_1$, and pick a PRF key K at random.
Compute $v = g^{\mathsf{v}}, u_i = g^{\mathsf{u}_i}, g_i = g^{\mathsf{g}_i}, h = g^{\mathsf{h}}$.
Output $PK = \{v, u_1, \ldots, u_m, g_1, \ldots, g_k, h, g \in \mathbb{G}_2, g'^{\alpha} \in \mathbb{G}_1, K\}$ and
$SK = \{\mathsf{v}, \mathsf{u}_1, \ldots, \mathsf{u}_m, \mathsf{g}_1, \ldots, \mathsf{g}_k, \mathsf{h}, \alpha \in \mathbb{Z}_p, g' \in \mathbb{G}_1\}$,
where $PRF : \{0,1\}^* \to \{0,1\}^{k\lceil \log Q \rceil}$ be a pseudorandom function family.

$\mathsf{Sign}(PK, M, SK)$: Uniformly choose $t \xleftarrow{\$} \mathbb{Z}_p$
Compute $PRF_K(M) = (tag_1, \ldots, tag_k)$,
$\sigma_1 = g'^{(\mathsf{v}+\sum_{i=1}^{m} \mathsf{u}_i M^i)\alpha + (\mathsf{h}+\sum_{i=1}^{k} \mathsf{g}_i tag_i)t}$ and $\sigma_2 = g'^{-t} \in \mathbb{G}_1$.
Output $\sigma = (\sigma_1, \sigma_2)$.

$\mathsf{Verify}(PK, M, \sigma)$: Parse σ to (σ_1, σ_2).
Compute $PRF_K(M) = (tag_1, \ldots, tag_k)$.
Check whether $e(\sigma_1, g) e(\sigma_2, h \prod_{i=1}^{k} g_i^{tag_i}) \stackrel{?}{=} e(g'^{\alpha}, v \prod_{i=1}^{m} u_i^{M^i})$.
Output 1 if the above equation holds; otherwise, 0.

Fig. 5. Instantiation using asymmetric pairings and PRF.

via dividing adversarial types according to tag vectors in the security proof, and this technique is independent of pairing's types. Therefore, we can prove the security of the instantiation using asymmetric pairings by following the same proof strategy used for the scheme with type-1 pairings. For type-2 pairings, the security can be reduced to the co-CDHP: given an efficiently computable homomorphic map $\phi : \mathbb{G}_2 \to \mathbb{G}_1$ and $g', g^b \in \mathbb{G}_1, g, g^a \in \mathbb{G}_2$ such that $g' = \phi(g)$, compute g'^{ab}. For type-3 pairings, the security of the proposed signature scheme can be reduced to the co-CDHP*: given $g', g'^a, g'^b \in \mathbb{G}_1, g, g^a \in \mathbb{G}_2$, compute g'^{ab}. Note that the security of the Waters signature scheme using asymmetric pairings is also based on the same problems [1,15].

Although the overall security proof for type-2 and type-3 pairings is essentially the same as that for type-1 pairing, let us briefly sketch to enhance confidence. For both the co-CDHP and the co-CDHP*, the simulator begins with $g', g'^b \in \mathbb{G}_1, g, g^a \in \mathbb{G}_2$. Let $A = g^a$ and $B = g'^B$. First, we observe all public keys are elements in $\mathbb{G}_2$ except $g'^{\alpha} \in \mathbb{G}_1$ and a signature consists of elements in $\mathbb{G}_1$. The first prefix-guessing phase is the same since we do not require pairings there yet. From Fig. 4, we can see that the simulator can generate all public keys using $g, A \in \mathbb{G}_2$ and $B \in \mathbb{G}_1$ since all exponents are chosen by the simulator. Similarly, from Eqs. (1) and (3), we can check that the simulator can answer to all signature queries by using $g', B \in \mathbb{G}_1$. (Again, the simulator knows all exponents used there.) At the end of interaction, the simulator can extract g'^{ab}, which is a solution of either co-CDHP or co-CDHP* according to the given problem instance, by using $\sigma_1^*, \sigma_2^*, B \in \mathbb{G}_1$ as in the Eq. 4. The analysis for the distribution of the transcripts and the success probability of the simulation will be exactly the same as that for type-1 pairing in Sect. 4.

Acknowledgements

The author would like to thank Dennis Hofheinz for insightful discussions about the asymptotic security of the proposed signature scheme, Shota Yamada for helpful comments on the early version of this paper, and anonymous reviewers of the Journal of

Cryptology for invaluable comments and constructive suggestions. This work was supported by 2014 Research Fund of Myongji University.

Appendix: Proof of Lemma 2

In this subsection, we prove Lemma 2. Let F be the set of all functions from $[1, q]$ to $\mathcal{T}^i$. For $\overrightarrow{y} \in \mathcal{T}^i$ and $f \in F$, let $|f^{-1}(\overrightarrow{y})|$ be the number of the distinct pre-images of $\overrightarrow{y}$. Let T_f be the set of all $\overrightarrow{y} \in Im(f)$ such that $|f^{-1}(\overrightarrow{y})| \geq m+1$, where $Im(f)$ means the set of all images of f. Then, we can consider $\Pr_{\overrightarrow{tag}_1,\ldots,\overrightarrow{tag}_q \xleftarrow{\$} \mathcal{T}^k}[|S_i| \geq j]$ as

$$\Pr_{f \xleftarrow{\$} F}[|T_f| \geq j].$$

To compute $\Pr_{f \xleftarrow{\$} F}[|T_f| \geq j]$, we count all functions f such that $|T_f| \geq j$ and then divide the result by $|\mathcal{T}^i|^q$ (the number of all elements in F). In fact, we count the number of f such that $|T_f| \geq j$, allowing duplications, so that we compute the upper bound of $\Pr_{f \xleftarrow{\$} F}[|T_f| \geq j]$. To define an f, we choose j distinct subsets $A_1, \ldots, A_j$ of size $m+1$ from $[1, q]$ and j distinct vectors $\overrightarrow{y}_1, \ldots, \overrightarrow{y}_j$ from $\mathcal{T}^i$, and then set $f(a) = \overrightarrow{y}_t$ for all $a \in A_t$ and $t \in [1, j]$. For other integers $a \in [1, q] \setminus (A_1 \cup \ldots \cup A_j)$, we arbitrarily define $f(a)$. This way of defining a function covers all f such that $|T_f| \geq j$. We count all f that are defined as above. Then, the number of such f is bounded by

$$\left(\prod_{t=0}^{j-1} \binom{q - t(m+1)}{m+1} \cdot (|\mathcal{T}^i| - t)\right) \cdot (|\mathcal{T}^i|)^{(q-j(m+1))},$$

where the notation $\binom{\cdot}{\cdot}$ denotes the binomial coefficient.

Therefore, we can obtain the desired result as follows:

$$\begin{aligned}
\Pr_{\overrightarrow{s}_1,\ldots,\overrightarrow{s}_q \xleftarrow{\$} S^k}[|S_i| \geq j] &= \Pr_{f \xleftarrow{\$} F}[|T_f| \geq j] \\
&< \frac{\left(\prod_{t=0}^{j-1} \binom{q-t(m+1)}{m+1} \cdot (Q^i - t)\right) \cdot (Q^i)^{(q-j(m+1))}}{|\mathcal{T}^i|^q} \\
&< \frac{\left(\frac{q^{m+1}}{(m+1)!}\right)^j Q^{ij+i(q-j(m+1))}}{Q^{iq}} \\
&= \left(\frac{q^{m+1}}{(m+1)!Q^{im}}\right)^j.
\end{aligned}$$

Remark 3. The result in Lemma 2 is similar as the lemma given in [30] called "generalized birthday bound." Note that Lemma 2 is more general than "generalized birthday bound"; e.g., if we set $i = 1$ and $j = 1$, then the result in Lemma 2 provides a more tighter upper bound than "generalized birthday bound" given in [30].

References

[1] M. Bellare, T. Ristenpart, Simulation without the artificial abort: simplified proof and improved concrete security for Waters' IBE scheme, in *EUROCRYPT 2009*, vol. 5479 of *LNCS* (Springer, 2009), pp. 407–424

[2] M. Bellare, P. Rogaway, The exact security of digital signatures: how to sign with RSA and Rabin, in *EUROCRYPT 1996*, vol. 1070 of *LNCS* (Springer, 1996), pp. 399–416

[3] O. Blazy, E. Kiltz, J. Pan, (hierarchical) identity-based encryption from affine message authentication, in *CRYPTO 2014, Part I*, vol. 8616 of *LNCS* (Springer, 2014), pp 408–425

[4] F. Böhl, D. Hofheinz, T. Jager, J. Koch, J. H. Seo, C. Striecks, Practical signatures from standard assumptions, in *EUROCRYPT2013*, vol. 7881 of *LNCS* (Springer, 2013)

[5] F. Böhl, D. Hofheinz, T. Jager, J. Koch, C. Striecks, Confined guessing: new signatures from standard assumptions, in *Cryptology ePrint Archive* (2013). http://eprint.iacr.org/2013/171

[6] F. Böhl, D. Hofheinz, T. Jager, J. Koch, C. Striecks, Confined guessing: new signatures from standard assumptions. *J. Cryptol.***28**, 176–208 (2015)

[7] D. Boneh, X. Boyen, Efficient selective-id identity based encryption without random oracles, in *EUROCRYPT 2004*, vol. 3027 of *LNCS* (Springer, 2004), pp. 223–238

[8] D. Boneh, X. Boyen, Short signatures without random oracles, in *EUROCRYPT 2004*, vol. 3027 of *LNCS* (Springer, 2004), pp. 382–400

[9] D. Boneh, X. Boyen, Efficient selective identity-based encryption without random oracles. *J. Cryptol.***24**(4), 659–693 (2011)

[10] D. Boneh, M. Franklin, Identity-based encryption from the Weil pairing, in *CRYPTO 2001*, vol. 2139 of *LNCS* (Springer-Verlag, 2001), pp. 19–23

[11] D. Boneh, B. Lynn, H. Shacham, Short signatures from the Weil pairing. *J. Cryptol.***17**, 297–319 (2004)

[12] D. Brown, R. Gallant, The static Diffie–Hellman problem. Available in http://eprint.iacr.org/2004/306

[13] J. Camenisch, A. Lysyanskaya, Signature schemes and anonymous credentials from bilinear maps, in *CRYPTO 2004*, vol. 3152 of *LNCS* (Springer, 2004), pp. 56–72

[14] R. Canetti, O. Goldreich, S. Halevi, The random oracle methodology, revisited, in *ACM STOC 2003*, pp. 209–218

[15] S. Chatterjee, D. Hankerson, E. Knapp, A. Menezes, Comparing two pairing-based aggregate signature schemes, in *Designs, Codes and Cryptography*, vol. 55 (Springer, 2010), pp. 141–167

[16] J. Chen, H.W. Lim, S. Ling, H. Wang, H. Wee, Shorter IBE and signatures via asymmetric pairings, in *Pairing 2012*, vol. 7708 of *LNCS* (Springer, 2012), pp. 122–140

[17] J. Chen, H. Wee, Fully, (almost) tightly secure IBE and dual system groups, in *CRYPTO 2013*, vol. 8043 of *LNCS* (Springer, 2013), pp. 435–460

[18] J.H. Cheon. Discrete logarithm problems with auxiliary inputs. *J. Cryptol.***23**, 457–476 (2010)

[19] R. Cramer, I.Damgård, New generation of secure and practical rsa-based signatures, in *CRYPTO 1996*, vol. 1109 of *LNCS* (Springer, 1996), pp. 173–185

[20] R. Cramer, V. Shoup, Signature schemes based on the strong rsa assumption, in *ACM CCS 1999* (ACM Press, 1999), pp. 46–51

[21] Y. Dodis, R. Oliveira, K. Pietrzak, On the generic insecurity of the full domain hash, in *CRYPTO 2005*, vol. 3621 of *LNCS* (Springer, 2005), pp. 449–466

[22] C. Dwork, M. Naor, An efficient existentially unforgeable signature scheme and its applications, in *CRYPTO 1994*, vol. 839 of *LNCS* (Springer, 1994), pp. 234–246

[23] T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in G.R. Blakely, D. Chaum, editors, *CRYPTO 1984*, vol. 196 of *LNCS* (Springer-Verlag, 1984), pp. 10–18

[24] M. Fischlin, The Cramer-Shoup strong-RSA signature scheme revisited, in *PKC 2003*, vol. 2567 of *LNCS* (Springer, 2003), pp. 116–129

[25] R. Gennaro, S. Halevi, T. Rabin, Secure hash-and-sign signatures without the random oracle, in *EUROCRYPT 1999*, vol. 1592 of *LNCS* (Springer, 1999), pp. 123–139

[26] C. Gentry, C. Peikert, V. Vaikuntanathan, Trapdoors for hard lattices and new cryptographic constructions, in *ACM STOC 2008* 2008, pp. 197–206

[27] E.-J. Goh, S. Jarecki, J. Katz, N. Wang Efficient signature schemes with tight reductions to the Diffie–Hellman problems. *J. Cryptol.***20**, 493–514 (2007)

[28] O. Goldreich, Two remarks concerning the Goldwasser–Micali–Rivest signature scheme, in *CRYPTO 1986*, vol. 263 of *LNCS* (Springer, 1987), pp. 104–110
[29] S. Goldwasser, S. Micali, R. L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. Comput.***17**, 281–308 (1988)
[30] D. Hofheinz, T. Jager, E. Kiltz, Short signatures from weaker assumptions. in *ASIACRYPT 2011*, vol. 7073 of *LNCS* (Springer, 2011), pp. 647–666
[31] D. Hofheinz, T. Jager, E. Knapp, Waters signatures with optimal security reduction, in *PKC 2012*, vol. 7293 of *LNCS* (Springer, 2012), pp. 66–83
[32] D. Hofheinz, E. Kiltz. Programmable hash functions and their applications. *J. Cryptol.***25**, 484–527 (2012)
[33] S. Hohenberger, B. Waters, Realizing hash-and-sign signatures under standard assumptions, in *EUROCRYPT 2009*, vol. 5479 of *LNCS* (Springer, 2009), pp. 333–350
[34] S. Hohenberger, B. Waters, Short and stateless signatures from the rsa assumption, in *CRYPTO 2009*, vol. 5677 of *LNCS* (Springer, 2009), pp. 654–670
[35] C.S. Jutla, A. Roy, Shorter quasi-adaptive NIZK proofs for linear subspaces, in *ASIACRYPT 2013, Part I*, vol. 8269 of *LNCS* (Springer, 2013), pp. 1–20
[36] H. Krawczyk, T. Rabin, Chameleon signatures, in *NDSS 2000* (The Internet Society, 2000)
[37] G. Leurent, P.Q. Nguyen, How risky is the random-oracle model?, in S. Halevi, editor, *CRYPTO 2009*, vol. 5677 of *LNCS* (Springer, 2009), pp. 445–464
[38] A. Lysyanskaya, R. Rivest, A. Sahai, S. Wolf, Pseudonym systems, in *SAC 1999*, vol. 1758 of *LNCS* (Springer, 1999), pp. 184–199
[39] D. Naccache. Secure and practical identity-based encryption. *IET Inf. Secur.***1**(2), 59–64 (2007)
[40] M. Naor, On cryptographic assumptions and challenges, in *CRYPTO 2003*, vol. 2729 of *LNCS* (Springer, 2003), pp. 96–109
[41] T. Okamoto, Provably secure and practical identification schemes and corresponding signature schemes, in *CRYPTO 1992*, vol. 740 of *LNCS* (Springer, 1992), pp. 31–53
[42] T. Okamoto, Efficient blind and partially blind signatures without random oracles, in *TCC 2006*, vol. 3876 of *LNCS* (Springer, 2006), pp. 80–99
[43] C.P. Schnorr. Efficient signature generation for smart cards. *J. Cryptol.***4**, 239–252 (1991)
[44] J.H. Seo, Short signature from Diffie–Hellman: Realizing short public key, in *Cryptology ePrint Archive* 2012. http://eprint.iacr.org/2012/480.
[45] A. Shamir, Y. Tauman, Improved online/offline signature schemes, in *CRYPTO 2001*, vol. 2139 of *LNCS* (Springer, 2001), pp. 355–367
[46] B. Waters, Efficient identity-based encryption without random oracles, in *EUROCRYPT 2005*, vol. 3494 of *LNCS* (Springer, 2005), pp. 114–127
[47] B. Waters, Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions, in S. Halevi, editor, *CRYPTO 2009*, vol. 5677 of *LNCS* (Springer, 2009), pp. 619–636
[48] S. Yamada, G. Hanaoka, N, Kunihiro. Space efficient signature schemes from the RSA assumption, in *PKC 2012*, vol. 7293 of *LNCS* (Springer, 2012), pp. 102–119