Secure Obfuscation for Encrypted Signatures

Satoshi Hada

IBM Research - Tokyo, satoshih@jp.ibm.com

Abstract. Obfuscation is one of the most intriguing open problems in cryptography and only a few positive results are known. In TCC'07, Hohenberger et al. proposed an obfuscator for a re-encryption functionality, which takes a ciphertext for a message encrypted under Alice's public key and transforms it into a ciphertext for the same message under Bob's public key [24]. It is the first complicated cryptographic functionality that can be securely obfuscated, but obfuscators for such cryptographic functionalities are still elusive. In this paper, we consider obfuscation for encrypted signature (ES) functionalities, which generate a signature on a given message under Alice's secret signing key and encrypt the signature under Bob's public encryption key. We propose a special ES functionality, which is the sequential composition of Waters's signature scheme [33] and the linear encryption scheme proposed by Boneh, Boyen, and Shacham [5], and construct a secure obfuscator for it. We follow the security argument by Hohenberger et al. to prove that our proposed obfuscator satisfies a virtual black-box property (VBP), which guarantees that the security of the signature scheme is preserved even when adversaries are given an obfuscated program. Our security argument is in the standard model.

Keywords: Obfuscation, encrypted signatures, signcryption, bilinear map

1 Introduction

An obfuscator is a tool to convert a program into a new *unintelligible* program while preserving the functionality. Several formal definitions have been proposed so far [22, 3, 27, 34, 20, 23, 24, 21, 9, 12]. Informally, obfuscators should satisfy the following two requirements: (1) functionality: the obfuscated program has the same functionality as the original one and (2) virtual black-box property (VBP): whatever one can efficiently compute given the obfuscated program can be computed given black-box access to the functionality. The functionality requirement is the syntactic requirement while the VBP is the security requirement to capture the unintelligibility of obfuscated programs.

As discussed in [3], obfuscators, if they exist, would have a wide variety of cryptographic applications including software protection, fully homomorphic encryption, removing random oracles, and transforming private-key encryption schemes into public-key encryption schemes. Unfortunately, the impossibility of generic obfuscation have been shown in [3, 20] even under very weak VBP definitions, which require that any *predicate* that can be efficiently computed from an

obfuscated program can also be efficiently computed given black-box access to the functionality. Specifically, they showed the existence of (contrived) functionalities for which no obfuscator can satisfy the weak VBPs. However, the negative results do not rule out the possibility that there exists an obfuscator for a *specific* functionality. Indeed, some positive results are known for point functions [8, 11, 27, 34, 20, 16, 23, 9, 12]. In spite of these positive results, obfuscators for traditional cryptographic functionalities have remained elusive.

In TCC'07, Hohenberger et al. proposed an obfuscator for a re-encryption functionality [24]. It is the first complicated cryptographic functionality that can be securely obfuscated. A re-encryption functionality for Alice and Bob takes a ciphertext for a message encrypted under Alice's public key and transforms it into a ciphertext for the same message under Bob's public key. The naive program contains both Alice's secret key and Bob's public key, and it simply decrypts the input ciphertext and encrypts the plain message. Clearly, this program reveals Alice's secret key to any party executing the program. If Alice can securely obfuscate the program, the VBP ensures that any party learns no more from the obfuscated program than it does from black-box access to the functionality. In particular, the obfuscated program does not reveal Alice's secret key and cannot be used for eavesdropping. Hohenberger et al. constructed a special encryption scheme and a secure obfuscator for the re-encryption functionality. Their security argument is based on a new VBP definition suitable for cryptographic functionalities. It ensures that the security of cryptographic functionalities can be preserved even when adversaries are given obfuscated programs (See the discussion in [24]). They showed that the security of their proposed encryption scheme is preserved even when adversaries are given an obfuscated re-encryption program.

From both the theoretical and practical perspectives, it is important to investigate the possibility of secure obfuscation for more cryptographic functionalities. In this paper, we consider obfuscation for encrypted signature (ES) functionalities. An ES functionality for Alice and Bob generates a signature on a given message under Alice's secret signing key and then encrypts the signature under Bob's public encryption key. As in re-encryption, the naive program contains Alice's secret key and Bob's public key, and reveals Alice's secret key. If Alice can securely obfuscate the program, the VBP ensures that any party executing the obfuscated program cannot forge Alice's signature. We propose a special pair of signature and encryption schemes and construct a secure obfuscator for the ES functionality. Also, we follow the security argument by Hohenberger et al. in [24] to show that the security of signature scheme is preserved even when adversaries are given an obfuscated ES program.

We believe that there are many useful applications of our proposed obfuscation. We informally describe an application to secure Webmail services. ES should not be confused with Signature-then-Encryption or Sign-then-Encrypt (StE). The StE functionality signs a given message and then encrypts both the message and signature. StE is the most widely-used approach to constructing a

signcryption scheme [2, 35][1]. On the other hand, the ES functionality does not encrypt the message itself and we can not necessarily use it as a signcryption scheme. However, as shown in Appendix A, we believe that one can use an ES functionality as a building block to construct a signcryption functionality and that an obfuscator for the ES functionality can be used to obfuscate the signcryption functionality. Potential target applications include Webmail services such as Yahoo! Mail and Google's Gmail where users use e-mail services via Web browsers. If users want to signcrypt their messages and their Web browsers do not have the required capability then their Webmail providers need to signcrypt their messages on behalf of them. This means that users need to securely delegate their signing capability to the Webmail providers. The combination of the proposed obfuscation for an ES functionality presented in this paper and the obfuscation for the signcryption scheme constructed in Appendix A would provide a solution. For example, even if a malicious operator inside Webmail providers is given an obfuscated signcryption program for Alice-to-Bob, he/she can neither forge Alice's signatures nor perform the signcryption operation for Alice-to-Carol. However, we must be careful. The obfuscation does not prevent the malicious operator from performing the signcryption for Alice-to-Bob. Also, if the malicious operator has access to Bob's secret decryption key, he/she can forge Alice's signatures (In the case of our proposed obfuscations, what is worse is that he/she can extract Alice's secret signing key from the obfuscated program). The formal security argument for the signcryption and obfuscation outlined in Appendix A is outside the scope of this proceedings version.

1.1 Basic Idea

Our obfuscation is based on the following basic idea: *We construct a special pair of signature and encryption schemes such that generating a signature on a message and then encrypting the signature is functionally equivalent to encrypting the signing key and then generating a signature on the message under the encrypted signing key. The former process is the ES functionality. In the latter process, "encrypting the signing key" can be viewed as an obfuscation of the ES functionality and "generating a signature on the message under the encrypted signing key" corresponds to executing the obfuscated program.*

We informally describe how to construct such a special pair using the BLS signature scheme proposed by Boneh et al. [7]. Let $(q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$ be a parameter for a bilinear map, where both $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of prime order q, $\mathbf{e}$ is an efficient bilinear mapping from $\mathbb{G} \times \mathbb{G}$ to $\mathbb{G}_T$, and g is a generator of $\mathbb{G}$. A public verification and secret signing key pair is (v, s) such that $v = g^s$, where s is a random number in Z_q^*. Given a message m, the signature σ is calculated as $H(m)^s$, where $H : \{0,1\}^* \rightarrow \mathbb{G}$ is a hash function. We can verify the signature by checking the equality $\mathbf{e}(g, \sigma) = \mathbf{e}(H(m), v)$. If the computational Diffie-Hellman problem is hard, the scheme is secure in the random oracle model [4], where H is

[1] Following [2], we use the term "signcryption" for any scheme achieving both privacy and authenticity in the public key setting.

modeled as a random oracle. Also, we use a secure key encapsulation mechanism (KEM) to encrypt the signature value $\sigma = H(m)^s$. Let $\mathsf{KEM.Enc}(pk)$ be the encryption algorithm of the secure KEM. Given a public encryption key pk, it generates a pair of a random key r and its ciphertext c. Given $\mathsf{KEM.Enc}$, we define an encryption algorithm Enc, which takes as input a plaintext $p(= H(m)^s) \in \mathbb{G}$ and a public key pk, generates $(r, c) \leftarrow \mathsf{KEM.Enc}(pk)$, and outputs (c, p^r) as the ciphertext. The key and message spaces of $\mathsf{KEM.Enc}$ and Enc are $\mathbb{Z}_q^*$ and $\mathbb{G}$, respectively. The decryption is straightforward (You can decrypt c to recover r and then p). Then we consider the ES functionality defined as the sequential composition of the BLS signing algorithm and the encryption algorithm Enc. That is, given a message m, it computes the signature $H(m)^s$, generates $(r, c) \leftarrow \mathsf{KEM.Enc}(pk)$, and outputs $(c, H(m)^{sr})$. The naive program implementing the ES functionality contains (s, pk) and reveals s. Our goal is to obfuscate it.

Approach 1. Given the naive program, we extract (s, pk) and encrypt s using $\mathsf{KEM.Enc}$. Specifically, we generate $(r, c) \leftarrow \mathsf{KEM.Enc}(pk)$ and compute $sr \bmod q$, where (c, sr) is an encryption of the secret signing key s. It reveals no information on s since $\mathsf{KEM.Enc}$ is secure. However, using it, we can still compute an encryption of the valid signature of a given message m. That is, we can construct an obfuscated program $C_{c,sr}$ containing (c, sr), which takes as input m and outputs $(c, H(m)^{sr})$. Note that the output is an encryption of the valid signature $H(m)^s$ by Enc. The problem here is that $C_{c,sr}$ does not preserve the *probabilistic* ES functionality since it is *deterministic*. If Enc is rerandomizable with pk[2], we can fix the problem simply by rerandomizing $(c, H(m)^{sr})$. That is, we can construct a new obfuscated program $C_{c,sr,pk}$ containing (c, sr, pk), which takes as input m, computes $(c, H(m)^{sr})$, rerandomizes it using pk, and outputs the rerandomized ciphertext. The contained information (c, sr, pk) reveals no information on s because it is a ciphertext. It is not difficult to see that the obfuscation satisfies a VBP under the assumption that $\mathsf{KEM.Enc}$ is secure. In other words, the VBP simply reduces to the security of $\mathsf{KEM.Enc}$.

Approach 1′ (A special case of Approach 1). We describe a sufficient condition under which Enc is rerandomizable with pk. If $\mathsf{KEM.Enc}$ satisfies a *scalar homomorphic* property (and is rerandomizable with pk), then Enc is rerandomizable with pk. By the scalar homomorphic property, we mean that, given a KEM ciphertext c, we can compute (r', c') such that r' is a new random key and c' is a ciphertext of $rr' \bmod q$. We denote the operation by $(r', c') \leftarrow \mathsf{multiply}_{pk}(c)$. In this approach, a modified obfuscated program $C'_{c,sr,pk}$ computes $(c, H(m)^{sr})$, generates $(r', c') \leftarrow \mathsf{multiply}_{pk}(c)$, and outputs $(c', H(m)^{srr'})$, which is a rerandomization of $(c, H(m)^{sr})$. Bob can decrypt c' to recover rr' and then $H(m)^s$. We are done if $C'_{c,sr,pk}$ preserves the probabilistic functionality. However, we still

[2] We mean that anybody having the public key can convert a ciphertext of a message into a different ciphertext that is distributed identically to a fresh encryption of the same message.

have a potential problem. The distribution of c' may be different from the original distribution produced by $\mathsf{KEM.Enc}$. If $\mathsf{KEM.Enc}$ is *rerandomizable* with pk, we can fix it simply by rerandomizing c'. That is, a modified program $C''_{c,sr,pk}$ computes $(c', H(m)^{srr'})$, rerandomizes c' as c'', and outputs $(c'', H(m)^{srr'})$. For example, we can use the Paillier encryption scheme as $\mathsf{KEM.Enc}$ [31]. However, since its message (key) space is $\mathbb{Z}_n$ such that n is the product of two large primes, we need to define the bilinear group for the BLS scheme as having order n.

Approach 2. When Enc cannot be rerandomizable and we can not take Approaches 1 and 1′, we can consider a new ES functionality. The new ES functionality is the sequential composition of the BLS signing algorithm and a new encryption algorithm Enc'. Enc' takes as input a plaintext $p(= H(m)^s) \in \mathbb{G}$ and a public key pk, runs $\mathsf{KEM.Enc}$ twice ($(r_1, c_1) \leftarrow \mathsf{KEM.Enc}(pk)$, $(r_2, c_2) \leftarrow \mathsf{KEM.Enc}(pk)$), and outputs $(c_1, c_2, p^{r_1 r_2})$. Clearly, the use of two random keys (r_1, r_2) is redundant, but we can obfuscate the naive program as follows: Given the naive program, we extract (s, pk), generate $(r_1, c_1) \leftarrow \mathsf{KEM.Enc}(pk)$, and compute $sr_1 \bmod q$. Then, we construct an obfuscated program $C_{c_1,sr_1,pk}$ containing (c_1, sr_1, pk), which takes as input m, generates $(r_2, c_2) \leftarrow \mathsf{KEM.Enc}(pk)$, and outputs $(c_1, c_2, H(m)^{sr_1 r_2})$. The contained information (c_1, sr_1, pk) is the same as in the previous approaches and reveals no information on s. Note that the $C_{c_1,sr_1,pk}$ does not preserve the *probabilistic* ES functionality since the value of c_1 is always the same. However, if $\mathsf{KEM.Enc}$ is rerandomizable with pk, then it is easy to fix the problem by rerandomizing c_1. In this approach, we can use any rerandomizable encryption scheme as $\mathsf{KEM.Enc}$.

Comparison. Let us briefly compare the above three approaches. Approaches 1 and 2 require that Enc and $\mathsf{KEM.Enc}$ are rerandomizable, respectively. Approach 1′ is a special case of Approach 1 and requires that a scalar homomorphic property (and rerandomizability) of $\mathsf{KEM.Enc}$, which is a strong requirement. Note that we may be able to take Approach 1 without using the scalar homomorphic property required by Approach 1′ (although we don't have a concrete example). Approach 2 requires a redundancy of ciphertexts. Therefore, Approach 1 seems to be the best approach.

1.2 Our Contributions

In this paper, we will use the pair of Waters's signature scheme [33] and the linear encryption scheme proposed by Boneh, Boyen, and Shacham [5] to take Approach 1′ and propose a secure obfuscator for the ES functionality. Following the security definition and argument by Hohenberger et al. in [24], we present a security analysis of our proposed obfuscation. Waters's signature scheme is more complicated than the BLS scheme, but it is provably secure in the standard model. All security arguments in this paper are in the standard model.

Our contributions are summarized as follows: In Section 3, we propose two security definitions of digital signature schemes in the context of ES. One requires that no adversary can existentially forge a signature even if it is given

black-box access to the ES functionality. The other requires the same even if it is given an *obfuscated* program for the ES functionality. We expect that the former/weaker definition implies the latter/stronger definition if the obfuscator satisfies a VBP. In Section 4, we propose a natural generalization of the VBP definition proposed by Hohenberger et al. in [24] so that we can show that the weaker existential unforgeability implies the stronger one. As stated in [24], their proposed VBP provides a meaningful security for cryptographic schemes if they satisfy a special property called *distinguishable attack property*. Unfortunately, digital signature schemes do not have this property in the context of ES. This is the reason why we need to introduce the generalized VBP definition. In Section 5, we propose a special ES functionality, which is the sequential composition of Waters's signature scheme and the linear encryption scheme, and construct an obfuscator for it. We prove that the obfuscator is secure under the generalized VBP definition and that Waters's signature scheme satisfies the stronger existential unforgeability with the obfuscator.

1.3 Related Works

Some related works are already mentioned in the previous sections. In particular, our work is inspired by the secure obfuscation for re-encryption in [24]. Both re-encryption and ES functionalities output a ciphertext and this common property enables us to simulate *real* obfuscated programs by randomly generating *junk* programs to prove the VBPs.

Our proposed obfuscation can be viewed as *public-key obfuscation* for signing functionalities [30, 1]. A generic construction of public key obfuscations with a fully homomorphic encryption scheme is discussed in [18].

There are some different definitional approaches than VBPs to capture the unintelligibility of obfuscation, e.g., indistinguishability of obfuscation [3], best-possible obfuscation [21], and non-malleable obfuscation [12].

2 Preliminaries

Given a positive integer n, we denote by $[n]$ the set $\{1, 2, \cdots, n\}$. We say that a function $\nu(\cdot) : \mathbb{N} \to \mathbb{R}^+$ is negligible in n if for every polynomial $p(\cdot)$ and all sufficiently large n's, it holds that $\nu(n) < 1/p(n)$. Given a probability distribution S, we denote by $x \leftarrow S$ the operation of selecting an element according to S. If A is a probabilistic machine then $A(x_1, x_2, \ldots, x_k)$ denotes the output distribution of A on inputs $(x_1, x_2, \ldots, x_k)$. Let $\Pr[x \leftarrow S_1; x_2 \leftarrow S_2; \ldots; x_k \leftarrow S_k : E]$ denote the probability of the event E after the processes $x_1 \leftarrow S_1, x_2 \leftarrow S_2, \ldots, x_k \leftarrow S_k$ are performed in order. PPT stands for "probabilistic polynomial time". All PPT machines in this paper run in probabilistic polynomial-time in the security parameter denoted by n. Also, some PPT machines (e.g., representing adversaries) are allowed to take non-uniform auxiliary input of polynomial length in n, which is denoted by z.

2.1 Circuit Obfuscators

A class of circuits is of the form $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$, where $\mathcal{C}_n$ is a set of polynomial-size circuits with input length $l_{in}(n)$ and output length $l_{out}(n)$, where $l_{in}(n)$ and $l_{out}(n)$ are polynomials. It has an associated PPT generation algorithm which takes as input 1^n and generates a random circuit C from $\mathcal{C}_n$. In this paper, it corresponds to the random selection of information such as cryptographic keys on security parameter 1^n. We denote the generation process by $C \leftarrow \mathcal{C}_n$. When a circuit is used as an input or an output argument of an algorithm, we assume that an encoding of circuits is used implicitly (e.g., obfuscators take as input a circuit and output a circuit). The results of this paper are independent of any particular encoding method.

Let $C(x, r)$ be a *probabilistic* circuit which takes the regular input x and the random input r. Given a regular input x, we can view $C(x, \cdot)$ as a sampling algorithm for the distribution obtained by evaluating $C(x, r)$ on random coins r. Given two probabilistic circuits (C_1, C_2) whose regular inputs are of the same length, we denote by $(C_1(x), C_2(x))$ the two distributions produced by $C_1(x, \cdot)$ and $C_2(x, \cdot)$ and by $\mathsf{StaDiff}(C_1(x), C_2(x))$ the statistical difference between them, i.e., $\mathsf{StaDiff}(C_1(x), C_2(x)) = \frac{1}{2}\sum_{y\in\{0,1\}^{l_{out}(n)}} |\Pr[o \leftarrow C_1(x) : o = y] - \Pr[o \leftarrow C_2(x) : o = y]|$.

When we say that a machine M has black-box access to a probabilistic circuit C, we have two different meanings: *oracle access* and *sampling access*. Oracle access is such that M is allowed to set both regular and random inputs. We denote it by M^C. Sampling access is such that M is allowed to set only the regular input, but not the random input. That is, when M makes an oracle query x, M obtains a uniform and independent sample from the distribution produced by $C(x, \cdot)$. We denote it by $M^{\ll C \gg}$.

An obfuscator for a class of circuits $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ is a PPT machine which takes as input a circuit $C \in \mathcal{C}_n$ and outputs an unintelligible circuit C' which preserve the functionality. In this paper, we require that the functionality should be perfectly preserved.

Definition 1. *A PPT machine* Obf *is a* circuit obfuscator *for a class of probabilistic circuits* $\mathcal{C} = \{\mathcal{C}_n\}_{n\in\mathbb{N}}$ *if, for every probabilistic circuit* $C \in \mathcal{C}_n$*, the following holds:* $\Pr[C' \leftarrow \mathsf{Obf}(C) : \forall x, \mathsf{StaDiff}(C(x), C'(x)) = 0] = 1$.

Remark 1. We can relax the functionality requirement by allowing a negligible statistical difference and a negligible error probability as in [23, 24]. In this paper, we use this stronger definition because our proposed obfuscator can satisfy it.

Definition 1 says nothing about the security requirement and we will formulate it based on VBPs in Section 4.

2.2 Public-Key Encryption and Digital Signatures

We review the security notions of public-key encryption (PKE) and digital signature (DS) schemes (Our definitions are based on [19]). Let Setup be an algorithm

which, on security parameter 1^n, generates a parameter to be used commonly by multiple users in a pair of PKE and DS schemes.

A PKE scheme consists of three algorithms $(\mathsf{EKG}, \mathsf{E}, \mathsf{D})$. The key generation algorithm EKG is a probabilistic algorithm which takes as input a common parameter p and returns a public-secret key pair (pk, sk). The encryption algorithm E is a probabilistic algorithm which takes a common parameter p, a public key pk, and a plaintext $m \in MS(p, pk)$ to return a ciphertext c, where $MS(p, pk)$ is the message space defined by (p, pk). The encryption process is denoted by $c \leftarrow \mathsf{E}(p, pk, m)$. The decryption algorithm D is a deterministic algorithm which takes a common parameter p, a secret key sk, and a ciphertext c to return the plaintext m, and the decryption process is denoted by $m = \mathsf{D}(p, sk, c)$. When the given ciphertext is invalid, the decryption algorithm produces a special symbol $\perp$ to indicate that the ciphertext was invalid. It is required that, for every key information (p, pk, sk) and every message $m \in MS(p, pk)$, the decryption always succeeds, i.e., $\Pr[c \leftarrow \mathsf{E}(p, pk, m) : \mathsf{D}(p, sk, c) = m] = 1$. The following is the standard indistinguishability requirement against chosen plaintext attacks (CPAs). The definition is for a single message, but implies the indistinguishability requirement for polynomially multiple messages [19].

Definition 2 (Indistinguishability of Encryptions against CPAs). *A PKE scheme $(\mathsf{EKG}, \mathsf{E}, \mathsf{D})$ satisfies the indistinguishability if the following condition holds: For every PPT machine pair (A_1, A_2) (adversary), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$2 \cdot \Pr\left[\begin{array}{l} p \leftarrow \mathsf{Setup}(1^n); (pk, sk) \leftarrow \mathsf{EKG}(p); \\ (m_1, m_2, h) \leftarrow A_1(p, pk, z); b \leftarrow \{0,1\}; c \leftarrow \mathsf{E}(p, pk, m_b); \\ d \leftarrow A_2(p, pk, (m_1, m_2, h), c, z) : \\ b = d \end{array}\right] - 1 < \frac{1}{p(n)},$$

where we assume that A_1 produces a valid message pair m_1 and $m_2 \in MS(p, pk)$.

A DS scheme consists of three algorithms $(\mathsf{SKG}, \mathsf{S}, \mathsf{V})$. The key generation algorithm SKG is a probabilistic algorithm which takes as input a common parameter p and returns a public-secret key pair (pk, sk). The signing algorithm S is a probabilistic algorithm which takes a common parameter p, a secret key sk, and a plaintext $m \in MS(p, pk)$ to return a signature σ, where $MS(p, pk)$ is the message space defined by (p, pk). The signing process is denoted by $\sigma \leftarrow \mathsf{S}(p, sk, m)$. The verification algorithm V is a deterministic algorithm which takes a common parameter p, a public key pk, a message m, and a signature σ to return Accept if σ is a valid signature of m, and the verification process is denoted by $d = \mathsf{V}(p, pk, m, \sigma)$. It is required that, for every key information (p, pk, sk) and every message $m \in MS(p, pk)$, the verification of valid signatures always succeeds, i.e., $\Pr[\sigma \leftarrow \mathsf{S}(p, sk, m) : \mathsf{V}(p, pk, m, \sigma) = \mathsf{Accept}] = 1$. The following is the standard existential unforgeability (EU) requirement against chosen-message attacks (CMAs).

Definition 3 (Existential Unforgeability against CMAs). *A DS scheme $(\mathsf{SKG}, \mathsf{S}, \mathsf{V})$ is existentially unforgeable if the following condition holds: For every*

PPT oracle machine A (adversary), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,

$$\Pr\left[\begin{array}{l} p \leftarrow \mathsf{Setup}(1^n); (pk, sk) \leftarrow \mathsf{SKG}(p); \\ (m, \sigma, Q) \leftarrow A^{\ll \mathsf{S}_{p,sk} \gg}(p, pk, z) : \\ \mathsf{V}(p, pk, m, \sigma) = \mathsf{Accept} \text{ and } m \notin Q \end{array}\right] < \frac{1}{p(n)},$$

where $\mathsf{S}_{p,sk}$ is the signing oracle (circuit) and Q is the set of messages queried by A adaptively.

3 Security of Digital Signatures in the Context of ES

In this section, we re-define the EU requirement on DS schemes in the context of ES. Let $(\mathsf{EKG}, \mathsf{E}, \mathsf{D})$ and $(\mathsf{SKG}, \mathsf{S}, \mathsf{V})$ be a pair of PKE and DS schemes. We consider the ES functionality $F_{ES} = \{F_n\}_{n \in \mathbb{N}}$ for the two schemes. Given a common parameter p, a secret signing key sk, and a public encryption key pk_e generated with the security parameter 1^n, the ES functionality $F_{p,sk,pk_e} \in F_n$ is defined as follows:

1. When F_{p,sk,pk_e} is run on a message m, it generates a signature on m under sk ($\sigma \leftarrow \mathsf{S}(p, sk, m)$), encrypts σ under pk_e ($c \leftarrow \mathsf{E}(p, pk_e, \sigma)$), and outputs c.
2. When F_{p,sk,pk_e} is run on the special input keys, it outputs (p, pk, pk_e), where pk is the public verification key corresponding to sk.

Also, we define a corresponding class of circuits $\mathcal{C}_{ES} = \{\mathcal{C}_n\}_{n \in \mathbb{N}}$ which implements F_{ES}. $\mathcal{C}_n$ is a set of circuits C_{p,sk,pk_e} implementing F_{p,sk,pk_e}. The associated generation algorithm takes as input 1^n, generates $p \leftarrow \mathsf{Setup}(1^n)$, $(sk, pk) \leftarrow \mathsf{SKG}(p)$ and $(sk_e, pk_e) \leftarrow \mathsf{EKG}(p)$, and outputs C_{p,sk,pk_e}.

The following is the EU requirement re-defined in the context of ES. The difference from Definition 3 is that A is given the public encryption key pk_e. However, it is still equivalent to Definition 3.

Definition 4 (EU w.r.t. ES Functionality). *Let $(\mathsf{EKG}, \mathsf{E}, \mathsf{D})$ and $(\mathsf{SKG}, \mathsf{S}, \mathsf{V})$ be a pair of PKE and DS schemes. The DS scheme is existentially unforgeable w.r.t. the ES functionality if the following condition holds: For every PPT machine A (adversary), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\Pr\left[\begin{array}{l} p \leftarrow \mathsf{Setup}(1^n); (pk, sk) \leftarrow \mathsf{SKG}(p); (pk_e, sk_e) \leftarrow \mathsf{EKG}(p); \\ (m, \sigma, Q) \leftarrow A^{\ll \mathsf{S}_{p,sk} \gg}(p, pk, pk_e, z) : \\ \mathsf{V}(pk, m, \sigma) = \mathsf{Accept} \text{ and } m \notin Q \end{array}\right] < \frac{1}{p(n)}.$$

Note that the adversary A implicitly has sampling access to F_{p,sk,pk_e} because it has sampling access to the signing oracle $\mathsf{S}_{p,sk}$ and takes as input the public encryption key (p, pk_e). In this sense, Definition 4 requires that the signature scheme is still existentially unforgeable *even when A is given sampling access to* F_{p,sk,pk_e}.

Next, we consider a stronger EU, which requires that the signature scheme is still existentially unforgeable *even when A is given an obfuscated circuit for F_{p,sk,pk_e}*. The following is the strengthened definition and the difference from Definition 4 is that A is given an obfuscated circuit for F_{p,sk,pk_e}.

Definition 5 (EU w.r.t. ES Obfuscator). *Let $(\mathsf{EKG}, \mathsf{E}, \mathsf{D})$ and $(\mathsf{SKG}, \mathsf{S}, \mathsf{V})$ be a pair of PKE and DS schemes. Also, let Obf be a circuit obfuscator for $\mathcal{C}_{ES}$. The DS scheme is existentially unforgeable w.r.t. Obf if the following condition holds: For every PPT machine A (adversary), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\Pr\left[\begin{array}{l} p \leftarrow \mathsf{Setup}(1^n); (pk, sk) \leftarrow \mathsf{SKG}(p); (pk_e, sk_e) \leftarrow \mathsf{EKG}(p); \\ C' \leftarrow \mathsf{Obf}(C_{p,sk,pk_e}); \\ (m, \sigma, Q) \leftarrow A^{\ll \mathsf{S}_{p,sk} \gg}(p, pk, pk_e, C', z) : \\ \mathsf{V}(pk, m, \sigma) = \mathsf{Accept} \text{ and } m \notin Q \end{array}\right] < \frac{1}{p(n)}.$$

We expect that if Obf satisfies a strong VBP, then the EU w.r.t. the ES functionality implies the (stronger) EU w.r.t. Obf. The question is what VBP Obf should satisfy for the implication to hold. We will answer it in the next section.

Remark 2. We can re-define the indistinguishability of encryptions in the context of ES in a similar way. However, we omit it because the main purpose of obfuscators is to hide Alice's secret signing key but not Bob's secret decryption key. Note that the indistinguishability is preserved even when the distinguisher D in Definition 2 is given the naive program for ES that reveals the signing key.

4 Virtual Black-Box Properties

In this section, we review average-case VBP (ACVBP) proposed by Hohenberger et al. in [24], under which the VBP of the re-encryption obfuscator is proved. As stated in [24], their ACVBP provides a meaningful security for cryptographic schemes if they satisfy a special property called *distinguishable attack property*. Unfortunately, DS schemes do not have this property in the context of ES. That is, even if Obf satisfies the ACVBP under their ACVBP definition, the EU w.r.t. the ES functionality does not necessarily imply the stronger EU w.r.t. Obf. Therefore, we propose a natural generalization of their ACVBP definition under which we can claim that, if Obf satisfies the (stronger) ACVBP, then the implication holds.

First of all, we review the definition of ACVBP proposed in [24].

Definition 6 (ACVBP [24]). *A circuit obfuscator Obf for $\mathcal{C}$ satisfies the* ACVBP *if the following condition holds: There exists a PPT oracle machine S (simulator) such that, for every PPT oracle machine D (distinguisher), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\left|\Pr\left[\begin{array}{l} C \leftarrow \mathcal{C}_n; \\ C' \leftarrow \mathsf{Obf}(C); \\ b \leftarrow D^{\ll C \gg}(C', z) \end{array} : b = 1\right] - \Pr\left[\begin{array}{l} C \leftarrow \mathcal{C}_n; \\ C'' \leftarrow S^{\ll C \gg}(1^n, z); \\ b \leftarrow D^{\ll C \gg}(C'', z) \end{array} : b = 1\right]\right| < \frac{1}{p(n)}.$$

It was proposed as a general definition in the sense that it is not specific to re-encryption. The authors gave an informal discussion that their proposed ACVBP provides a meaningful security in cryptographic settings [24, Section 2.1]. We briefly review it below. In general, VBPs should guarantee that *if a cryptographic scheme is secure when the adversary is given black-box access to a program, then it remains secure when the adversary is given the obfuscated program.* The authors claim that for a large class of applications (including re-encryption), obfuscators satisfying Definition 6 indeed give this guarantee. More specifically, the authors propose to use the following informal argument: **If** a cryptographic scheme has the following three properties:

1. The scheme is secure against black-box adversaries with sampling access to functionality X selected randomly from a family F;
2. A distinguisher D with sampling access to X can test whether an adversary A can break the security guarantee of the scheme (*distinguishable attack property*);
3. There exists a circuit obfuscator satisfying ACVBP for a class C_F of circuits implementing F;

Then the cryptographic scheme is also secure against adversaries who are given an obfuscation of a circuit selected at random from the class C_F.

The argument works for re-encryption functionalities as discussed in [24], where F is a re-encryption functionality and the cryptographic scheme is the underlying encryption scheme. However, it does NOT work for ES functionalities, where the cryptographic scheme is a pair of PKE and DS schemes, F is the ES functionality F_{ES}, and X is F_{p,sk,pk_e}. Let us check whether the argument goes through for the DS scheme. We have no problem with the first and third conditions. The first condition requires that the DS scheme satisfies the standard EU requirement according to Definition 4. The third condition requires that there exists a circuit obfuscator satisfying ACVBP for $\mathcal{C}_{ES}$. The problem is that the second condition is not satisfied in this case. The reason is that A has sampling access to the signing oracle, but D does not have.

Remark 3. Readers might ask if Definition 6 still provides a meaningful security for cryptographic schemes even when they do NOT satisfy the *distinguishable attack property.* However, it is not the case. We can show that there exists a cryptographic functionality using a secret information such that (1) the secret operation does not satisfy the distinguishable attack property and (2) it has an obfuscator satisfying the ACVBP under Definition 6, but any obfuscated circuit reveals the secret information.

As discussed above, Definition 6 is not strong enough for our purpose. In order to make it stronger, we propose a natural generalization. The generalization allows distinguishers to have sampling access not only to $\ll C \gg$ but also to a set of oracles dependent on C.

Definition 7 (ACVBP w.r.t. Dependent Oracles). *Let $T(C)$ be a set of oracles dependent on the circuit C. A circuit obfuscator* Obf *for $\mathcal{C}$ satisfies the*

ACVBP w.r.t. dependent oracle set T *if the following condition holds: There exists a PPT oracle machine S (simulator) such that, for every PPT oracle machine D (distinguisher), every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\left| \Pr\left[\begin{array}{l} C \leftarrow \mathcal{C}_n; \\ C' \leftarrow \mathsf{Obf}(C); \\ b \leftarrow D^{\ll C, T(C) \gg}(C', z) \end{array} : b = 1\right] - \Pr\left[\begin{array}{l} C \leftarrow \mathcal{C}_n; \\ C'' \leftarrow S^{\ll C \gg}(1^n, z); \\ b \leftarrow D^{\ll C, T(C) \gg}(C'', z) \end{array} : b = 1\right] \right| < \frac{1}{p(n)},$$

where $D^{\ll C, T(C) \gg}$ means that D has sampling access to all oracles contained in $T(C)$ in addition to C.

Clearly, for every T, this generalized ACVBP implies the ACVBP in Definition 6.

Remark 4. Since $T(C)$ can be viewed as *dependent* auxiliary-input to adversaries, it is natural to allow the simulator S to have access to $T(C)$. However, we did not allow it because the security proof of our obfuscator does not need it.

Now we can clarify the condition on Obf under which the EU w.r.t. the ES functionality implies the EU w.r.t. Obf.

Theorem 1. *Let $T(C_{p,sk,pk_e})$ be $\{\mathsf{S}_{p,sk}\}$. If an obfuscator Obf for $\mathcal{C}_{ES}$ satisfies ACVBP w.r.t. dependent oracle set T, then the EU w.r.t the ES functionality implies the EU w.r.t. Obf.*

Proof. We show that, if the EU w.r.t. the ES functionality is satisfied, but the stronger EU w.r.t. Obf is NOT satisfied, then it contradicts the ACVBP w.r.t. dependent oracle set T. Let A be the adversary that breaks the stronger EU. Consider the following distinguisher D that uses sampling access to $T(C_{p,sk,pk_e})$ to check whether the adversary A succeeds in breaking the stronger EU.

1. Take as input a circuit C and an auxiliary-input z. (C is either an obfuscated circuit or a simulated circuit).
2. Use the sampling access to C_{p,sk,pk_e} to get (p, pk, pk_e).
3. Use the sampling access to $\mathsf{S}_{p,sk}$ to simulate $(m, \sigma, Q) \leftarrow A^{\ll \mathsf{S}_{p,sk} \gg}(p, pk, pk_e, C, z)$.
4. Output 1 if and only if $\mathsf{V}(pk, m, \sigma) = \mathsf{Accept}$ and $m \notin Q$.

If C is an obfuscated circuit, then the probability D outputs 1 is equal to the probability that A breaks the stronger EU, which is not negligible by the assumption. On the other hand, if C is a simulated circuit, then the probability D outputs 1 is negligible, otherwise, A can be used to break the standard EU w.r.t. the ES functionality. Therefore, it contradicts the ACVBP w.r.t. dependent oracle set T. □

Remark 5. Note that the proof argument does not work under the ACVBP definition (Definition 6), where distinguishers are not allowed to use dependent oracle set T.

5 Secure Obfuscator for A Special ES Functionality

In this section, we propose an obfuscator for a special ES functionality and prove the security based on the generalized ACVBP definition. Our proposed ES functionality is the sequential composition of Waters's signature scheme and the linear encryption scheme.

5.1 Algebraic Setting and Complexity Assumptions

First of all, we review the required algebraic setting and complexity assumptions. Let Setup be an algorithm which, on input the security parameter 1^n, randomly generates the parameters for a bilinear map $(q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, where q is a prime of length n, both $\mathbb{G}$ and $\mathbb{G}_T$ are groups of order q, $\mathbf{e}$ is an efficient bilinear mapping from $\mathbb{G} \times \mathbb{G}$ to $\mathbb{G}_T$, g is a generator of $\mathbb{G}$ (e.g., refer to [6, Section 5]). The mapping $\mathbf{e}$ satisfies the following two properties: (i) $\mathsf{Bilinear}$: for all $g \in \mathbb{G}$ and $a, b \in \mathbb{Z}_q$, $\mathbf{e}(g^a, g^b) = \mathbf{e}(g, g)^{ab}$. (ii) $\mathsf{Non\text{-}degenerate}$: if g generates $\mathbb{G}$, then $\mathbf{e}(g^a, g^b) \neq 1$.

In this paper, we use the following two Diffie-Hellman assumptions. All assumptions are standard ones which have been used in the literature. The first one is so-called the Decisional Bilinear Diffie-Hellman (DBDH) assumption (e.g., see [25, 33]), which assumes that, given $g, g^a, g^b, g^c, \mathbf{e}(g, g)^d$, it is hard to check whether $abc = d$. The second one is the Decisional Linear (DL) assumption (e.g., see [5, 24]), which assumes that, given $g, g^a, g^b, g^t, (g^a)^r, (g^b)^s$, it is hard to check whether $r + s = t$.

Definition 8 (DBDH Assumption). *For every PPT machine D, every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\left| \Pr \left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{Setup}(1^n); \\ a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q; c \leftarrow \mathbb{Z}_q; \\ \mathit{decision} \leftarrow D(p, g^a, g^b, g^c, \mathbf{e}(g, g)^{abc}, z) \end{array} : \mathit{decision} = 1 \right] - \right.$$
$$\left. \Pr \left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, h) \leftarrow \mathsf{Setup}(1^n); \\ a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q; c \leftarrow \mathbb{Z}_q; d \leftarrow \mathbb{Z}_q; \\ \mathit{decision} \leftarrow D(p, g^a, g^b, g^c, \mathbf{e}(g, g)^d, z) \end{array} : \mathit{decision} = 1 \right] \right| < \frac{1}{p(n)}.$$

Definition 9 (DL Assumption). *For every PPT machine D, every polynomial $p(\cdot)$, all sufficiently large $n \in \mathbb{N}$, and every $z \in \{0,1\}^{\mathsf{poly}(n)}$,*

$$\left| \Pr \left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{Setup}(1^n); \\ a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q; r \leftarrow \mathbb{Z}_q; s \leftarrow \mathbb{Z}_q; \\ \mathit{decision} \leftarrow D(p, (g^a, g^b), (g^{r+s}, (g^a)^r, (g^b)^s), z) \end{array} : \mathit{decision} = 1 \right] - \right.$$
$$\left. \Pr \left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{Setup}(1^n); \\ a \leftarrow \mathbb{Z}_q; b \leftarrow \mathbb{Z}_q; r \leftarrow \mathbb{Z}_q; s \leftarrow \mathbb{Z}_q; t \leftarrow \mathbb{Z}_q; \\ \mathit{decision} \leftarrow D(p, (g^a, g^b), (g^t, (g^a)^r, (g^b)^s), z) \end{array} : \mathit{decision} = 1 \right] \right| < \frac{1}{p(n)}.$$

5.2 Waters's Signature Scheme

We recall Waters's signature scheme [33]. The message space is $\{0,1\}^n$.

$\mathsf{SKG}(p)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$.
2. Randomly select $\alpha \leftarrow \mathbb{Z}_q$ and compute $g_1 = g^\alpha$.
3. Randomly select $g_2 \leftarrow \mathbb{G}$ and $u' \leftarrow \mathbb{G}$.
4. Randomly select $u_i \leftarrow \mathbb{G}$ for every $i \in [n]$ and set $U = \{u_i\}_{i\in[n]}$.
5. Output $pk = (g_1, g_2, u', U)$ and $sk = (g_2^\alpha, u', U)$ as public and secret keys, respectively.

$\mathsf{Sign}(p, sk, m)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, $sk = (g_2^\alpha, u', \{u_i\}_{i\in[n]})$, and $m = (m_1, m_2, \cdots, m_n)$, where m_i denotes the i'th bit of m.
2. Randomly select $x \leftarrow \mathbb{Z}_q$.
3. Compute $(\sigma_1, \sigma_2) = (g_2^\alpha (u' \prod_{i\in\mathcal{M}} u_i)^x, g^x)$, where $\mathcal{M}$ is the set of all i such that $m_i = 1$.
4. Output $\sigma = (\sigma_1, \sigma_2)$.

$\mathsf{Verify}(p, pk, m, \sigma)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, $pk = (g_1, g_2, u', \{u_i\}_{i\in[n]})$, $m = (m_1, m_2, \cdots, m_n)$, and $\sigma = (\sigma_1, \sigma_2)$.
2. Output Accept if $\mathbf{e}(\sigma_1, g)/\mathbf{e}(\sigma_2, u' \prod_{i\in\mathcal{M}} u_i) = \mathbf{e}(g_1, g_2)$. Output Reject otherwise.

The security is proved under the DBDH assumption.

Theorem 2 ([33]). *Under the DBDH assumption, Waters's signature scheme is existentially unforgeable.*

5.3 Linear Encryption Scheme

We recall the linear encryption scheme [5]. The message space is $\mathbb{G}$.

$\mathsf{EKG}(p)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$.
2. Randomly select $a \leftarrow \mathbb{Z}_q$ and $b \leftarrow \mathbb{Z}_q$.
3. Output $pk_e = (g^a, g^b)$ and $sk_e = (a, b)$ as public and secret keys, respectively.

$\mathsf{Enc}(p, pk_e, m)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$ and $pk_e = (g^a, g^b)$.
2. Randomly select $r \leftarrow \mathbb{Z}_q$ and $s \leftarrow \mathbb{Z}_q$.
3. Compute $(c_1, c_2, c_3) = ((g^a)^r, (g^b)^s, g^{r+s}m)$.
4. Output $c = (c_1, c_2, c_3)$.

$\mathsf{Dec}(p, sk_e, c)$:

1. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, $sk_e = (a, b)$, and $c = (c_1, c_2, c_3)$.
2. Output $m = c_3/(c_1^{1/a} \cdot c_2^{1/b})$.

The security is proved under the DL assumption.

Theorem 3 ([5]). *Under the DL assumption, the linear encryption scheme satisfies the indistinguishability.*

We can view g^{r+s} as a random key generated by a KEM and (c_1, c_2) as its ciphertext. Note that the KEM encryption algorithm has the scalar homomorphic property described in Section 1.1 and Enc is rerandomizable. Specifically, given a ciphertext $c = (c_1, c_2, c_3)$ and the public key $pk_e = (g^a, g^b)$, we can rerandomize the ciphertext by computing $(c_1(g^a)^{r'}, c_2(g^b)^{s'}, c_3 g^{r'+s'})$, where r' and s' are random numbers in $\mathbb{Z}_q$. We denote by $\mathsf{ReRand}(p, pk_e, (c_1, c_2, c_3))$ the rerandomization algorithm.

5.4 The Obfuscator for the ES Functionality

Our special ES functionality is the sequential composition of Waters's signature scheme and the linear encryption scheme. Given a common parameter p, a secret signing key sk, and a public encryption key pk_e, the ES functionality F_{p,sk,pk_e} provides the following two functions:

- $\mathsf{ES}_{p,sk,pk_e}(m)$:
  1. Run $(\sigma_1, \sigma_2) \leftarrow \mathsf{Sign}(p, sk, m)$.
  2. Run $C_1 \leftarrow \mathsf{Enc}(p, pk_e, \sigma_1)$.
  3. Run $C_2 \leftarrow \mathsf{Enc}(p, pk_e, \sigma_2)$.
  4. Output (C_1, C_2).
- $\mathsf{Keys}_{p,sk,pk_e}(\mathsf{keys})$:
  1. Output (p, pk, pk_e), where pk is the public key corresponding to sk.

We define a (naive) class of circuits $\mathcal{C}_{ES} = \{\mathcal{C}_n\}_{n \in \mathbb{N}}$ for the ES functionality, which we want to obfuscate. $\mathcal{C}_n$ is a set of circuits C_{p,sk,pk_e} and each C_{p,sk,pk_e} is a naive implementation of F_{p,sk,pk_e}. Without loss of generality, we assume that we can extract (p, sk, pk_e) from C_{p,sk,pk_e}. The associated generation algorithm takes as input 1^n, generates a common parameter $p \leftarrow \mathsf{Setup}(1^n)$, runs $(pk, sk) \leftarrow \mathsf{SKG}(p)$, runs $(pk_e, sk_e) \leftarrow \mathsf{EKG}(p)$, and outputs C_{p,sk,pk_e}.

Now, we describe our proposed obfuscator Obf_{ES} for $\mathcal{C}_{ES}$ below. According to the basic idea in Section 1.1, the obfuscation is done by encrypting the signing key g_2^α and the obfuscated circuit generates a signature using the encrypted signing key.

Given a circuit C_{p,sk,pk_e}, the obfuscator Obf_{ES}

1. Extracts (p, sk, pk_e).
2. Gets pk using the Keys function.

3. Parses $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$ and $sk = (g_2^\alpha, u', U)$.
4. Runs $(c_1, c_2, c_3) \leftarrow \mathsf{Enc}(p, pk_e, g_2^\alpha)$ to encrypt g_2^α. (NOTE: We have $(c_1, c_2, c_3) = ((g^a)^r, (g^b)^s, g^{r+s} g_2^\alpha)$).
5. Sets $sk' = (c_3, u', U)$, which is an encrypted form of the signing key sk.
6. Constructs and outputs an obfuscated circuit that contains the values (p, pk_e, pk, sk', (c_1, c_2)) and does the following: (1) On input keys, outputs (p, pk, pk_e) and (2) On input a message $m \in \{0,1\}^n$,
   (a) Runs $(\sigma_1, \sigma_2) \leftarrow \mathsf{Sign}(p, sk', m)$. Note that (c_1, c_2, σ_1) is an encryption of the first part of a valid signature by Enc. (NOTE: we have $(c_1, c_2, \sigma_1) = ((g^a)^r, (g^b)^s, g^{r+s} g_2^\alpha (u' \prod_{i \in \mathcal{M}} u_i)^x)$).
   (b) Computes $C_1 = (c_1', c_2', c_3') \leftarrow \mathsf{ReRand}(p, pk_e, (c_1, c_2, \sigma_1))$.
   (c) Runs $C_2 \leftarrow \mathsf{Enc}(p, pk_e, \sigma_2)$.
   (d) Outputs (C_1, C_2).

Remark 6. For simplicity, we used Enc to encrypt σ_2 in the definition of F_{p,sk,pk_e}. However, we can use an arbitrary encryption algorithm instead of Enc and it is easy to modify the obfuscator Obf_{ES}. Furthermore, we may want to omit the encryption of σ_2 since it is just a random number and leaks no meaningful information (as long as σ_1 is encrypted). In this case, we have $C_2 = \sigma_2$ in the both definitions of F_{p,sk,pk_e} and Obf_{ES}.

Clearly, it satisfies the functionality requirement according to Definition 1. We prove that it satisfies ACVBP even though distinguishers are given sampling access to the signing oracle according to Definition 7.

Theorem 4. *Let $T(C_{p,sk,pk_e})$ be $\{\mathsf{S}_{p,sk}\}$. Under the DL assumption, Obf_{ES} satisfies ACVBP w.r.t. dependent oracle set T.*

Proof. Since we can identify an obfuscated ES circuit with the values (p, pk_e, pk, sk', (c_1, c_2)) contained in the circuit, it is sufficient to construct a simulator which simulates the values by the help of sampling access to the original circuit C_{p,sk,pk_e}. The first three values (p, pk_e, pk) can be obtained from the sampling access to C_{p,sk,pk_e} using the $\mathsf{Keys}_{p,sk,pk_e}$ function and so the question is how to simulate the last three values $(sk', (c_1, c_2))$. We show that it is sufficient to generate *junk* values for them because it is essentially an encryption of the signing key g_2^α.

Consider the following simulator S having sampling access to C_{p,sk,pk_e}.

1. Take as input the security parameter 1^n and an auxiliary-input z.
2. Use the sampling access to C_{p,sk,pk_e} to get (p, pk, pk_e).
3. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$ and $pk = (g_1, g_2, u', U)$.
4. Randomly select $Junk \leftarrow \mathbb{G}$.
5. Run $(c_1, c_2, c_3) \leftarrow \mathsf{Enc}(p, pk_e, Junk)$.
6. Set $sk' = (c_3, u', U)$.
7. Output $(p, pk_e, pk, sk', (c_1, c_2))$.

We need to show that the output distribution of S is indistinguishable from the real distribution of $(p, pk_e, pk, sk', (c_1, c_2))$ for any PPT distinguisher *even when it is allowed to have sampling access to* $CS = \{C_{p,sk,pk_e}, \mathsf{S}_{p,sk}\}$. For contradiction, assume that the probability that a distinguisher $D^{\ll CS \gg}$ can distinguish between them is not negligible. That is, the difference between the following two probabilities is not negligible. They are the probabilities that D outputs 1 given the real and simulated distributions, respectively. g_2^α is encrypted in the real distribution while $Junk$ is encrypted in the simulated distribution. It is the only difference.

$$\Pr\left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{Setup}(1^n); \\ (pk_e, sk_e) \leftarrow \mathsf{EKG}(p); \\ (pk, sk) = ((g_1, g_2, u', U), (g_2^\alpha, u', U)) \leftarrow \mathsf{SKG}(p); \\ (c_1, c_2, c_3) \leftarrow \mathsf{Enc}(p, pk_e, g_2^\alpha); \\ sk' = (c_3, u', U); \\ b \leftarrow D^{\ll CS \gg}((p, pk_e, pk, sk', (c_1, c_2)), z) : \\ b = 1 \end{array}\right]$$

$$\Pr\left[\begin{array}{l} p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g) \leftarrow \mathsf{Setup}(1^n); \\ (pk_e, sk_e) \leftarrow \mathsf{EKG}(p); \\ (pk, sk) = ((g_1, g_2, u', U), (g_2^\alpha, u', U)) \leftarrow \mathsf{SKG}(p); \\ Junk \leftarrow \mathbb{G}; \\ (c_1, c_2, c_3) \leftarrow \mathsf{Enc}(p, pk_e, Junk); \\ sk' = (c_3, u', U); \\ b \leftarrow D^{\ll CS \gg}((p, pk_e, pk, sk', (c_1, c_2)), z) : \\ b = 1 \end{array}\right]$$

Then we can construct an adversary pair (A_1, A_2) which breaks the indistinguishability of the linear encryption scheme. A_1 produces a message pair (m_1, m_2) and an associated hint h as follows:

1. Take as input a common parameter p, a public key pk_e, and auxiliary input z.
2. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$.
3. Randomly generate $(pk, sk) = ((g_1, g_2, u', U), (g_2^\alpha, u', U)) \leftarrow \mathsf{SKG}(p)$.
4. Randomly generate $Junk \leftarrow \mathbb{G}$.
5. Set $m_1 = g_2^\alpha$, $m_2 = Junk$, and $h = pk$.
6. Output (m_1, m_2, h).

Given a ciphertext c (of either m_1 or m_2), A_2 can use the distinguisher D to distinguish between m_1 and m_2 as follows:

1. Take as input a common parameter p, a public key pk_e, A_1's output (m_1, m_2, h), a ciphertext c, and auxiliary input z.
2. Parse $p = (q, \mathbb{G}, \mathbb{G}_T, \mathbf{e}, g)$, $h = pk = (g_1, g_2, u', U)$, and $c = (c_1, c_2, c_3)$.
3. Compute $sk' = (c_3, u', U = \{u_i\}_{i \in [n]})$.
4. Simulate $D^{\ll CS \gg}((p, pk_e, pk, sk', (c_1, c_2)), z)$, where the oracle queries can be perfectly simulated using p, $sk = (m_1, u', U)$, and pk_e.

5. Output the output of D.

If the target ciphertext c is a ciphertext of m_1, the probability that A_2 outputs 1 is equal to the former probability, otherwise, it is equal to the latter probability. Since the difference is not negligible, it contradicts Theorem 3. □

As a corollary, we can conclude that Waters's signature scheme is existentially unforgeable even when adversaries are given an obfuscated circuit for F_{ES}. It immediately follows from Theorems 1, 2, and 4.

Corollary 1. *Under the DL and DBDH assumptions, Waters's signature scheme is existentially unforgeable w.r.t.* Obf_{ES}.

6 Concluding Remarks

In this paper, we have constructed an obfuscator for a special ES functionality and presented a security analysis. We can generalize our construction to clarify the properties that a pair of PKE and DS schemes should satisfy so that the ES functionality can be securely obfuscated. We omit it due to the space limitation. Here, we list several DS schemes satisfying all the required properties: Lysyanskaya's unique signature scheme [28], Dodis's verifiable random function (signature scheme) [15], the undeniable signature scheme by Chaum and Antwerpen [13], the DDH-based pseudo-random function (MAC) proposed by Naor and Reingold [29], and Schnorr's signature scheme [32].

We have proposed generic approaches to obfuscating ES functionalities (Approaches 1,1′, and 2). We took Approach 1′ using the scalar homomorphic property of the linear encryption scheme. If we take Approaches 1 and 2 where the only requirement on Enc and $\mathsf{KEM.Enc}$ is rerandomizability, then we can use an encryption scheme with a relaxed version of chosen-ciphertext attack (CCA) security [10]. It is an interesting research issue to investigate what kind of CCA security we can achieve in the context of ES obfuscation.

Finally, we believe that our proposed obfuscation can be used to securely obfuscate a signcryption scheme as described in Appendix A. The formal security argument is a future work item.

Acknowledgments. I would like to thank valuable comments from anonymous reviewers of TCC'2009, ACM CCS'2009, and Eurocrypt'2010. In particular, I would like to thank an Eurocrypt reviewer for suggesting the use of the Paillier encryption scheme as mentioned in Section 1.1, another Eurocrypt reviewer for suggesting the ES functionality and obfuscator presented in Section 5.4, which are much simpler than my original ones, and Henri Gilbert for helping me to revise the submitted version.

References

1. B. Adida, D. Wikstrom, "How to shuffle in public," Proceedings of TCC 2007, pp. 555-574, 2007.

2. J. H. An, Y. Dodis, and T. Rabin, "On the Security of Joint Signature and Encryption," Proceedings of Eurocrypt'02, pp. 83-107, 2002.
3. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, K. Yang, "On the (Im)possibility of Obfuscating Programs," ECCC,Report No. 57, 2001. (A conference version appeared in Proceedings of CRYPTO'2001, pp. 1-18, 2001)
4. M. Bellare and P. Rogaway, "Random Oracles are Practical: a paradigm for designing efficient protocols," Proceedings of the 1st ACM Conference on Computer and Communications Security, pp. 62-73, 1993.
5. D. Boneh, X. Boyen, and H. Shacham, "Short Group Signatures," Proceedings of CRYPTO'04, pp. 41-55, 2004.
6. D. Boneh and M. Franklin, "Identity based encryption from the Weil pairing, " SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2003.
7. D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing," Proceedings of ASIACRYPT'2001, pp. 514-532, 2001.
8. R. Canetti, "Towards Realizing Random Oracles: Hash Functions that Hide All Partial Information," Proceedings of CRYPTO'97, pp.455-469, 1997.
9. R. Canetti and R. R. Dakdouk, "Obfuscating Point Functions with Multibit Output," Proceedings of Eurocrypt'2008, pp. 489-508, 2008.
10. R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing Chosen-Ciphertext Security," Proceedings of CRYPTO'03, pp. 565-582, 2003.
11. R. Canetti, D. Micciancio, and O. Reingold, "Perfectly One-way Probabilistic Hash Functions," Proceedings of 30st STOC, 1998.
12. R. Canetti and M. Varia, "Non-malleable Obfuscation," Proceedings of TCC'09, pp. 73-90, 2009.
13. D. Chaum, H. van Antwerpen, "Undeniable Signatures," Proceedings of CRYPTO'89, 212-216.
14. R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," SIAM Journal on Computing, Vol. 33, No. 11, pp. 167-226, 2003.
15. Y. Dodis, "Efficient Construction of (Distributed) Verifiable Random Functions," Proceedings of PKC'02, pp. 1-17, 2002.
16. Y. Dodis and A. Smith, "Correcting Errors without Leaking Partial Information," Proceedings of 37th STOC, 2005.
17. T. ElGamal, "A public key cryptosystem and signature scheme based on discrete logarithms," IEEE Trans. Inform. Theory, Vol. 31, pp. 469-472, 1985.
18. C. Gentry, "A fully homomorphic encryption scheme," PhD Thesis, 2009.
19. O. Goldreich, "Foundations of Cryptography: Volume II Basic Applications," Cambridge University Press, 2004.
20. S. Goldwasser and Y. T. Kalai, "On the Impossibility of Obfuscation with Auxiliary Input," Proceedings of FOCS'05, pp. 553-562, 2005.
21. S. Goldwasser and G. N. Rothblum, "On Best-Possible Obfuscation," Proceedings of TCC'07, 2007.
22. S. Hada, "Zero-Knowledge and Code Obfuscation, " Proceedings of Asiacrypt'2000, pp. 443-457, 2000.
23. D. Hofheinz, J. Malone-Lee, and M. Stam, "Obfuscation for Cryptographic Purposes," Proceedings of TCC'07, 2007.
24. S. Hohenberger, G. N. Rothblum, a. shelat, and V. Vaikuntanathan, "Securely Obfuscating Re-Encryption," Proceedings of TCC'07, 2007.
25. A. Joux, "The Weil and Tate pairings as building blocks for public key cryptosystems," Proceedings of the Fifth Algorithmic Number Theory Symposium, LNCS Vol. 2369, pp. 20-32, 2002.

26. A. Joux and K. Nguyen, "Separating Decision Diffie-Hellman from Diffie-Hellman in Cryptographic Groups," Journal of Cryptology, Vol. 16, Num. 4, pp. 239-247, 2003.
27. B. Lynn, M. Prabhakaran, and A. Sahai, "Positive Results and Techniques for Obfuscation," In Proceedings of Eurocrypt, 2004.
28. A. Lysyanskaya, "Unique Signatures and Verifiable Random Functions from the DH-DDH Separation," Proceedings of CRYPTO'02, 2002.
29. M. Naor and O. Reingold, "Number-Theoretic Constructions of Efficient Pseudo-Random Functions," Journal of the ACM, Vol. 51, Issue 2, pp. 231-262, 2004.
30. R. Ostrovsky and W. E. Skeith III, "Private searching on streaming data," Proceedings of CRYPTO'05, pp. 223-240, 2005.
31. P. Paillier, "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes," Proceedings of Eurocrypt'99, pp. 223-238, 1999.
32. C. P. Schnorr, "Efficient Signature Generation by Smart Cards," *Journal of Cryptology*, Vol.4, No.3, pp.161–174, 1991.
33. B. Waters, "Efficient Identity-Based Encryption Without Random Oracles," Proceedings of Eurocrypt'2005, pp. 114-127, 2005.
34. H. Wee, "On Obfuscating Point Functions," Proceedings of STOC'05, pp. 523-532, 2005.
35. Y. Zheng, "Digital Signcryption or How to Achieve Cost (Signature & Encryption) << Cost (Signature) + Cost (Encryption)", Proceedings of CRYPTO'97, pp.165-179, 1997.

A A Relation to Signcryption

In this appendix, we informally describe (1) how to use an ES functionality as a building block to construct a secure signcryption scheme and (2) how to obfuscate the resulting signcryption scheme using an obfuscator for the ES functionality.

A.1 EncryptedSignature-then-Encryption

We propose a new composition method which we call EncryptedSignature-then-Encryption (EStE) as a new approach to constructing a secure signcryption scheme: To signcrypt a message, we generate an encrypted signature and encrypt both the message and encrypted signature. More specifically, given a message m, we compute $\sigma \leftarrow \mathsf{S}(p, sk, m)$, $c_1 \leftarrow \mathsf{E}_1(p, pk, \sigma)$, and $c_2 \leftarrow \mathsf{E}_2(p, pk, (m, c_1))$, where the sequential composition of S and E_1 is the ES functionality, c_1 is the encrypted signature, and c_2 is the resulting signcryption of m. The difference from the standard StE composition is that the signature σ is doubly encrypted. The first encryption E_1 is by the ES functionality and the second encryption E_2 can be done by a standard hybrid encryption as in StE (using the same public encryption key pk).

We follow the security argument of [2] to show that the EStE-based signcryption scheme satisfies a meaningful security requirement (privacy and authenticity properties). In [2], two security formalizations are considered: Outsider security and insider security. In outsider security, adversaries are outsiders who only know

the public keys (p, pk, pk_e). On the other hand, in insider security, adversaries are insiders who know either the signing key sk or the decryption key sk_e in addition to the public keys (p, pk, pk_e). We focus on insider security since it is stronger. The insider security is defined in terms of *induced* PKE and DS schemes (See [2] for the meaning of *induced*). That is, we say that a signcryption scheme is *insider-secure* if the induced PKE and DS schemes are secure. More specifically, we say that a signcryption scheme is *insider-secure* against CPAs and CMAs if the induced PKE and DS schemes satisfy the indistinguishability against CPAs and the existential unforgeability against CMAs, respectively (For simplicity, we don't consider the indistinguishability against chosen-ciphertext attacks). Following the security argument in [2], we can show that (1) if the PKE scheme of E_2 satisfies the indistinguishability against CPAs then the induced PKE scheme does so (The indistinguishability of E_1 does not matter) and (2) if the DS scheme of S satisfies the existential unforgeability against CMAs then the induced DS scheme does so. Therefore, we can say that the EStE-based signcryption scheme provides a meaningful security if E_2 and S are secure as in the two statements. A next question is how to obfuscate the EStE-based signcryption functionality.

A.2 Obfuscation for EStE

A secure obfuscator for an ES functionality can be used to obfuscate the EStE-based signcryption functionality since the second encryption E_2 is just a public operation. In other words, given an obfuscated ES program, we can append a program for performing the second encryption to it so that the resulting program computes the EStE composition, where we don't need any extra secret information. Therefore, by an argument similar to Section 5.4, we can show that the resulting obfuscator satisfies the ACVBP against distinguishers having sampling access to the signcryption and signing oracles and that the security of the DS scheme is preserved even when adversaries are given an obfuscated signcryption program. A question here is what kind of security we can achieve for the signcryption scheme (rather than the DS scheme) when adversaries are given an obfuscated signcryption program. This is not a trivial question. For example, the insider security of the signcryption scheme is violated when adversaries have access to the obfuscated program and the secret decryption key. The formal security argument is outside the scope of this proceedings version.