Indistinguishability Obfuscation for All Circuits from Secret-Key Functional Encryption

Fuyuki Kitagawa [1] Ryo Nishimaki [2] Keisuke Tanaka [1]

[1] Tokyo Institute of Technology, Japan
{kitagaw1,keisuke}@titech.ac.jp
[2] NTT Secure Platform Laboratories, Japan
nishimaki.ryo@lab.ntt.co.jp

Abstract

We show that indistinguishability obfuscation (IO) for all circuits can be constructed solely from secret-key functional encryption (SKFE). In the construction, SKFE need to be able to issue a-priori unbounded number of functional keys, that is, collusion-resistant.

Our strategy is to replace public-key functional encryption (PKFE) in the construction of IO proposed by Bitansky and Vaikuntanathan (FOCS 2015) with *puncturable SKFE*. Bitansky and Vaikuntanathan introduced the notion of puncturable SKFE and observed that the strategy works. However, it has not been clear whether we can construct puncturable SKFE without assuming PKFE. In particular, it has not been known whether puncturable SKFE is constructed from ordinary SKFE.

In this work, we show that a relaxed variant of puncturable SKFE can be constructed from collusion-resistant SKFE. Moreover, we show that the relaxed variant of puncturable SKFE is also sufficient for constructing IO.

Keywords: Indistinguishability obfuscation, Secret-key functional encryption, Puncturable secret-key functional encryption

Contents

1 Introduction

1.1 Backgrounds

Program obfuscation is now one of the central topics in cryptography. Program obfuscation aims to turn programs "unintelligible" while preserving its functionality. The theoretical study of program obfuscation was initiated by Barak, Goldreich, Impagliazzo, Rudich, Sahai, Vadhan, and Yang [BGI+01]. They introduced *virtual-black-box* obfuscation as a formal definition of obfuscation. The definition of virtual black-box obfuscation is intuitive and naturally captures the requirement that obfuscators hide information about programs. However, Barak et al. showed that it is impossible to achieve virtual black-box obfuscation for all circuits. In order to avoid the impossibility result, they also defined an weaker variant of obfuscation called *indistinguishability obfuscation (IO)*. Impossibility of IO for all circuits is not known.

Garg, Gentry, Halevi, Raykova, Sahai, and Waters [GGH+13] proposed the first candidate construction of IO for all circuits. Subsequently, many works have shown that IO is powerful enough in the sense that we can achieve a wide variety of cryptographic primitives based on IO though it is weaker than virtual-black-box obfuscation [GGH+13, SW14, HSW14, BGMS15, KLW15, BGL+15, CHJV15, BPW16, CHN+16, HJK+16].

While we know the usefulness of IO well, we know very little about how to achieve IO. Although the first candidate construction was demonstrated, we are still at the embryonic stage for constructing IO. All known constructions of IO are based on a little-studied cryptographic tool called multi-linear maps [GGH+13, BGK+14, BR14, AGIS14, PST14, Zim15, AB15, BMSZ16, GMM+16, Lin16a, LV16, AS17, Lin16b, FRS16]. Moreover, security flaws were discovered in some IO constructions [CGH+15, MSZ16, ADGM17, CLLT17, CGH17].

Therefore, constructing IO based on a standard assumption is still standing as a major open question in the study of cryptography. As a stepping-stone for solving the question, it is important to find a seemingly weaker primitive that implies IO. As such a cryptographic primitive, we already have *functional encryption*.

Functional encryption is one of the most advanced cryptographic primitives which enable a system having flexibility in controlling encrypted data [SW05, BSW11, O'N10]. In functional encryption, an owner of a master secret key MSK can generate a functional decryption key sk_f for a function f belonging to a function family $\mathcal{F}$. By decrypting a ciphertext of a message m using sk_f, a holder of sk_f can learn only a value $f(m)$. No information about x except $f(m)$ is revealed from the ciphertext of m. This feature enables us to construct a cryptographic system with fine-grained access control. In addition, it is known that functional encryption is a versatile building block to construct other cryptographic primitives. In particular, we can construct IO for all circuits by using functional encryption that satisfies certain security notions and efficiency requirements [AJ15, BV15, AJS15, BNPW16].

Bitansky and Vaikuntanathan [BV15] and Ananth and Jain [AJ15] independently showed that we can construct IO based on public-key functional encryption (PKFE) which supports a single functional key and whose encryption circuit size is sub-linear in the size of functions. A functional encryption scheme that supports a single key is called a *single-key* scheme. A functional encryption scheme that satisfies the efficiency property above is said to be *weakly-succinct*.

Bitansky, Nishimaki, Passelègue, and Wichs [BNPW16] subsequently showed that *collusion-resistant* secret-key functional encryption (SKFE) is powerful enough to yield IO if we additionally assume plain public key encryption. Collusion-resistant functional encryption is functional encryption that can securely issue a-priori unbounded number of functional keys.

From these results, we see that the combination of functional encryption with some property and a public-key cryptographic primitive is sufficient for achieving IO. This fact is a great progress as a stepping-stone for achieving IO based on a standard assumption.

However, one natural question arises for this situation. The question is whether we really need public-key primitives to constructing IO or not. In other words, we have the following fundamental question:

Is it possible to achieve IO for all circuits based solely on secret-key primitives?

SKFE is the best possible candidate for a secret-key cryptographic primitive that gives an affirmative answer to this question. However, Asharov and Segev [AS15] gave a somewhat negative answer to the question. Their result can be seen as a substantial evidence that SKFE is somewhat unlikely to imply IO as long as we use *black-box* techniques.[1] Thus, we need a *non-black-box* technique to achieve IO based on SKFE.

The real power of IO appears in the fact that it can transform secret-key primitives into public-key ones. Therefore, solving the above problem is a key advancement to discover the exact requirements for achieving IO.

[1] More precisely, Asharov and Segev [AS15] introduced an extended model for black-box reductions to include a limited class of non-black-box reductions into their impossibility results.

1.2 Our Results

We give an affirmative answer to the question above. More precisely, we prove the following theorem.

Theorem 1.1 (Informal). *Assuming there exists sub-exponentially secure collusion-resistant SKFE for all circuits. Then, there exists IO for all circuits.*

Since our construction of IO is *non-black-box*, we can circumvent the impossibility result shown by Asharov and Segev [AS15].

The security loss of our construction of IO is exponential in the input length of circuits, but is independent of the size of circuits. Thus, if the input length of circuits is poly-logarithmic in the security parameter, our construction of IO incurs only quasi-polynomial security loss regardless of the size of circuits. Therefore, we can obtain IO for circuits of *polynomial size* with input of poly-logarithmic length from *quasi-polynomially secure* collusion-resistant SKFE for all circuits. This is an improvement over the IO construction by Komargodski and Segev [KS17]. They showed that IO for circuits of *sub-polynomial size* with input of poly-logarithmic length is constructed from quasi-polynomially secure collusion-resistant SKFE for all circuits.

We show Theorem 1.1 by using *puncturable SKFE*. The notion of puncturable SKFE was introduced by Bitansky and Vaikuntanathan [BV15]. They showed that in their construction of IO, the building block PKFE can be replaced with puncturable SKFE. However, it has been an open issue whether we can achieve puncturable SKFE without assuming the existence of PKFE.

In this work, we show how to construct puncturable SKFE that is sufficient for constructing IO, based solely on SKFE. More precisely, we show the following theorem.

Theorem 1.2 (Informal). *Assuming there exists collusion-resistant SKFE for all circuits. Then, there exists single-key weakly-succinct puncturable SKFE for all circuits.*

Note that our definition of puncturable SKFE is slightly different from that proposed by Bitansky and Vaikuntanathan. Our requirement for puncturable SKFE looks weaker than that of Bitansky and Vaikuntanathan. However, they are actually incomparable. In fact, we show that puncturable SKFE defined in this paper is also sufficient for a building block of IO. See Section 2 for the details of the notion of puncturable SKFE and the difference between our definition and that of Bitansky and Vaikuntanathan.

Our construction is a generic transformation and does not yield a new instantiation of IO. This is because all known assumptions that imply SKFE also imply PKFE. However, SKFE is an weaker primitive than PKFE, and thus the requirements for constructing IO looks to be relaxed than ever by our result. We believe that our result makes easier to design IO based on other cryptographic primitives. Moreover, our result makes a progress on the study of IO and functional encryption as we note in the next paragraph.

Impacts on the hierarchy of cryptographic primitives. It is known that we can classify cryptographic primitives into two hierarchies Minicrypt and Cryptomania since the beautiful work of Impagliazzo and Rudich [IR89] showed that public-key encryption is not implied by one-way functions via black-box reductions. The terminologies, Minicrypt and Cryptomania, were introduced by Impagliazzo [Imp95]. In Minicrypt, one-way functions exist, but public-key encryption does not. In Cryptomania, public-key encryption also exists.

We have recently started to consider a new hierarchy called Obfustopia. Garg, Pandey, Srinivasan, and Zhandry [GPSZ17] introduced the term Obfustopia, which seems to indicate the "world" where there exists IO. Garg et al. did not give a formal definition of Obfustopia. In this paper, we explicitly define Obfustopia as the "world" where there exists efficient IO for all circuits and one-way functions.[2] It is known that we can construct almost all existing cryptographic primitives which are stronger than public-key encryption by using IO. This is the reason why we consider the new hierarchy beyond Cryptomania.[3]

The landscape of Obfustopia is not clear while those of Minicrypt and Cryptomania are. In particular, we do not know how to construct IO based on standard assumptions. There has been significant effort to find out cryptographic primitives that are in Obfustopia. That is, we have been asking what kind of cryptographic primitive

[2] Komargodski, Moran, Naor, Pass, Rosen, and Yogev proved that IO implies one-way functions under a mild complexity theoretic assumption [KMN+14]. More specifically, the complexity assumption is $\mathsf{NP} \not\subseteq \mathsf{io\text{-}BPP}$, where $\mathsf{io\text{-}BPP}$ is the class of languages that is decided by probabilistic polynomial-time algorithms for infinitely many input sizes. Therefore, under the assumption, we say that Obfustopia is the complexity spectrum where efficient IO for all circuits exists.

[3] Strictly speaking, it was known that there are stronger primitives than public-key encryption before the candidate of obfuscation appeared. For example, public-key encryption does not imply identity-based encryption [BPR+08].

implies the existence of IO. We know that sub-exponentially-secure succinct PKFE exists in Obfustopia [BV15, AJ15].

It is natural to ask whether SKFE is also in Obfustopia or not since SKFE seems to be a strong primitive similarly to PKFE. Asharov and Segev [AS15] gave a somewhat negative answer to this question. They showed that SKFE is unlikely to imply IO as long as we use well-known black-box techniques. They also showed that SKFE does not imply any primitive in Cryptomania via black-box reductions. Moreover, it was not known whether SKFE implies any primitive outside Minicrypt even if we use it in a non-black-box manner before the work of Bitansky et al. [BNPW16].

Bitansky et al. showed that the combination of sub-exponentially secure collusion-resistant SKFE and exponentially secure one-way functions implies quasi-polynomially secure public-key encryption. This also implies that the above combination yields quasi-polynomially secure succinct PKFE from their main result showing that the combination of collusion-resistant SKFE and PKE implies succinct PKFE.

Komargodski and Segev [KS17] showed that quasi-polynomially secure IO for circuits of sub-polynomial size with input of poly-logarithmic length can be constructed from quasi-polynomially secure collusion-resistant SKFE for all circuits. In addition, they showed that by combining quasi-polynomially secure collusion-resistant SKFE and sub-exponentially secure one-way functions, we can construct quasi-polynomially secure succinct PKFE. However, in this construction, the resulting PKFE supports only circuits of sub-polynomial size with input of poly-logarithmic length though the building block SKFE supports all polynomial size circuits.

These two results surely demonstrated that SKFE is stronger than we thought. Nevertheless, we see that both two results involves degradation of security level or functionality. Thus, it is still open whether SKFE implies a cryptographic primitive other than those in Minicrypt without such degradation, and especially SKFE is in Obfustopia or not.

We gives an affirmative answer to this question. More concretely, we can construct sub-exponentially secure IO for all circuits from sub-exponentially secure collusion-resistant SKFE for all circuits through our transformation by setting security parameter appropriately. This result means that sub-exponentially secure collusion-resistant SKFE exists in Obfustopia. In addition, by combining this result and the result by Garg et al. [GGH+13], we see that the existence of sub-exponentially secure collusion-resistant PKFE for all circuits is equivalent to that of sub-exponentially secure collusion-resistant SKFE for all circuits.

Organization. The rest of this paper consists of the following parts. In Section 2, we provide an informal overview of our constructions and proofs so that readers understand our main ideas. In Section 3, we provide notations and definitions of cryptographic primitives. In Section 4, we present our definition of puncturable SKFE. In Section 5, we provide the construction of single-key non-succinct puncturable SKFE and prove its security. In Section 6, we provide transformation from a non-succinct puncturable SKFE scheme to an weakly succinct one and prove its security. In Section 7, we provide our IO for all circuits based on single-key weakly succinct puncturable SKFE and analyze its security and efficiency.

2 Overview of Our Technique

Before we introduce formal definitions and constructions, we give an overview of our construction of IO based on SKFE in this section.

Our basic strategy is to replace PKFE in the construction of Bitansky and Vaikuntanathan [BV15] with puncturable SKFE. Bitansky and Vaikuntanathan observed that this strategy works. However, it is not known whether puncturable SKFE is constructed from cryptographic primitives other than PKFE or IO.

In this work, we show that we can construct a relaxed variant of puncturable SKFE that is a single-key scheme and weakly-succinct from collusion-resistant SKFE. Moreover, we show that such a relaxed variant of puncturable SKFE is sufficient for constructing IO.

We first give an overview of the construction of Bitansky and Vaikuntanathan [BV15] in Section 2.1 and explain why SKFE must be "puncturable" when we replace PKFE with SKFE in their construction in Section 2.2. Next, we give an overview of how to construct our puncturable SKFE scheme and IO in Section 2.3 and Section 2.4, respectively.

2.1 Construction of IO based on PKFE

The main idea of Bitansky and Vaikuntanathan is to design an obfuscator $i\mathcal{O}_i$ for circuits with i-bit input from an obfuscator $i\mathcal{O}_{i-1}$ for circuits with $(i-1)$-bit input. If we can design such a bit extension construction, for any polynomial n, we can construct an obfuscator $i\mathcal{O}_n$ for circuits with n-bit input since we can easily achieve $i\mathcal{O}_1$ for circuits with 1-bit input by outputting an entire truth table of a circuit with 1-bit input.

When we construct IO based on the bit extension construction above, it is important to avoid a circuit-size blow-up of circuits to be obfuscated at each recursive step. In fact, if we allow a circuit-size blow-up, we can obtain the bit extension construction by defining $i\mathcal{O}_i(C(x_1 \cdots x_i)) := i\mathcal{O}_{i-1}(C(x_1 \cdots x_{i-1}\|0))\|i\mathcal{O}_{i-1}(C(x_1 \cdots x_{i-1}\|1))$. However, this construction obviously incurs an exponential blow-up and thus we cannot rely on this solution. Bitansky and Vaikuntanathan showed how to achieve the bit extension construction without an exponential blow-up using *weakly-succinct* PKFE.

In their construction, a functional key of PKFE should hide information about the corresponding circuit. Such security property is called function privacy. However, it is not known how to achieve function private PKFE. Then, Bitansky and Vaikuntanathan explicitly accommodated the technique for function private SKFE proposed by Brakerski and Segev [BS15] to their IO construction based on PKFE.

We review their construction based on PKFE. For simplicity, we ignore the issue of the randomness for encryption algorithms. It is generated by puncturable pseudorandom function (PRF) in the actual construction.

$i\mathcal{O}_i$ based on $i\mathcal{O}_{i-1}$ and PKFE works as follows. The construction additionally uses plain secret key encryption (SKE) to implement the technique used by Brakerski and Segev [BS15]. To obfuscate a circuit C with i-bit input, it first generates a key pair $(\mathsf{PK}_i, \mathsf{MSK}_i)$ of PKFE. Then, using MSK_i, it generates a functional key sk_{C^*} tied to the following circuit C^*. C^* has hardwired two SKE ciphertexts $\mathsf{CT}_0^{\mathsf{ske}}$ and $\mathsf{CT}_1^{\mathsf{ske}}$ of plaintext C under independent keys K_0 and K_1, respectively. C^* expects as an input not only an i-bit string $\boldsymbol{x}_i$ but also an SKE key K_b. On those inputs, C^* first obtains C by decrypting $\mathsf{CT}_b^{\mathsf{ske}}$ by K_b and outputs $U(C, \boldsymbol{x}_i) = C(\boldsymbol{x}_i)$, where $U(\cdot,\cdot)$ is an universal circuit. Finally, the construction obfuscates the following encryption circuit E_{i-1} by $i\mathcal{O}_{i-1}$. E_{i-1} has hardwired PK_i and K_b. On input $(i-1)$-bit string $\boldsymbol{x}_{i-1}$, it outputs ciphertexts $\mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|0, K_b))$ and $\mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|1, K_b))$, where Enc is the encryption algorithm of PKFE. The resulting obfuscation of C is a tuple $(\mathsf{sk}_{C^*}, i\mathcal{O}_{i-1}(\mathsf{E}_{i-1}))$. Note that we always set the value of b as 0 in the actual construction. We set b as 1 only in the security proof.

// Description of (simplified) C^*	// Description of (simplified) E_{i-1}
Hard-Coded Constants: $\mathsf{CT}_0^{\mathsf{ske}}$, $\mathsf{CT}_1^{\mathsf{ske}}$. **Input:** $\boldsymbol{x}_i$, K_b 1. Compute $C = \mathsf{D}(K_b, \mathsf{CT}_b^{\mathsf{ske}})$. 2. Return $U(C, \boldsymbol{x}_i)$.	**Hard-Coded Constants**: PK_i, K_b. **Input:** $\boldsymbol{x}_{i-1} \in \{0,1\}^{i-1}$ 1. Compute $\mathsf{CT}_{i,x_i} \xleftarrow{r} \mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|x_i, K_b))$. 2. Output $\mathsf{CT}_{i,0}$ and $\mathsf{CT}_{i,1}$.

When we evaluate the above obfuscated C on input $\boldsymbol{x}_i = x_1 \cdots x_{i-1}x_i \in \{0,1\}^i$, we first invoke $i\mathcal{O}(\mathsf{E}_{i-1})$ on input $\boldsymbol{x}_{i-1} = x_1 \cdots x_{i-1}$ and obtain $\mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|0, K_b))$ and $\mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|1, K_b))$. Then, by decrypting $\mathsf{Enc}(\mathsf{PK}_i, (\boldsymbol{x}_{i-1}\|x_i, K_b))$ by sk_{C^*}, we obtain $C(\boldsymbol{x}_i)$.

Consequently, by using this bit extension construction, the obfuscation of a circuit C with n-bit input consists of n functional keys $\mathsf{sk}_1, \cdots, \mathsf{sk}_n$ each of which is generated under a different master secret key MSK_i, and pair of ciphertexts of 0 and 1 under PK_1 corresponding to MSK_1. For any $\boldsymbol{x}_n = x_1 \cdots x_n \in \{0,1\}^n$, we can first compute a ciphertext of $\boldsymbol{x}_n$ by repeatedly decrypting a ciphertext of $\boldsymbol{x}_{i-1} = x_1 \cdots x_{i-1}$ by sk_{i-1} and obtaining a ciphertext of $\boldsymbol{x}_i = x_1 \cdots x_i$ for every $i \in \{2, \cdots, n\}$. We can finally obtain $C(\boldsymbol{x}_n)$ by decrypting the ciphertext of $\boldsymbol{x}_n$ by sk_n.

In this construction, each instance of PKFE needs to issue only one functional key. This is a minimum requirement for functional encryption. However, for efficiency, PKFE in the construction above should satisfy a somewhat strong requirement, that is, weak-succinctness to avoid a circuit-size blow-up of circuits to be obfuscated at each recursive step. Therefore, we need to use a single-key weakly-succinct PKFE scheme in the IO construction above.

We can prove the security of the construction recursively. More precisely, we can prove the security of $i\mathcal{O}_i$ based on those of $i\mathcal{O}_{i-1}$, PKFE, and SKE. Note that it is sufficient that PKFE satisfies a mild selective-security to complete the proof. Their security proof relies on the argument of probabilistic IO formalized by Canneti, Lin, Tessaro, and Vaikuntanathan [CLTV15], and thus the security loss of each recursive step is exponential in i, that is 2^i. This is the reason their building block PKFE must be sub-exponentially secure.

2.2 Replacing PKFE with SKFE: Need of Puncturable SKFE

The security proof of Bitansky and Vaikuntanathan relies on the fact that we can use the security of PKFE even when its encryption circuit is publicly available. Concretely, PK_i is hardwired into obfuscated encryption circuit $i\mathcal{O}_{i-1}(\mathsf{E}_{i-1})$ and this encryption circuit is public when we use the security of PKFE under the key pair $(\mathsf{PK}_i, \mathsf{MSK}_i)$.

The above security argument might not work if ordinary SKFE is used instead of PKFE. This intuition comes from the impossibility result shown by Barak et al. [BGI+01]. In fact, Bitansky and Vaikuntanathan showed that it is impossible to instantiate their IO by using SKFE. More precisely, they showed that there exists a secure SKFE scheme such that their transformation results in insecure IO if the SKFE scheme is used as the building block. This is why they adopted PKFE as their building block. Therefore, in order to replace PKFE with SKFE in the construction above, we need SKFE whose security holds even when its encryption circuit is publicly available. As one of such primitives, Bitansky and Vaikuntanathan proposed *puncturable SKFE*.

In puncturable SKFE defined by Bitansky and Vaikuntanathan, there are a puncturing algorithm Punc and a punctured encryption algorithm PEnc in addition to algorithms of ordinary SKFE. We can generate a punctured master secret key $\mathsf{MSK}^*\{m_0, m_1\}$ at two messages m_0 and m_1 from a master secret key MSK by using Punc. Puncturable SKFE satisfies the following two properties: *functionality preserving under puncturing* and *semantic security at punctured point*. Functionality preserving under puncturing requires that

$$\mathsf{Enc}(\mathsf{MSK}, m; r) = \mathsf{PEnc}(\mathsf{MSK}^*\{m_0, m_1\}, m; r)$$

holds for any message m other than m_0 and m_1 and for any randomness r. Semantic security at punctured point requires that

$$(\mathsf{MSK}^*\{m_0, m_1\}, \mathsf{Enc}(\mathsf{MSK}, m_0) \stackrel{\mathsf{c}}{\approx} (\mathsf{MSK}^*\{m_0, m_1\}, \mathsf{Enc}(\mathsf{MSK}, m_1))$$

holds for all adversaries, where $\stackrel{\mathsf{c}}{\approx}$ denotes computational indistinguishability.

Bitansky and Vaikuntanathan showed that single-key weakly-succinct puncturable SKFE is also a sufficient building block for their IO construction while ordinary SKFE is not. Note that weak-succinctness of puncturable SKFE requires that not only the encryption circuit but also the punctured encryption circuit should be weakly-succinct. However, as stated earlier, there was no instantiation of puncturable SKFE other than regarding PKFE as puncturable SKFE at that point. In particular, it was not clear whether we can construct puncturable SKFE based on ordinary SKFE.

2.3 Puncturable SKFE from SKFE

In this work, we show we can construct single-key weakly-succinct puncturable SKFE from collusion-resistant SKFE. More specifically, we show the following two results. First, we show how to construct single-key non-succinct puncturable SKFE based only on one-way functions. In addition, we show that we can transform it into single-key weakly-succinct one using collusion-resistant SKFE. Our formalization of puncturable SKFE is different from that of Bitansky and Vaikuntanathan [BV15] in several aspects. Nevertheless, we show that our puncturable SKFE is also sufficient for constructing IO.

Below, we give the overview of these two constructions.

Single-Key Non-Succinct Puncturable SKFE based on One-Way Functions

Our starting point is the SKFE variant of the single-key non-succinct PKFE scheme proposed by Sahai and Seyalioglu [SS10]. It is constructed from garbled circuit and SKE, which are implied by one-way functions. Their construction is as follows.

Setup: A master secret key consists of $2s$ secret keys $\{K_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}$ of SKE, where s is the length of a binary representation of functions supported by the resulting SKFE scheme.

Enc: When we encrypt a message m, we first generates a garbled circuit $\widetilde{U}_m$ with labels $\{L_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}$ by garbling an universal circuit $U(\cdot, m)$ into which m is hardwired. Then, we encrypt $L_{j,\alpha}$ under $K_{j,\alpha}$ and obtain an SKE ciphertext $c_{j,\alpha}$ for every $j \in [s]$ and $\alpha \in \{0, 1\}$. The resulting ciphertext of the scheme is $(\widetilde{U}_m, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$.

KeyGen: A functional key sk_f for a function f consists of $\{K_{j,f[j]}\}_{j\in[s]}$, where $f[1]\cdots f[s]$ is the binary representation of f and each $f[j]$ is a single bit.

Dec: A decryptor who has a ciphertext $(\widetilde{U}_m, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$ and a functional key $\{K_{j,f[j]}\}_{j\in[s]}$ can compute $\{L_{j,f[j]}\}_{j\in[s]}$ by decrypting each $c_{j,f[j]}$ by $K_{j,f[j]}$ and obtain $\widetilde{U}_m(\{L_{j,f[j]}\}_{j\in[s]}) = U(f,m) = f(m)$.

In the construction above, we observe that if we use puncturable PRF instead of SKE, the resulting scheme is puncturable in some sense. More specifically, a master secret key now consists of $2s$ puncturable PRF keys $\{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}$. When we encrypt a message m, we first generate $(\widetilde{U}_m, \{L_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$ and encrypt each label by using a puncturable PRF value. That is, $c_{j,\alpha} \leftarrow L_{j,\alpha} \oplus \mathsf{F}_{S_{j,\alpha}}(\mathsf{tag})$, where F is puncturable PRF and tag is a public tag chosen in some way.

In this case, we can generate a punctured master secret key $\mathsf{MSK}^*\{\mathsf{tag}\}$ at a tag tag. Thus, we define an encryption algorithm in a tag-based manner. The encryption algorithm Enc, given MSK, tag, and m, outputs a ciphertext of m under the tag tag. That is, $\mathsf{Enc}(\mathsf{MSK},\mathsf{tag},m) = (\widetilde{U}_m, \{L_{j,\alpha} \oplus \mathsf{F}_{S_{j,\alpha}}(\mathsf{tag})\}_{j\in[s],\alpha\in\{0,1\}})$. A punctured master secret key $\mathsf{MSK}^*\{\mathsf{tag}\}$ consists of $2s$ puncturable PRF keys $\{S^*_{j,\alpha}\{\mathsf{tag}\}\}_{j\in[s],\alpha\in\{0,1\}}$ all of which are punctured at tag.

By using $\mathsf{MSK}^*\{\mathsf{tag}\}$, we can generate a ciphertext of any message m under a tag tag' different from tag, that is, $\mathsf{PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\},\mathsf{tag}',m) = (\widetilde{U}_m, \{L_{j,\alpha} \oplus \mathsf{F}_{S^*_{j,\alpha}\{\mathsf{tag}\}}(\mathsf{tag}')\}_{j\in[s],\alpha\in\{0,1\}})$. Then, we have

$$\mathsf{Enc}(\mathsf{MSK},\mathsf{tag}',m;r) = \mathsf{PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\},\mathsf{tag}',m;r)$$

for any tag tag and tag' such that $\mathsf{tag} \neq \mathsf{tag}'$, message m, and randomness r due to the functionality preserving property of puncturable PRF. Namely, this scheme satisfies functionality preserving under puncturing.

In addition, we can prove that $\mathsf{Enc}(\mathsf{MSK},\mathsf{tag},m_0)$ and $\mathsf{Enc}(\mathsf{MSK},\mathsf{tag},m_1)$ are indistinguishable for adversaries that have $\mathsf{MSK}^*\{\mathsf{tag}\}$ based on the security of puncturable PRF. In other words, it satisfies semantic security at punctured tag.

This formalization is different from that proposed by Bitansky and Vaikuntanathan. Nevertheless, our formalization of puncturable SKFE is sufficient for constructing IO. In fact, when we construct IO, we set the tag same as the message to be encrypted itself. Then, our formalization is conceptually the same as that of Bitansky and Vaikuntanathan. Our tag-based definition is well-suited for our constructions.

Achieving Weak-Succinctness via Collusion-Succinctness

We cannot directly use the puncturable SKFE scheme above as a building block of IO since it is non-succinct. We need to transform it into an weakly-succinct scheme while preserving security and functionality.

We accomplish this transformation via a *collusion-succinct* scheme. Collusion-succinctness requires that each size of the encryption circuit and punctured encryption circuit is sub-linear in the number of functional keys that the scheme can issue. Note that when we consider collusion-succinctness, the size of these circuits can be polynomial of the size of functions. We first show that we can construct collusion-succinct puncturable SKFE based on single-key non-succinct puncturable SKFE constructed above and collusion-resistant SKFE. Then, we transform the collusion-succinct scheme into an weakly-succinct scheme via a transformation based on decomposable randomized encoding. The transformation is similar to that proposed by Bitansky and Vaikuntanathan [BV15]. We give an illustration of our construction path in Figure 1.

There is a technical hurdle in the former transformation while we can accomplish the latter based on a known technique. We show the overview of the former transformation and explain the technical hurdle.

Construction of collusion-succinct scheme. Our goal of this step is to construct a collusion-succinct scheme, that is, a scheme which supports q functional keys and the size of whose encryption and punctured encryption circuits are sub-linear in q, where q is an a-priori fixed polynomial. The key tool for achieving this goal is strong exponentially-efficient IO (SXIO) proposed by Lin, Pass, Seth, and Telang [LPST16].

SXIO is a relaxed variant of IO. SXIO is required that, given a circuit C with n-bit input, it runs in $2^{\gamma n} \cdot \mathrm{poly}(\lambda, |C|)$-time, where γ is a constant smaller than 1, poly is some polynomial, and λ is the security parameter. We call γ the compression factor since it represents how SXIO can compress the truth table of the circuit to be obfuscated. SXIO with arbitrarily small constant compression factor can be constructed from collusion-resistant SKFE [BNPW16].

We show how to construct collusion-succinct puncturable SKFE from single-key non-succinct one and SXIO. To achieve a collusion-succinct scheme, we need to increase the number of functional keys to some polynomial q while compressing the size of its encryption circuits into sub-linear in q.

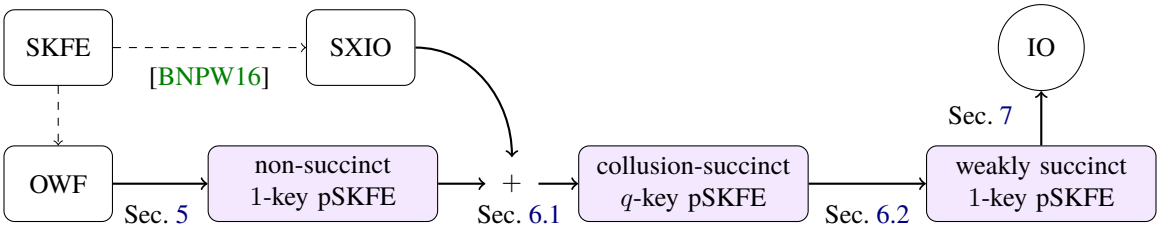

Figure 1: Illustration of our construction path. pSKFE denotes puncturable SKFE. Dashed lines denote known or trivial implications. White boxes denote our ingredients or goal. Purple boxes denote our core schemes. A transformation from an object in a rectangle to one in a rectangle incurs only polynomial security loss. A transformation from an object in a rectangle to one in a circle incurs super-polynomial security loss.

The most naive way to increase the number of functional keys is to run multiple instances of the single-key scheme. If we have q master secret keys $\mathsf{MSK}_1, \cdots, \mathsf{MSK}_q$, we can generate q functional keys since we can generate one functional key under each master secret key. In this case, to ensure that we can decrypt a ciphertext using every functional key under different master secret keys MSK_i for every $i \in [q]$, a ciphertext should be composed of q ciphertexts each of which is generated under MSK_i for every $i \in [q]$. In addition, when we generate a punctured master secret key punctured at tag, we generate q punctured master secret keys $\mathsf{MSK}_i^*\{\mathsf{tag}\}$ for every $i \in [q]$ all of which are punctured at tag.

In the naive construction above, we see that if the single-key scheme satisfies functionality preserving under puncturing and semantic security at punctured tag, then so does the resulting scheme since a ciphertext of the resulting scheme consists of only ciphertexts of the single-key scheme. However, if a ciphertext of the resulting scheme consists of q ciphertexts of the single-key scheme, the encryption time is obviously at least linear in q. Therefore, we cannot construct a collusion-succinct scheme based on this naive idea.

We then consider to compress the encryption time by using SXIO. We extend the technique used by Lin et al. [LPST16] and Bitansky et al. [BNPW16]. Let $\mathsf{sxi}\mathcal{O}$ be SXIO. We set a ciphertext as a circuit computing q ciphertexts obfuscated by $\mathsf{sxi}\mathcal{O}$ instead of setting it as q ciphertexts themselves. Concretely, we obfuscate the following circuit $\mathsf{E}_{\mathsf{1Key}}$ using $\mathsf{sxi}\mathcal{O}$. $\mathsf{E}_{\mathsf{1Key}}$ has hardwired message m, tag tag, and puncturable PRF key S, and on input $i \in [q]$, it first generates MSK_i pseudorandomly from S and i, and then outputs a ciphertext of m under MSK_i and tag. Note that the master secret key of this scheme is now one puncturable PRF key S. In other words,

Hard-Coded Constants: S, tag, m. // Description of (simplified) $\mathsf{E}_{\mathsf{1Key}}$
Input: $i \in [q]$

1. Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$.
2. Compute $\mathsf{MSK}_i \leftarrow \mathsf{Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
3. Return $\mathsf{CT}_i \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \mathsf{tag}, m)$.

the scheme generates q master secret keys of the single-key scheme from one puncturable PRF key. For the formal description of $\mathsf{E}_{\mathsf{1Key}}$, see Figure 2 in Section 6.1.

The size of $\mathsf{E}_{\mathsf{1Key}}$ is independent of q since $\mathsf{E}_{\mathsf{1Key}}$ consists of one PRF evaluation and setup and encryption procedure of the single-key scheme.[4] Therefore, the time needed to compute $\mathsf{sxi}\mathcal{O}(\mathsf{E}_{\mathsf{1Key}})$ is bounded by $2^{\gamma \log q} \cdot \mathrm{poly}(\lambda, |m|) = q^\gamma \cdot \mathrm{poly}(\lambda, |m|)$ for some constant $\gamma < 1$ and polynomial poly, that is, sub-linear in q. Namely, we succeeds in reducing the encryption time from linear to sub-linear in q.

However, we need more complicated structure to compress the running-time of a punctured encryption algorithm into sub-linear in q. The main reason is that we cannot give master secret key S in the clear in the punctured encryption circuit to reduce the security to that of the building block single-key scheme.

We first argue how to set a punctured master secret key. We cannot rely on the trivial way that sets q punctured master secret keys of the single-key scheme as a punctured master secret key since the size of the punctured encryption circuit becomes linear in q in this trivial way. Our solution is to set a punctured master secret key as also an obfuscated circuit under SXIO. More precisely, we obfuscate the following circuit $\mathsf{P}_{\mathsf{1Key}}$. $\mathsf{P}_{\mathsf{1Key}}$ has hardwired

[4] Strictly speaking, the domain of PRF is $[q]$, and thus the size of $\mathsf{E}_{\mathsf{1Key}}$ depends on q in logarithmic. However, it does not matter since logarithmic factor is absorbed by sub-linear factor. We ignore this issue here for simplicity.

tag tag and puncturable PRF key S. Note that S is the master secret key thus is the same puncturable PRF key as that hardwired into $\mathsf{E}_{\mathsf{1Key}}$. On input $i \in [q]$, $\mathsf{P}_{\mathsf{1Key}}$ first generates MSK_i pseudorandomly from S and i, and then outputs a punctured master secret key $\mathsf{MSK}_i^*\{\mathsf{tag}\}$ of the single-key scheme. For the formal description of $\mathsf{P}_{\mathsf{1Key}}$, see Figure 3 in Section 6.1.

// Description of (simplified) $\mathsf{P}_{\mathsf{1Key}}$	// Description of (simplified) $\mathsf{PE}_{\mathsf{1Key}}$
Hard-Coded Constants: S, tag. **Input:** $i \in [q]$ 1. Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$. 2. Compute $\mathsf{MSK}_i \leftarrow \mathsf{Setup}(1^\lambda; r^i_{\mathsf{Setup}})$. 3. Return $\mathsf{MSK}_i^*\{\mathsf{tag}\} \leftarrow \mathsf{Punc}(\mathsf{MSK}_i, \mathsf{tag})$.	**Hard-Coded Constants**: $\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{tag}', m$. **Input:** $i \in [q]$ 1. Parse $\mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}}) \leftarrow \mathsf{MSK}^*\{\mathsf{tag}\}$. 2. Compute $\mathsf{MSK}_i^*\{\mathsf{tag}\} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}})(i)$. 3. Return $\mathsf{CT}_i \leftarrow \mathsf{PEnc}(\mathsf{MSK}_i^*\{\mathsf{tag}\}, \mathsf{tag}', m)$.

In addition, we define the punctured encryption algorithm as follows. On input $\mathsf{MSK}^*\{\mathsf{tag}\}$ that is $\mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}})$, tag tag', and message m, the punctured encryption algorithm obfuscates the following circuit $\mathsf{PE}_{\mathsf{1Key}}$ using $\mathsf{sxi}\mathcal{O}$ and outputs the obfuscated circuit. $\mathsf{PE}_{\mathsf{1Key}}$ has hardwired $\mathsf{MSK}^*\{\mathsf{tag}\}$, tag', and m, and on input $i \in [q]$, it first generates the i-th punctured key $\mathsf{MSK}_i^*\{\mathsf{tag}\}$ by feeding i into $\mathsf{MSK}^*\{\mathsf{tag}\} = \mathsf{sxi}\mathcal{O}(\mathsf{PE}_{\mathsf{1Key}})$, and then outputs a ciphertext of m under $\mathsf{MSK}_i^*\{\mathsf{tag}\}$ and tag' using the punctured encryption algorithm of the single-key scheme. If the compression factor of $\mathsf{sxi}\mathcal{O}$ is *sufficiently small*, we ensure that the running time of this punctured encryption algorithm is sub-linear in q. For the formal description of $\mathsf{PE}_{\mathsf{1Key}}$, see Figure 4 in Section 6.1.

We can prove the semantic security at punctured tag by the punctured programming technique proposed by Sahai and Waters [SW14]. However, the construction above does not satisfy functionality preserving under puncturing. This is because ciphertexts output by the encryption and punctured encryption algorithms are different. The ciphertexts are obfuscation of different circuits $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$.

In fact, it seems difficult to avoid this problem as long as we use SXIO to gain succinctness. To the best of our knowledge, how to achieve succinctness in a generic way without using SXIO is not known.

Indistinguishability of functionality under puncturing. To overcome the problem above, we introduce a relaxed variant functionality preserving property that is compatible with the construction based on SXIO. We call it *indistinguishability of functionality under puncturing*. Informally speaking, the property requires that

$$(\mathsf{MSK}, \mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{Enc}(\mathsf{MSK}, \mathsf{tag}', m)) \overset{\mathsf{c}}{\approx} (\mathsf{MSK}, \mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{tag}', m))$$

holds for any tag tag and tag' such that $\mathsf{tag} \neq \mathsf{tag}'$, and message m, where $\overset{\mathsf{c}}{\approx}$ denotes computational indistinguishability. In other words, it requires that no distinguisher can distinguish ciphertexts output by Enc and PEnc even given both the master secret key and punctured master secret key.

We see that the collusion-succinct construction based on SXIO above satisfies indistinguishability of functionality under puncturing. This comes from the security guarantee of SXIO and the fact that $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$ are functionally equivalent as long as the above tag and tag' are different.

Overall, we can construct collusion-succinct puncturable SKFE with indistinguishability of functionality under puncturing from a single-key non-succinct scheme and SXIO.

Transforming into an weakly-succinct scheme. As stated earlier, we can in turn transform a collusion-succinct scheme into an weakly-succinct one based on the transformation using decomposable randomized encoding proposed by Bitansky and Vaikuntanathan [BV15]. In this transformation, a ciphertext of the weakly-succinct scheme is a ciphertext of the collusion-succinct scheme itself. Thus, if the collusion-succinct scheme satisfies semantic security at punctured tag and indistinguishability of functionality under puncturing, then so does the weakly-succinct scheme. Therefore, we can construct a single-key weakly-succinct puncturable SKFE with indistinguishability of functionality under puncturing.

Indistinguishability of functionality under puncturing looks to be insufficient for constructing IO. Nevertheless, we show that we can replace PKFE in the construction of IO proposed by Bitansky and Vaikuntanathan with our puncturable SKFE that satisfies only indistinguishability of functionality under puncturing if we allow more but asymptotically the same security loss.

2.4 IO from Puncturable SKFE

Finally, we give an overview of our IO construction below.

The construction of IO based on puncturable SKFE is almost the same as that based on PKFE proposed by Bitansky and Vaikuntanathan [BV15]. It does not depend on which functionality preserving property puncturable SKFE satisfies. Recall that, in their construction, a key pair $(\mathsf{PK}_i, \mathsf{MSK}_i)$ of PKFE is generated and the circuit E_{i-1} that has hardwired PK_i is obfuscated at every recursive step. In our construction based on puncturable SKFE, a master secret key MSK_i of puncturable SKFE is generated and E_{i-1} that has hardwired MSK_i is obfuscated at each recursive step. Concretely, we construct E_{i-1} as a circuit that has hardwired MSK_i and a SKE key K, and on $(i-1)$-bit input $\boldsymbol{x}_{i-1}$, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K)$ for $x_i \in \{0,1\}$ under MSK_i and a tag $\boldsymbol{x}_{i-1}$, that is, $\mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K))$ for $x_i \in \{0,1\}$. In the proof, we replace MSK_i hardwired into E_{i-1} with the tuple of a punctured master secret key $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ punctured at $\boldsymbol{j} \in \{0,1\}^{i-1}$ and a ciphertext of $(\boldsymbol{j}\|x_i, K)$ for $x_i \in \{0,1\}$, where $\boldsymbol{j}$ is a string in $\{0,1\}^{i-1}$ that we focus on at that time.

Outline of Security Proof

We give an overview of the security proof of IO based on puncturable SKFE. If the building block puncturable SKFE satisfies functionality preserving under puncturing, the security proof is almost the same as that of Bitansky and Vaikuntanathan. However, our puncturable SKFE satisfies only indistinguishability of functionality under puncturing, and thus we need more complicated arguments. The first half of the following overview is similar to that of Bitansky and Vaikuntanathan. The rest is an overview of proofs that we additionally need due to indistinguishability of functionality under puncturing.

Analogous to IO based on PKFE, we can accomplish this proof recursively. More precisely, we can prove the security of $i\mathcal{O}_i$ based on those of $i\mathcal{O}_{i-1}$, puncturable SKFE, and plain SKE. We proceed the proof as follows. Note again that, we ignore the issue of the randomness for the encryption algorithm and punctured encryption algorithm for simplicity. It is generated by puncturable PRF in the actual construction.

Suppose that we have two functionally equivalent circuits C_0 and C_1 both of which expect an i-bit input. We show that no efficient distinguisher $\mathcal{D}$ can distinguish $i\mathcal{O}_i(C_0)$ and $i\mathcal{O}_i(C_1)$. We consider the following sequence of hybrid experiments. Below, for two hybrids $\mathcal{H}$ and $\mathcal{H}'$, we write $\mathcal{H} \sim \mathcal{H}'$ to denote that the behavior of $\mathcal{D}$ does not change between $\mathcal{H}$ and $\mathcal{H}'$.

In the first hybrid $\mathcal{H}_0$, $\mathcal{D}$ is given $i\mathcal{O}_i(C_0)$. Recall that $i\mathcal{O}_i(C_0)$ consists of sk_{C^*} and $i\mathcal{O}_{i-1}(\mathsf{E}_{i-1})$. C^* has hardwired two SKE ciphertexts $\mathsf{CT}_0^{\mathsf{ske}}$ and $\mathsf{CT}_1^{\mathsf{ske}}$ of C_0 under independent keys K_0 and K_1. On i-bit input $\boldsymbol{x}_i$ and SKE key K_b, C^* first obtains C by decrypting $\mathsf{CT}_b^{\mathsf{ske}}$ by K_b and outputs $C(\boldsymbol{x}_i)$.

In the next hybrid $\mathcal{H}_1$, we change how $\mathsf{CT}_1^{\mathsf{ske}}$ hardwired in C^* is generated. Concretely, we generate $\mathsf{CT}_1^{\mathsf{ske}}$ as a ciphertext of C_1 under the key K_1. It holds that $\mathcal{H}_0 \sim \mathcal{H}_1$ due to the security of SKE. Then, in the next hybrid $\mathcal{H}_2$, we change the circuit E_{i-1} so that, on $(i-1)$-bit input $\boldsymbol{x}_{i-1}$, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K_1)$ instead of $(\boldsymbol{x}_{i-1}\|x_i, K_0)$ for $x_i \in \{0,1\}$ under MSK_i and a tag $\boldsymbol{x}_{i-1}$.

If we prove $\mathcal{H}_1 \sim \mathcal{H}_2$, we also prove $\mathcal{H}_0 \sim \mathcal{H}_2$ and almost complete the security proof. This is because we can argue that the behavior of $\mathcal{D}$ does not change between $\mathcal{H}_2$ and the hybrid where $\mathcal{D}$ is given $i\mathcal{O}_i(C_1)$ by a similar argument for $\mathcal{H}_0 \sim \mathcal{H}_2$.

Therefore, the main part of the proof is how we change the circuit E_{i-1} from encrypting K_0 in $\mathcal{H}_1$ to encrypting K_1 in $\mathcal{H}_2$. As mentioned earlier, we accomplish this task by relying on the argument of probabilistic IO formalized by Canneti et al. [CLTV15].

Concretely, we consider $2^{i-1}+1$ intermediate hybrid experiments $\mathcal{H}_{1,j}$ for $j \in \{0, \cdots, 2^{i-1}\}$ between $\mathcal{H}_1$ and $\mathcal{H}_2$. Between $\mathcal{H}_{1,j}$ and $\mathcal{H}_{1,j+1}$, we change E_{i-1} so that on input $\boldsymbol{j} \in \{0,1\}^{i-1}$, it outputs ciphertexts of $(\boldsymbol{j}\|x_i, K_1)$ instead of $(\boldsymbol{j}\|x_i, K_0)$ for $x_i \in \{0,1\}$, where $\boldsymbol{j}$ is the binary representation of j. More precisely, we construct E_{i-1} in $\mathcal{H}_{1,j}$ as follows. E_{i-1} has hardwired MSK_i, K_0, and K_1. On $(i-1)$-bit input $\boldsymbol{x}_{i-1}$,

- if $\boldsymbol{x}_{i-1} < \boldsymbol{j}$, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K_1)$ for $x_i \in \{0,1\}$ under MSK_i and a tag $\boldsymbol{x}_{i-1}$.
- Otherwise, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K_0)$ for $x_i \in \{0,1\}$ under MSK_i and a tag $\boldsymbol{x}_{i-1}$.

We see that E_{i-1} in $\mathcal{H}_1$ has the same functionality as E_{i-1} in $\mathcal{H}_{1,0}$. In addition, E_{i-1} in $\mathcal{H}_2$ has the same functionality as E_{i-1} in $\mathcal{H}_{1,2^{i-1}}$. Therefore, we have $\mathcal{H}_1 \sim \mathcal{H}_{1,0}$ and $\mathcal{H}_2 \sim \mathcal{H}_{1,2^{i-1}}$ from the security guarantee of $i\mathcal{O}_{i-1}$.

We show how to prove $\mathcal{H}_{1,j} \sim \mathcal{H}_{1,j+1}$. For simplicity, we first assume that puncturable SKFE satisfies functionality preserving under puncturing. In this case, we show $\mathcal{H}_{1,j} \sim \mathcal{H}_{1,j+1}$ by the following three steps.

(1) In the first step, we hardwire ciphertexts of $(\boldsymbol{j}\|x_i, K_0)$ under MSK_i and a tag $\boldsymbol{j}$ for $x_i \in \{0,1\}$ in E_{i-1}. In addition, we replace hardwired MSK_i in E_{i-1} with $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ that is a master secret key punctured at a tag $\boldsymbol{j}$. On $(i-1)$-bit input $\boldsymbol{x}_{i-1}$,

- if $\boldsymbol{x}_{i-1} = \boldsymbol{j}$, E_{i-1} outputs hardwired ciphertexts of $(\boldsymbol{j}\|x_i, K_0)$ for $x_i \in \{0,1\}$.
- if $\boldsymbol{x}_{i-1} \neq \boldsymbol{j}$, it generates ciphertexts of $(\boldsymbol{x}_{i-1}\|x_i, K_\beta)$ under $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ and a tag $\boldsymbol{x}_{i-1}$ and outputs them, where $\beta = 1$ if $\boldsymbol{x}_{i-1} < \boldsymbol{j}$ and $\beta = 0$ otherwise.

We see that this change does not affect the functionality of E_{i-1} if puncturable SKFE satisfies functionality preserving under puncturing. Thus, this step is done by the security of $i\mathcal{O}_{i-1}$.

(2) In the second step, we change the hardwired ciphertexts to ciphertexts of $(\boldsymbol{j}\|x_i, K_1)$ for $x_i \in \{0,1\}$. This is done by the semantic security at punctured tag of puncturable SKFE.

(3) In the final step, we change E_{i-1} so that it does not have hardwired ciphertexts of $(\boldsymbol{j}\|x_i, K_1)$ for $x_i \in \{0,1\}$. Moreover, we change E_{i-1} so that E_{i-1} has hardwired MSK_i and use it to generate the output ciphertexts. This change also does not affect the functionality of E_{i-1}, and thus we can accomplish this step by relying on the security of $i\mathcal{O}_{i-1}$ again.

From the above, if puncturable SKFE satisfies functionality preserving under puncturing, we have $\mathcal{H}_{1,j} \sim \mathcal{H}_{1,j+1}$ for every $j \in \{0, \cdots, 2^{i-1}-1\}$. By combining $\mathcal{H}_1 \sim \mathcal{H}_{1,0}$ and $\mathcal{H}_{1,2^{i-1}} \sim \mathcal{H}_2$, we obtain $\mathcal{H}_1 \sim \mathcal{H}_2$.

Therefore, we complete the entire proof. In fact, in this case, the proof is essentially the same as that for the case where PKFE is used as a building block shown by Bitansky and Vaikuntanathan.

Additional hybrids for the case of indistinguishability of functionality under puncturing. Recall that our puncturable SKFE satisfies only indistinguishability of functionality under puncturing. Thus, the above argument for steps 1 and 3 do not work straightforwardly. This is because if puncturable SKFE satisfies only indistinguishability of functionality under puncturing, the functionality of E_{i-1} might change at each step of 1 and 3. Therefore, we cannot directly use the security of $i\mathcal{O}_{i-1}$.

Nevertheless, even if puncturable SKFE satisfies only indistinguishability of functionality under puncturing, we can proceed steps 1 and 3 by introducing more additional hybrids. Since steps 1 and 3 are symmetric, we focus on proceeding the step 1. We can apply the following argument for the step 3. Below, we let $\mathcal{H}^0_{1,j}$ denote the hybrid experiment after applying the step 1 to $\mathcal{H}_{1,j}$.

To accomplish the step 1, we introduce the additional intermediate hybrids $\mathcal{H}_{1,j,k}$ for every $k \in \{0, \cdots, 2^{i-1}\} \setminus \{j\}$ between $\mathcal{H}_{1,j}$ and $\mathcal{H}^0_{1,j}$. Between $\mathcal{H}_{1,j,k}$ and $\mathcal{H}_{1,j,k+1}$, we change E_{i-1} so that, on input $\boldsymbol{k} \in \{0,1\}^{i-1}$, it outputs ciphertexts under $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ instead of ciphertexts under MSK, where $\boldsymbol{k}$ is the binary representation of k. More precisely, we construct E_{i-1} in $\mathcal{H}_{1,j,k}$ as follows. E_{i-1} has hardwired $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ in addition to MSK_i, K_0, and K_1. On $(i-1)$-bit input $\boldsymbol{x}_{i-1}$, it runs as follows.

- If $\boldsymbol{x}_{i-1} < \boldsymbol{j}$, it sets $\beta = 1$ and $\beta = 0$ otherwise.
- If $\boldsymbol{x}_{i-1} < k$ and $\boldsymbol{x}_{i-1} \neq j$, for $x_i \in \{0,1\}$, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K_\beta)$ under $\mathsf{MSK}_{i-1}^*\{\boldsymbol{j}\}$ and a tag $\boldsymbol{x}_{i-1}$, that is, $\mathsf{PEnc}(\mathsf{MSK}_i^*\{\boldsymbol{j}\}, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_\beta))$.
- Otherwise ($\boldsymbol{x}_{i-1} \geq k$ or $\boldsymbol{x}_{i-1} = j$), for $x_i \in \{0,1\}$, it outputs a ciphertext of $(\boldsymbol{x}_{i-1}\|x_i, K_\beta)$ under MSK_i and tag $\boldsymbol{x}_{i-1}$, that is, $\mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_\beta))$.

We see that E_{i-1} in $\mathcal{H}_{1,j}$ and $\mathcal{H}^0_{1,j}$ have the same functionality as that in $\mathcal{H}_{1,j,0}$ and $\mathcal{H}_{1,j,2^{i-1}}$, respectively. In addition, E_{i-1} in $\mathcal{H}_{1,j,j}$ has the same functionality as that in $\mathcal{H}_{1,j,j+1}$. Therefore, we have $\mathcal{H}_{1,j} \sim \mathcal{H}_{1,j,0}$, $\mathcal{H}^0_{1,j} \sim \mathcal{H}_{1,j,2^{i-1}}$, and $\mathcal{H}_{1,j,j} \sim \mathcal{H}_{1,j,j+1}$ from the security guarantee of $i\mathcal{O}_{i-1}$.

We can prove $\mathcal{H}_{1,j,k} \sim \mathcal{H}_{1,j,k+1}$ for every $k \in \{0, \cdots, 2^{i-1}\} \setminus \{j\}$ by three steps again based on indistinguishability of functionality under puncturing.

(1) We hardwire ciphertexts of $(\boldsymbol{k}\|x_i, K_\beta)$ under MSK_i and a tag $\boldsymbol{k}$, that is, $\mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{k}, (\boldsymbol{k}\|x_i, K_\beta))$ for $x_i \in \{0,1\}$ in E_{i-1} in the first step. In addition, we change E_{i-1} so that it outputs the hardwired ciphertext of $(\boldsymbol{k}\|x_i, K_0)$ for $x_i \in \{0,1\}$ if the input is $\boldsymbol{k}$. We see that this change does not affect the functionality of E_{i-1}. Thus, this step is done by the security of $i\mathcal{O}_{i-1}$.

(2) In the second step, we change the hardwired ciphertexts to a ciphertext of $(\boldsymbol{k}\|x_i, K_\beta)$ under $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$, that is $\mathsf{PEnc}(\mathsf{MSK}_i\{\boldsymbol{j}\}, \boldsymbol{k}, (\boldsymbol{k}\|x_i, K_\beta))$ for $x_i \in \{0,1\}$. This is done by the indistinguishability of functionality under puncturing of puncturable SKFE.

(3) In the final step, we change E_{i-1} so that it does not have a hardwired ciphertext of $(\boldsymbol{k}\|x_i, K_1)$ for $x_i \in \{0,1\}$. Namely, we change E_{i-1} so that on input $\boldsymbol{k}$, E_{i-1} generates ciphertexts of $\boldsymbol{k}$ under $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ and outputs them. This change does not affect the functionality of E_{i-1}, and thus we can accomplish this step by relying on the security of $i\mathcal{O}_{i-1}$ again.

From the above, we see that $\mathcal{H}_{1,j,k} \sim \mathcal{H}_{1,j,k+1}$ holds for every $k \in \{0, \cdots, 2^{i-1}\} \setminus \{j\}$. By combining $\mathcal{H}_{1,j} \sim \mathcal{H}_{1,j,0}$, $\mathcal{H}^0_{1,j} \sim \mathcal{H}_{1,j,2^{i-1}}$, and $\mathcal{H}_{1,j,j} \sim \mathcal{H}_{1,j,j+1}$, we obtain $\mathcal{H}_{1,j} \sim \mathcal{H}^0_{1,j}$.

Therefore, we obtain $\mathcal{H}_{1,j} \sim \mathcal{H}^0_{1,j}$ even if puncturable SKFE satisfies only indistinguishability of functionality under puncturing. Overall, we can complete the entire security proof.

We note that our security proof incurs more security loss than those of Bitansky and Vaikuntanathan [BV15] and the case where puncturable SKFE satisfies functionality preserving under puncturing. Our security proof incurs roughly $2^{2\cdot i}$ security loss while the latter proofs incurs 2^i security loss when we prove the security of $i\mathcal{O}_i$ based on that of $i\mathcal{O}_{i-1}$. Nevertheless, this difference is not an issue in the sense that if the building block primitives are roughly $2^{\Omega(n^2)}$-secure, we can prove the security of our indistinguishability obfuscator. This requirement is the same as that of Bitansky and Vaikuntanathan.

3 Preliminaries

We define some notations and cryptographic primitives here.

3.1 Notations

We write $x \xleftarrow{\mathsf{r}} X$ to denote that an element x is chosen from a finite set X uniformly at random and $y \leftarrow \mathsf{A}(x; r)$ to denote that the output of an algorithm A on an input x and a randomness r is assigned to y. When there is no need to write the randomness explicitly, we omit it and simply write $y \leftarrow \mathsf{A}(x)$. For strings x and y, $x\|y$ denotes the concatenation of x and y. Throughout this paper, λ denotes a security parameter. A function $f(\lambda)$ is a negligible function if $f(\lambda)$ tends to 0 faster than $\frac{1}{\lambda^c}$ for every constant $c > 0$. We write $f(\lambda) = \mathsf{negl}(\lambda)$ to denote that $f(\lambda)$ is a negligible function. PPT stands for probabilistic polynomial time. Let $[\ell]$ denote the set of integers $\{1, \cdots, \ell\}$.

3.2 Standard Cryptographic Tools

In this section, we review standard cryptographic tools, pseudorandom function (PRF), puncturable PRF, secret-key encryption (SKE), garbling scheme, and decomposable randomized encoding.

Definition 3.1 (Pseudorandom functions). *Sets $\mathcal{D}$ and $\mathcal{R}$, let $\{\mathsf{F}_S(\cdot) : \mathcal{D} \to \mathcal{R} | S \in \{0,1\}^\lambda\}$ be a family of polynomially computable functions. We say that F is pseudorandom if for any PPT adversary $\mathcal{A}$, it holds that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{F},\mathcal{A}}(\lambda) \quad = \quad & |\Pr[\mathcal{A}^{\mathsf{F}_S(\cdot)}(1^\lambda) = 1 : S \xleftarrow{\mathsf{r}} \{0,1\}^\lambda] \\ & - \Pr[\mathcal{A}^{\mathsf{R}(\cdot)}(1^\lambda) = 1 : \mathsf{R} \xleftarrow{\mathsf{r}} \mathcal{U}]| = \mathsf{negl}(\lambda) \ ,\end{aligned}$$

where $\mathcal{U}$ is the set of all functions from $\mathcal{D}$ to $\mathcal{R}$. Moreover, for some concrete negligible function $\epsilon(\cdot)$, we say that F is ϵ-secure if for any PPT $\mathcal{A}$ the above indistinguishability gap is smaller than $\epsilon(\lambda)^{\Omega(1)}$.

Theorem 3.2 ([GGM86]). *If one-way functions exist, then for all efficiently computable functions $n(\lambda)$ and $m(\lambda)$, there exists a pseudorandom function that maps $n(\lambda)$ bits to $m(\lambda)$ bits (i.e., $\mathcal{D} := \{0,1\}^{n(\lambda)}$ and $\mathcal{R} := \{0,1\}^{m(\lambda)}$).*

Definition 3.3 (Puncturable pseudorandom function). *For sets $\mathcal{D}$ and $\mathcal{R}$, a puncturable pseudorandom function* PPRF *consists of a tuple of algorithms* $(\mathsf{F}, \mathsf{Punc})$ *that satisfies the following two conditions.*

Functionality preserving under puncturing: *For all polynomial size subset $\{x_i\}_{i\in[k]}$ of $\mathcal{D}$, and for all $x \in \mathcal{D} \setminus \{x_i\}_{i\in[k]}$, we have $\Pr[\mathsf{F}_S(x) = \mathsf{F}_{S^*}(x) : S \leftarrow \{0,1\}^\lambda, S^* \leftarrow \mathsf{Punc}(S, \{x_i\}_{i\in[k]})] = 1$.*

Pseudorandomness at punctured points: *For all polynomial size subset $\{x_i\}_{i\in[k]}$ of $\mathcal{D}$, and any PPT adversary $\mathcal{A}$, it holds that*

$$\Pr[\mathcal{A}(S^*, \{\mathsf{F}_S(x_i)\}_{i\in[k]}) = 1] - \Pr[\mathcal{A}(S^*, U^k) = 1] = \mathsf{negl}(\lambda) \ ,$$

where $S \xleftarrow{r} \{0,1\}^\lambda$, $S^* \leftarrow \mathsf{Punc}(S, \{x_i\}_{i\in[k]})$, and U denotes the uniform distribution over $\mathcal{R}$.

Moreover, for some concrete negligible function $\epsilon(\cdot)$, we say that PPRF is ϵ-secure if for any $\mathcal{A}$ the above indistinguishability gap is smaller than $\epsilon(\lambda)^{\Omega(1)}$.

Theorem 3.4 ([GGM86, BW13, BGI14, KPTZ13]). *If one-way functions exist, then for all efficiently computable functions $n(\lambda)$ and $m(\lambda)$, there exists a puncturable pseudorandom function that maps $n(\lambda)$ bits to $m(\lambda)$ bits (i.e., $\mathcal{D} := \{0,1\}^{n(\lambda)}$ and $\mathcal{R} := \{0,1\}^{m(\lambda)}$).*

Definition 3.5 (Secret key encryption). *An SKE scheme SKE is a two tuple (E, D) of PPT algorithms.*

- *The encryption algorithm E, given a key $K \in \{0,1\}^\lambda$ and a message $m \in \mathcal{M}$, outputs a ciphertext c, where $\mathcal{M}$ is the plaintext space of SKE.*
- *The decryption algorithm D, given a key K and a ciphertext c, outputs a message $\tilde{m} \in \{\perp\} \cup \mathcal{M}$. This algorithm is deterministic.*

Correctness: *We require $\mathsf{D}(K, \mathsf{E}(K, m)) = m$ for every $m \in \mathcal{M}$ and key $K \in \{0,1\}^\lambda$.*

CPA security: *We define the security game between a challenger and an adversary $\mathcal{A}$ as follows.*

1. *The challenger generates $K \xleftarrow{r} \{0,1\}^\lambda$ and chooses the challenge bit $b \xleftarrow{r} \{0,1\}$. Then, the challenger sends 1^λ to $\mathcal{A}$.*
2. *$\mathcal{A}$ may make polynomially many encryption queries adaptively. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$ to the challenger. Then, the challenger returns $c \leftarrow \mathsf{E}(K, m_b)$.*
3. *$\mathcal{A}$ outputs $b' \in \{0,1\}$.*

In this game, we define the advantage of the adversary $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{cpa}}_{\mathsf{SKE},\mathcal{A}}(\lambda) = 2|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \ .$$

For a negligible function $\epsilon(\cdot)$, We say that SKE is ϵ-secure if for any PPT $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{cpa}}_{\mathsf{SKE},\mathcal{A}}(\lambda) < \epsilon(\lambda)^{\Omega(1)}$.

Theorem 3.6 ([LR88]). *If there exists one-way functions, there exists a CPA-secure SKE scheme.*

Definition 3.7 (Garbling scheme). *Let $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ be a family of circuits where each circuit in $\mathcal{C}_n$ takes an n-bit input. A circuit garbling scheme GC is a two tuple $(\mathsf{Grbl}, \mathsf{Eval})$ of PPT algorithms.*

- *The garbling algorithm Grbl, given a security parameter 1^λ and a circuit $C \in \mathcal{C}_n$, outputs a garbled circuit $\widetilde{C}$, together with $2n$ labels $\{L_{j,\alpha}\}_{j\in[n],\alpha\in\{0,1\}}$.*
- *The evaluation algorithm, given a garbled circuit $\widetilde{C}$ and n labels $\{L_j\}_{j\in[n]}$, outputs y.*

Correctness: *We require $\mathsf{Eval}(\widetilde{C}, \{L_{j,x_j}\}_{j\in[n]}) = C(x)$ for every $n \in \mathbb{N}$, $C \in \mathcal{C}_n$, and $x \in \{0,1\}^n$, where $(\widetilde{C}, \{L_{j,\alpha}\}_{j\in[n],\alpha\in\{0,1\}}) \leftarrow \mathsf{Grbl}(1^\lambda, C)$ and x_j is the j-th bit of x for every $j \in [n]$.*

Security: *Let Sim be a PPT simulator. We define the following game between a challenger and an adversary $\mathcal{A}$ as follows.*

1. *The challenger chooses the challenge bit $b \xleftarrow{r} \{0,1\}$ and sends security parameter 1^λ to $\mathcal{A}$.*
2. *$\mathcal{A}$ sends a circuit $C \in \mathcal{C}_n$ and an input $x \in \{0,1\}^n$ for the challenger.*
3. *If $b = 0$, the challenger computes $(\widetilde{C}, \{L_{j,\alpha}\}_{j\in[n],\alpha\in\{0,1\}}) \leftarrow \mathsf{Grbl}(1^\lambda, C)$ and returns $(\widetilde{C}, \{L_{j,x_j}\}_{j\in[n]})$ to $\mathcal{A}$. Otherwise, the challenger returns $(\widetilde{C}, \{L_j\}_{j\in[n]}) \leftarrow \mathsf{Sim}(1^\lambda, |C|, C(x))$.*
4. *$\mathcal{A}$ outputs $b' \in \{0,1\}$.*

In this game, we define the advantage of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{gc}}_{\mathsf{GC},\mathcal{A},\mathsf{Sim}}(\lambda) = 2|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \ .$$

For a concrete negligible function $\epsilon(\cdot)$, We say that GC is ϵ-secure if there exists a PPT Sim such that for any PPT $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{gc}}_{\mathsf{GC},\mathcal{A},\mathsf{Sim}}(\lambda) < \epsilon(\lambda)^{\Omega(1)}$.

Theorem 3.8 ([Yao86, BHR12, LP09]). *If there exists one-way functions, there exists a secure garbling scheme for any polynomial size circuits.*

Definition 3.9 (Decomposable randomized encoding). *Let $c \geq 1$ be an integer constant. A c-local decomposable randomized encoding RE, given security parameter 1^λ and a function f of size s and n-bit input, outputs a function $\widehat{f} : \{0,1\}^n \times \{0,1\}^\rho \to \{0,1\}^\mu$ with the following properties. ρ and μ are polynomials bounded by $s \cdot \mathrm{poly}_{\mathsf{RE}}(\lambda, n)$, where $\mathrm{poly}_{\mathsf{RE}}$ is a fixed polynomial.*

Correctness: *There is a polynomial time decoder that, given $\widehat{f}(x;r)$, outputs $f(x)$ for any $x \in \{0,1\}^n$ and $r \in \{0,1\}^\rho$.*

Decomposability: *Computation of $\widehat{f}$ can be decomposed into computation of μ functions. That is, there exist μ functions $\widehat{f}_1, \cdots, \widehat{f}_\mu$ such that $\widehat{f}(x;r) = (\widehat{f}_1(x;r), \cdots, \widehat{f}_\mu(x;r))$. Each $\widehat{f}_i$ depends on a single bit of x at most and c bits of r. We write $\widehat{f}(x;r) = (\widehat{f}_1(x;r_{S_1}), \cdots, \widehat{f}_\mu(x;r_{S_\mu}))$, where S_i denotes the subset of bits of r that $\widehat{f}_i$ depends on.*

Security: *Let Sim be a PPT simulator. We define the following game between a challenger and an adversary $\mathcal{A}$ as follows.*

1. *The challenger chooses a bit $b \xleftarrow{\mathsf{r}} \{0,1\}$ and sends security parameter 1^λ to $\mathcal{A}$.*
2. *$\mathcal{A}$ sends a function f of size s and n-bit input and an input $x \in \{0,1\}^n$ to the challenger.*
3. *If $b = 0$, the challenger computes $\widehat{f} \leftarrow \mathsf{RE}(1^\lambda, f)$, generates $r \leftarrow \{0,1\}^\rho$, and returns $\widehat{f}(x;r)$ to $\mathcal{A}$. Otherwise, the challenger returns $\mathsf{Sim}(1^\lambda, s, f(x))$.*
4. *$\mathcal{A}$ outputs $b' \in \{0,1\}$.*

In this game, we define the advantage of $\mathcal{A}$ as

$$\mathsf{Adv}^{\mathsf{re}}_{\mathsf{RE},\mathsf{Sim},\mathcal{A}}(\lambda) = |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \ .$$

For a negligible function $\epsilon(\cdot)$, we say that RE is ϵ-secure if there exists a PPT Sim such that for any PPT $\mathcal{A}$, we have $\mathsf{Adv}^{\mathsf{re}}_{\mathsf{RE},\mathsf{Sim},\mathcal{A}}(\lambda) < \epsilon(\lambda)^{\Omega(1)}$.

It is known that a decomposable randomized encoding can be based on one-way functions.

Theorem 3.10 ([Yao86, AIK06]). *If there exists one-way functions, there exists a secure decomposable randomized encoding for all polynomial size functions.*

3.3 Secret-Key Functional Encryption

We review the definition of ordinary secret-key functional encryption (SKFE).

Definition 3.11 (Secret-key functional encryption). *An SKFE scheme SKFE is a four tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec})$. Below, let $\mathcal{M}$ and $\mathcal{F}$ be the message space and function space of SKFE, respectively.*

- *The setup algorithm Setup, given a security parameter 1^λ, outputs a master secret key MSK.*
- *The key generation algorithm KG, given a master secret key MSK and a function $f \in \mathcal{F}$, outputs a functional decryption key sk_f.*
- *The encryption algorithm Enc, given a master secret key MSK and a message $m \in \mathcal{M}$, outputs a ciphertext CT.*

- *The decryption algorithm* Dec*, given a functional decryption key* sk_f *and a ciphertext* CT*, outputs a message* $\tilde{m} \in \{\bot\} \cup \mathcal{M}$.

Correctness: *We require* $\mathsf{Dec}(\mathsf{KG}(\mathsf{MSK}, f), \mathsf{Enc}(\mathsf{MSK}, m)) = f(m)$ *for every* $m \in \mathcal{M}$*,* $f \in \mathcal{F}$*, and* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$.

Next, we introduce selective-message message privacy for SKFE schemes.

Definition 3.12 (Selective-message message privacy). *Let* SKFE *be an SKFE scheme whose message space and function space are* $\mathcal{M}$ *and* $\mathcal{F}$*, respectively. Let* q *be a polynomial of* λ*. We define the selective-message message privacy game between a challenger and an adversary* $\mathcal{A}$ *as follows.*

1. *The challenger generates a master secret key* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$ *and chooses the challenge bit* $b \xleftarrow{\mathsf{r}} \{0,1\}$*. Then, the challenger sends security parameter* 1^λ *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *sends* $\{(m_0^\ell, m_1^\ell)\}_{\ell \in [p]}$ *to the challenger, where* p *is an a-priori unbounded polynomial of* λ*.*
3. *The challenger generates ciphertexts* $\mathsf{CT}^{(\ell)} \leftarrow \mathsf{Enc}(\mathsf{MSK}, m_b^\ell)(\ell \in [p])$ *and sends them to* $\mathcal{A}$*.*
4. $\mathcal{A}$ *may adaptively make key queries* q *times at most. For a key query* $f \in \mathcal{F}$ *from* $\mathcal{A}$*, the challenger generates* $sk_f \leftarrow \mathsf{KG}(\mathsf{MSK}, f)$*, and returns* sk_f *to* $\mathcal{A}$*. Here,* f *needs to satisfy* $f(m_0^\ell) = f(m_1^\ell)$ *for all* $\ell \in [p]$.
5. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$.

In this game, we define the advantage of $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{sm\text{-}mp}}_{\mathsf{SKFE},\mathcal{A}}(\lambda) = 2|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b' = 1 | b = 0] - \Pr[b' = 1 | b = 1]| \ .$$

$\mathcal{A}$ *is said to be valid if each function query* f *made by* $\mathcal{A}$ *satisfies that* $f(m_0^\ell) = f(m_1^\ell)$ *for all* $\ell \in [p]$ *in the above game. For a negligible function* $\epsilon(\cdot)$*, We say that* SKFE *is* (q, ϵ)*-selective-message message private if for any valid PPT* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{sm\text{-}mp}}_{\mathsf{SKFE},\mathcal{A}}(\lambda) < \epsilon(\lambda)^{\Omega(1)}$.

We further say that an SKFE scheme is ϵ-secure collusion-resistant SKFE if it is (q, ϵ)-selective-message message private for any polynomial q.

3.4 Indistinguishability Obfuscation

Definition 3.13 (Indistinguishability obfuscator (IO)). *A PPT algorithm* $i\mathcal{O}$ *is an indistinguishability obfuscator for a circuit class* $\{\mathcal{C}_\lambda\}_{\lambda \in \mathbb{N}}$ *if it satisfies the following two conditions.*

Functionality: *for all security parameters* $\lambda \in \mathbb{N}$*, for all* $C \in \mathcal{C}_\lambda$*, for all inputs* x*, we have that*

$$\Pr[C'(x) = C(x) : C' \leftarrow i\mathcal{O}(1^\lambda, C)] = 1 \ .$$

Indistinguishability: *for any poly-size distinguisher* $\mathcal{D}$*, there exists a negligible function* $\mathsf{negl}(\cdot)$ *such that the following holds: for all security parameters* $\lambda \in \mathbb{N}$*, for all pairs of circuits* $C_0, C_1 \in \mathcal{C}_\lambda$ *of the same size and such that* $C_0(x) = C_1(x)$ *for all inputs* x*, then*

$$|\Pr\left[\mathcal{D}(i\mathcal{O}(1^\lambda, C_0)) = 1\right] - \Pr\left[\mathcal{D}(i\mathcal{O}(1^\lambda, C_1)) = 1\right]| = \mathsf{negl}(\lambda) \ .$$

We further say that $i\mathcal{O}$ *is* ϵ*-secure, for some concrete negligible function* $\epsilon(\cdot)$*, if for any PPT distinguisher the above advantage is smaller than* $\epsilon(\lambda)^{\Omega(1)}$.

3.5 Strong Exponentially-Efficient Indistinguishability Obfuscation

Definition 3.14 (Strong Exponentially-Efficient Indistinguishability Obfuscation (SXIO)). *Let $\gamma < 1$ be a constant. A PPT algorithm $\mathsf{sxi}\mathcal{O}$ is a γ-compressing strong exponentially-efficient indistinguishability obfuscator (SXIO) for a circuit class $\{\mathcal{C}\}_{\lambda\in\mathbb{N}}$ if it satisfies the functionality and indistinguishability in Definition 3.13 and the following efficiency requirement:*

Non-trivial time efficiency *We require that the running time of $\mathsf{sxi}\mathcal{O}$ on input $(1^\lambda, C)$ is at most $2^{n\gamma}\cdot\mathrm{poly}(\lambda, |C|)$ for every $\lambda\in\mathbb{N}$ and circuit $C\in\{C_\lambda\}_{\lambda\in\mathbb{N}}$ with input length n.*

We have the following theorem.

Theorem 3.15 ([BNPW16]). *Assuming there exists ϵ-secure collusion-resistant SKFE for all circuits, where $\epsilon(\cdot)$ is a negligible function. Then, for any constant $\gamma < 1$, there exists ϵ-secure γ-compressing SXIO for polynomial-size circuits with logarithmic size input.*

4 Puncturable Secret-Key Functional Encryption

In this section, we introduce puncturable secret-key functional encryption (puncturable SKFE).

The notion of puncturable SKFE was introduced by Bitansky and Vaikuntanathan [BV15]. They showed that in their construction of IO, the building block PKFE can be replaced with puncturable SKFE. However, it has been open whether we can achieve puncturable SKFE without assuming the existence of PKFE.

In this work, we answer the question affirmatively. We show how to construct a relaxed variant of puncturable SKFE scheme that is single-key weakly-succinct. Our relaxed variant is sufficient for constructing IO. Our construction consists of two steps.

1. We prove that a single-key non-succinct puncturable SKFE scheme is constructed only from one-way functions.
2. We prove that we can transform the non-succinct scheme into a weakly-succinct scheme by using SXIO.

We can construct SXIO based on standard (i.e., not puncturable) SKFE by Theorem 3.15. Therefore, we can construct our puncturable SKFE from standard SKFE.

4.1 Syntax

Our definition of puncturable SKFE introduced below is slightly different from that proposed by Bitansky and Vaikuntanathan [BV15]. However, we show that puncturable SKFE defined in this paper is also a sufficient building block of IO. We state differences between our definition and theirs after describing the syntax and security of our puncturable SKFE.

Definition 4.1 (Puncturable secret-key functional encryption). *A puncturable SKFE scheme pSKFE is a tuple $(\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Punc}, \mathsf{PEnc})$ of six PPT algorithms. Below, let $\mathcal{M}$, $\mathcal{F}$, and $\mathcal{T}$ be the message space, function space, and tag space of pSKFE, respectively. In addition, let q be a polynomial denoting the upper bound of the number of issuable functional keys.*

- *The setup algorithm Setup, given a security parameter 1^λ, outputs a master secret key MSK.*
- *The key generation algorithm KG, given a master secret key MSK, function $f\in\mathcal{F}$, and an index $i\in[q]$, outputs a functional key sk_f.*
- *The encryption algorithm Enc, given a master secret key MSK, a tag tag, and a message $m\in\mathcal{M}$, outputs a ciphertext CT.*
- *The decryption algorithm Dec, given a functional key sk_f, a tag tag, and a ciphertext CT, outputs a message $\tilde{m}\in\{\bot\}\cup\mathcal{M}$.*
- *The puncturing algorithm Punc, given a master secret key MSK and a tag tag, outputs a punctured master secret key $\mathsf{MSK}^*\{\mathsf{tag}\}$*

- *The punctured encryption algorithm* PEnc*, given a punctured master secret key* MSK^**, a tag* tag'*, and a message* m*, outputs a ciphertext* CT.

We require the following property.

Correctness: *For every* $m \in \mathcal{M}$*,* $f \in \mathcal{F}$*,* $i \in [q]$*,* $\mathsf{tag} \in \mathcal{T}$*, and* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$*, it holds that*

$$\mathsf{Dec}(\mathsf{KG}(\mathsf{MSK}, f, i), \mathsf{tag}, \mathsf{Enc}(\mathsf{MSK}, \mathsf{tag}, m)) = f(m) \ .$$

4.2 Security

In this section, we introduce two variants of security. Their difference is the functionality of punctured encryption algorithms.

Definition 4.2 (Secure puncturable SKFE). *Let* $\mathsf{pSKFE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Punc}, \mathsf{PEnc})$ *be puncturable SKFE. Below, let* $\mathcal{M}$*,* $\mathcal{F}$*, and* $\mathcal{T}$ *be the message space, function space, and tag space of* pSKFE*, respectively. In addition, let* q *be a polynomial denoting the upper bound of the number of issuable functional keys. We say that* pSKFE *is secure puncturable SKFE if it satisfies the following properties.*

Functionality preserving under puncturing: *For every* $m \in \mathcal{M}$*,* $(\mathsf{tag}, \mathsf{tag}') \in \mathcal{T} \times \mathcal{T}$ *such that* $\mathsf{tag} \neq \mathsf{tag}'$*, randomness* r*,* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$*, and* $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{Punc}(\mathsf{MSK}, \mathsf{tag})$*, it holds that*

$$\mathsf{PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{tag}', m; r) = \mathsf{Enc}(\mathsf{MSK}, \mathsf{tag}', m; r) \ .$$

Semantic security at punctured tag: *We define punctured semantic security game between a challenger and an adversary* $\mathcal{A}$ *as follows.*

1. *The challenger generates a master secret key* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$ *and chooses a challenge bit* $b \xleftarrow{\mathsf{r}} \{0,1\}$*. The challenger sends security parameter* 1^λ *to* $\mathcal{A}$*.*
2. $\mathcal{A}$ *sends* $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$*,* $\mathsf{tag} \in \mathcal{T}$*, and* $\{f_i\}_{i\in[q]} \in \mathcal{F}^q$ *to the challenger. We require that for every* $i \in [q]$ *it holds that* $f_i(m_0) = f_i(m_1)$*.*
3. *The challenger computes* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MSK}, \mathsf{tag}, m_b)$*,* $\mathsf{sk}_{f_i} \leftarrow \mathsf{KG}(\mathsf{MSK}, f_i, i)$ *for every* $i \in [q]$*, and* $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{Punc}(\mathsf{MSK}, \mathsf{tag})$.

   Then, the challenger returns $(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{CT}, \{\mathsf{sk}_{f_i}\}_{i\in[q]})$ *to* $\mathcal{A}$*.*
4. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$*.*

In this game, we define the advantage of the adversary $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda) = 2|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \ .$$

$\mathcal{A}$ *is said to be valid if* $f_i(m_0) = f_i(m_1)$ *holds for every* $i \in [q]$ *in the above game. We say that* pSKFE *satisfies semantic security at punctured tag if for any valid PPT* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$*.*

We further say that pSKFE *satisfies* ϵ*-semantic security at punctured tag, for some concrete negligible function* $\epsilon(\cdot)$*, if for any valid PPT* $\mathcal{A}$ *the above advantage* $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda)$ *is smaller than* $\epsilon(\lambda)^{\Omega(1)}$*.*

In addition, we say that pSKFE *is* ϵ*-secure puncturable SKFE if it satisfies functionality preserving under puncturing and* ϵ*-semantic security at punctured tag.*

Instead of functionality preserving under puncturing, we can consider a relaxed variant which we call *indistinguishability of functionality under puncturing*. This property requires that any PPT distinguisher cannot distinguish ciphertexts output by Enc and PEnc even given both master secret key and punctured master secret key. The formal definition is as follows.

Definition 4.3 (Indistinguishability of functionality under puncturing). *Let* $\mathsf{pSKFE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Punc}, \mathsf{PEnc})$ *be puncturable SKFE whose message space and tag space are* $\mathcal{M}$ *and* $\mathcal{T}$*, respectively. We define indistinguishability of functionality game between a challenger and an adversary* $\mathcal{A}$ *as follows.*

1. *The challenger generates a master secret key* $\mathsf{MSK} \leftarrow \mathsf{Setup}(1^\lambda)$ *and chooses a challenge bit* $b \xleftarrow{r} \{0,1\}$. *The challenger sends security parameter* 1^λ *to* $\mathcal{A}$.

2. $\mathcal{A}$ *sends* $m \in \mathcal{M}$ *and* $(\mathsf{tag}, \mathsf{tag}') \in \mathcal{T} \times \mathcal{T}$ *such that* $\mathsf{tag} \neq \mathsf{tag}'$ *to the challenger.*

3. *The challenger first computes* $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{Punc}(\mathsf{MSK}, \mathsf{tag})$. *Then, the challenger computes* $\mathsf{CT} \leftarrow \mathsf{Enc}(\mathsf{MSK}, \mathsf{tag}', m)$ *if* $b = 0$, *and* $\mathsf{CT} \leftarrow \mathsf{PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{tag}', m)$ *otherwise.*

   Then, the challenger returns $(\mathsf{MSK}, \mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{CT})$ *to* $\mathcal{A}$.

4. $\mathcal{A}$ *outputs* $b' \in \{0,1\}$.

In this game, we define the advantage of the adversary $\mathcal{A}$ *as*

$$\mathsf{Adv}^{\mathsf{if}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda) = 2|\Pr[b = b'] - \frac{1}{2}| = |\Pr[b' = 1|b = 0] - \Pr[b' = 1|b = 1]| \ .$$

We say that pSKFE *satisfies indistinguishability of functionality under puncturing if for any PPT* $\mathcal{A}$, *we have* $\mathsf{Adv}^{\mathsf{if}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda) = \mathsf{negl}(\lambda)$.

We further say that pSKFE *satisfies* ϵ*-indistinguishability of functionality under puncturing, for some concrete negligible function* $\epsilon(\cdot)$, *if for any PPT* $\mathcal{A}$ *the above advantage* $\mathsf{Adv}^{\mathsf{if}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda)$ *is smaller than* $\epsilon(\lambda)^{\Omega(1)}$.

Definition 4.4 (Secure puncturable SKFE with indistinguishability of functionality). *Let* pSKFE *be puncturable SKFE. For some concrete negligible functions* $\epsilon_1(\cdot)$ *and* $\epsilon_2(\cdot)$, *if* pSKFE *satisfies* ϵ_1*-semantic security at punctured tag and* ϵ_2*-indistinguishability of functionality under puncturing, then we say that* pSKFE *is* (ϵ_1, ϵ_2)*-secure puncturable SKFE with indistinguishability of functionality.*

Efficiency. We introduce the notion of succinctness for puncturable SKFE.

Definition 4.5 (Succinctness). *Let* λ *be a security parameter,* n *the input length of functions in* $\mathcal{F}$, *and* s *the maximum size of circuits contained in* $\mathcal{F}$.

Weakly succinct: *A puncturable SKFE scheme is said to be weakly succinct if the size of both the encryption circuit and punctured encryption circuit are bounded by* $s^\gamma \cdot \mathrm{poly}(\lambda, n)$, *where* $\gamma < 1$ *is a fixed constant and* poly *is a fixed polynomial. We call* γ *the compression factor.*

Collusion-succinct: *We say that a puncturable SKFE scheme is said to be collusion succinct if the size of both the encryption circuit and punctured encryption circuit are bounded by* $q^\gamma \cdot \mathrm{poly}(n, \lambda, s)$, *where* q *is the upper bound of issuable functional decryption keys,* $\gamma < 1$ *is a fixed constant, and* poly *is a fixed polynomial. We call* γ *the compression factor.*

4.3 Difference from Definition of Bitansky and Vaikuntanathan

There are three main differences between our definition of puncturable SKFE and that of Bitansky and Vaikuntanathan [BV15]. Two are about syntax. The other is about security.

Syntactical differences are as follows.

Tag-based encryption and decryption: In the definition of Bitansky and Vaikuntanathan, a master secret key is punctured at *two messages*. Their semantic security requires that no PPT adversary can distinguish ciphertexts of these two messages given the punctured master secret key.

We adopt the tag based syntax for the encryption and decryption algorithms while Bitansky and Vaikuntanathan do not. A tag-based definition is well-suited for our non-succinct puncturable SKFE scheme. When our non-succinct scheme encrypts a message, it generates a garbled circuit of an universal circuit into which the message is hardwired, and then masks labels of the garbled circuit by a string generated by puncturable PRF. A tag fed to the encryption algorithm is used as an input to puncturable PRF. See Section 5 and 6 for details.

In our construction of IO in Section 7, we use an input to an obfuscated circuit as a tag for ciphertexts of puncturable SKFE. Therefore, our IO construction is not significantly different from the IO construction based on puncturable SKFE by Bitansky and Vaikuntanathan from the syntactical point of view though ours is based on tag-based puncturable SKFE.

Index based key generation: We define the key generation algorithm as a stateful algorithm. In other words, for the i-th invocation, we need to feed an index i to the key generation algorithm in addition to a master secret key and a function. This is because we transform a non-succinct scheme into a weakly-succinct one *via a collusion-succinct scheme whose key generation algorithm is stateful* in Section 6.

We note that our stateful collusion-succinct scheme is just an intermediate scheme to achieve IO. We also emphasize the fact that the index-based key generation is not an issue to construct IO because our main building block is a *single-key* weakly-succinct puncturable SKFE scheme. For a single-key scheme, we do not need any state for key generation because it can issue only a single functional key.

Hereafter, we omit the index of single-key schemes in the syntax for simplicity.

Functionality under puncturing. In addition to the syntactic differences above, there is a difference about security. We defined indistinguishability of functionality under puncturing in Definition 4.3. The reason why we introduce the relaxed notion of functionality preserving property is that our weakly-succinct scheme does not satisfy functionality preserving under puncturing in Definition 4.2 but the relaxed one. Our *non-succinct* scheme satisfies functionality preserving under puncturing. One might think that a puncturable SKFE scheme that satisfies indistinguishability of functionality under puncturing is not sufficient to construct IO. This is not the case. We show that indistinguishability of functionality under puncturing suffices for constructing IO and our weakly-succinct scheme satisfies the property.

5 Construction of Single-Key Non-Succinct Puncturable SKFE

We show we can construct a single-key (non-succinct) puncturable SKFE scheme assuming only one-way functions. This construction is similar to that of a single-key non-succinct public-key functional encryption scheme proposed by Sahai and Seyalioglu [SS10]. Their construction is based on garbling scheme and public-key encryption. In our construction, we use puncturable PRF instead of public-key encryption, and, as a result, achieve the puncturable property. We recall that we can realize both garbling scheme and puncturable PRF assuming only one-way functions. We give the construction below.

Let $\mathsf{GC} = (\mathsf{Grbl}, \mathsf{Eval})$ be a garbling scheme, and $\mathsf{PPRF} = (\mathsf{F}, \mathsf{Punc}_{\mathsf{F}})$ be a puncturable PRF. Using GC and PPRF, we construct a puncturable SKFE scheme $\mathsf{OneKey} = (\mathsf{1Key.Setup}, \mathsf{1Key.KG}, \mathsf{1Key.Enc}, \mathsf{1Key.Dec}, \mathsf{1Key.Punc}, \mathsf{1Key.PEnc})$ supporting only one functional key as follows. Note that the tag space of OneKey is the same as the domain of PPRF. In addition, the index space of OneKey is $[1]$, and thus we omit the index from the description by assuming the index is always fixed to 1. Below, we assume that we can represent every function f by an n-bit string $(f[1], \cdots, f[s])$.

Construction. The scheme consists of the following algorithms.

$\mathsf{1Key.Setup}(1^\lambda)$:

- Generate $S_{j,\alpha} \xleftarrow{r} \{0,1\}^\lambda$ for every $j \in [s]$ and $\alpha \in \{0,1\}$.
- Return $\mathsf{MSK} \leftarrow \{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}$.

$\mathsf{1Key.KG}(\mathsf{MSK}, f)$:

- Parse $\{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}} \leftarrow \mathsf{MSK}$ and $(f[1], \cdots, f[s]) \leftarrow f$.
- Return $\mathsf{sk}_f \leftarrow (f, \{S_{j,f[j]}\}_{j\in[s]})$.

$\mathsf{1Key.Enc}(\mathsf{MSK}, \mathsf{tag}, m)$:

- Parse $\{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}} \leftarrow \mathsf{MSK}$.
- Compute $(\widetilde{U}, \{L_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}) \leftarrow \mathsf{Grbl}(1^\lambda, U(\cdot, m))$.
- For every $j \in [s]$ and $\alpha \in \{0,1\}$, compute $R_{j,\alpha} \leftarrow \mathsf{F}(S_{j,\alpha}, \mathsf{tag})$ and $c_{j,\alpha} \leftarrow L_{j,\alpha} \oplus R_{j,\alpha}$.
- Return $\mathsf{CT} \leftarrow (\widetilde{U}, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$.

$\mathsf{1Key.Dec}(\mathsf{sk}_f, \mathsf{tag}, \mathsf{CT})$:

- Parse $(f, \{S_j\}_{j\in[s]}) \leftarrow \mathsf{sk}_f$ and $(\widetilde{U}, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}) \leftarrow \mathsf{CT}$.
- For every $j \in [s]$, compute $R_j \leftarrow \mathsf{F}(S_j, \mathsf{tag})$ and $L_j \leftarrow c_{j,f[j]} \oplus R_j$.
- Return $y \leftarrow \mathsf{Eval}(\widetilde{U}, \{L_j\}_{j\in[s]})$.

$\mathsf{1Key.Punc}(\mathsf{MSK}, \mathsf{tag})$:

- Parse $\{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}} \leftarrow \mathsf{MSK}$.
- For every $j \in [s]$ and $\alpha \in \{0,1\}$, compute $S^*_{j,\alpha}\{\mathsf{tag}\} \leftarrow \mathsf{Punc}_{\mathsf{F}}(S_{j,\alpha}, \mathsf{tag})$.
- Return $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \{S^*_{j,\alpha}\{\mathsf{tag}\}\}_{j\in[s],\alpha\in\{0,1\}}$.

$\mathsf{1Key.PEnc}(\mathsf{MSK}^*, \mathsf{tag}', m)$

- Parse $\{S^*_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}} \leftarrow \mathsf{MSK}^*$.
- Compute $(\widetilde{U}, \{L_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}) \leftarrow \mathsf{Grbl}(1^\lambda, U(\cdot, m))$.
- For every $j \in [s]$ and $\alpha \in \{0,1\}$, compute $R_{j,\alpha} \leftarrow \mathsf{F}_{S^*_{j,\alpha}}(\mathsf{tag}')$ and $c_{j,\alpha} \leftarrow L_{j,\alpha} \oplus R_{j,\alpha}$.
- Return $\mathsf{CT} \leftarrow (\widetilde{U}, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$.

Then, we have the following theorem.

Theorem 5.1. *Let* GC *be a δ-secure garbling scheme, and* PPRF *a δ-secure puncturable PRF, where $\delta(\cdot)$ is some negligible function. Then,* OneKey *is a δ-secure single-key puncturable SKFE scheme.*

Proof of Theorem 5.1. The correctness follows from those of GC and PPRF. We first prove the functionality preserving under puncturing of OneKey. Then, we show that OneKey satisfies semantic security at punctured tag.

Functionality preserving under puncturing. We have $\mathsf{1Key.PEnc}(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{tag}', m; r) = \mathsf{1Key.Enc}(\mathsf{MSK}, \mathsf{tag}', m; r)$ for every $m \in \mathcal{M}$, $(\mathsf{tag}, \mathsf{tag}') \in \mathcal{T} \times \mathcal{T}$ such that $\mathsf{tag} \neq \mathsf{tag}'$, randomness r, $\mathsf{MSK} \leftarrow \mathsf{1Key.Setup}(1^\lambda)$, and $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{1Key.Punc}(\mathsf{MSK}, \mathsf{tag})$, since the underlying PPRF satisfies functionality preserving under puncturing property. This implies that OneKey satisfies functionality preserving under puncturing property.

Semantic security at punctured tag. Let $\mathcal{A}$ be a valid adversary that attacks the semantic security at punctured tag of OneKey. We proceed the proof via a sequence of games. Below, for every $\ell \in \{0, \cdots, 3\}$, let $\mathtt{SUC}_\ell$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game ℓ.

Game 0**:** This is the original security game regarding OneKey. Then, we have $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{pSKFE},\mathcal{A}}(\lambda) = 2|\Pr[\mathtt{SUC}_0] - \frac{1}{2}|$. The detailed description is as follows.

1. The challenger generates $S_{j,\alpha} \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ for every $j \in [s]$ and $\alpha \in \{0,1\}$, and sets $\mathsf{MSK} \leftarrow \{S_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}$. The challenger also chooses a challenge bit $b \xleftarrow{\mathsf{r}} \{0,1\}$. The challenger sends security parameter 1^λ to $\mathcal{A}$.
2. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$, $\mathsf{tag} \in \mathcal{T}$, and a function f to the challenger.
3. The challenger computes $(\widetilde{U}, \{L_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}}) \leftarrow \mathsf{Grbl}(1^\lambda, U(\cdot, m_b))$ and $R_{j,\alpha} \leftarrow \mathsf{F}(S_{j,\alpha}, \mathsf{tag})$ and $c_{j,\alpha} \leftarrow L_{j,\alpha} \oplus R_{j,\alpha}$ for every $j \in [s]$ and $\alpha \in \{0,1\}$, and sets $\mathsf{CT} \leftarrow (\widetilde{U}, \{c_{j,\alpha}\}_{j\in[s],\alpha\in\{0,1\}})$.
   Next, the challenger sets $\mathsf{sk}_f \leftarrow (f, \{S_{j,f[j]}\}_{j\in[s]})$.
   Then, the challenger computes $S^*_{j,\alpha}\{\mathsf{tag}\} \leftarrow \mathsf{Punc}_{\mathsf{F}}(S_{j,\alpha}, \mathsf{tag})$ for every $j \in [s]$ and $\alpha \in \{0,1\}$, and sets $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \{S^*_{j,\alpha}\{\mathsf{tag}\}\}_{j\in[s],\alpha\in\{0,1\}}$.
   The challenger returns $(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{CT}, \mathsf{sk}_f)$ to $\mathcal{A}$.
4. $\mathcal{A}$ outputs $b' \in \{0,1\}$.

Game 1**:** Same as Game 1 except that the challenger generates $\{R_{j,1-f[j]}\}_{j\in[n]}$ as truly random strings.

From the pseudorandomness of punctured point of PPRF, we see that $|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| \leq \delta^{\Omega(1)}$.

Game 2**:** Same as Game 2 except that for every $j \in [n]$, the challenger generates $c_{j,1-f[j]} \leftarrow R_{j,f[j]}$.

In Game 1, $c_{j,1-j[j]}$ is generated as $c_{j,1-f[j]} \leftarrow L_{j,1-f[j]} \oplus R_{j,1-f[j]}$ for every $j \in [n]$. However, in Game 1, $R_{j,1-f[j]}$ is generated as a truly random string for every $j \in [n]$, and thus the distribution of $c_[j, 1-f[j]$ is uniformly random. Therefore, In Game 1 and 2, the distribution of $c_{j,1-f[j]}$ for every $j \in [n]$ is the same and we have $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| = 0$.

Game 3**:** Same as Game 2 except that the challenger computes $(\widetilde{U}, \{L_j\}_{j\in[n]}) \leftarrow \mathsf{Sim}(1^\lambda, y)$ and $c_{j,f[j]} \leftarrow R_{j,f[j]} \oplus L_j$, where $y = f(m_0) = f(m_1)$.

In both Game 2 and 3, $\mathcal{A}$ is not given any information of labels $\{L_{j,1-f[j}\}_{j\in[n]}$. Therefore, we can use the security guarantee of GC, and obtain $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \delta^{\Omega(1)}$.

In Game 3, the choice of the challenge bit b is information theoretically hidden from the view of $\mathcal{A}$, and thus we have $|\Pr[\mathtt{SUC}_3] - \frac{1}{2}| = 0$. Then, we can estimate the advantage of $\mathcal{A}$ as

$$\begin{aligned}\frac{1}{2}\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{OneKey},\mathcal{A}}(\lambda) &= |\Pr[\mathtt{SUC}_0] - \frac{1}{2}| \\ &\leq |\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_3]| \\ &\leq \sum_{\ell=0}^{2} |\Pr[\mathtt{SUC}_\ell] - \Pr[\mathtt{SUC}_{\ell+1}]| \ . \end{aligned} \tag{1}$$

From the above argument, each term of the right side of inequality 1 is bounded by $\delta^{\Omega(1)}$. Therefore, we see that $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{OneKey},\mathcal{A}}(\lambda) \leq \delta^{\Omega(1)}$. Since the choice of $\mathcal{A}$ is arbitrary, OneKey satisfies δ-semantic security at punctured tag. □ (**Theorem 5.1**)

6 Weakly-Succinct Puncturable SKFE from Non-Succinct One

In this section, we show how to transform a single-key non-succinct puncturable SKFE scheme into a single-key weakly-succinct one using SXIO. Note that the resulting scheme satisfies only indistinguishability of functionality under puncturing property even if we start the transformation with a non-succinct scheme satisfying functionality preserving under puncturing property.

The transformation consists of 2 steps. First, we show how to construct a collusion-succinct puncturable SKFE scheme from a single-key non-succinct puncturable SKFE scheme and SXIO. Then, we give the transformation from a collusion-succinct scheme to a weakly-succinct scheme.

In fact, the intermediate collusion-succinct scheme satisfies only indistinguishability of functionality under puncturing property. This is because we adopt a construction technique similar to that proposed by Lin et al. [LPST16], and thus we use an obfuscated encryption circuit of the building block scheme by SXIO as a ciphertext of the resulting scheme. This fact is the reason the resulting weakly-succinct scheme satisfies only indistinguishability of functionality under puncturing property.

Below, we start with the first step.

6.1 From Non-Succinct to Collusion-Succinct by Using SXIO

For any q which is a fixed polynomial of λ, we show how to construct a puncturable SKFE scheme whose index space is $[q]$ based on a single-key puncturable SKFE scheme. The construction is collusion-succinct, that is, the running time of both the encryption algorithm and the punctured encryption algorithm are sub-linear in q. We show the construction below.

Let $\mathsf{OneKey} = (\mathsf{1Key.Setup}, \mathsf{1Key.KG}, \mathsf{1Key.Enc}, \mathsf{1Key.Dec}, \mathsf{1Key.Punc}, \mathsf{1Key.PEnc})$ be a puncturable SKFE scheme that we constructed in Section 5. Let $\mathsf{sxi}\mathcal{O}$ be an SXIO and $\mathsf{PPRF} = (\mathsf{F}, \mathsf{Punc}_\mathsf{F})$ a puncturable PRF. Using OneKey, $\mathsf{sxi}\mathcal{O}$, and PPRF, we construct a puncturable SKFE scheme $\mathsf{CollSuc} = (\mathsf{CS.Setup}, \mathsf{CS.KG}, \mathsf{CS.Enc}, \mathsf{CS.Dec}, \mathsf{CS.Punc}, \mathsf{CS.PEnc})$ as follows. We again note that q is a fixed polynomial of λ. The tag space of $\mathsf{CollSuc}$ is the same as that of OneKey.

Construction. The scheme consists of the following algorithms.

$\mathsf{CS.Setup}(1^\lambda)$:

- Generate $S \xleftarrow{r} \{0,1\}^\lambda$.
- Return $\mathsf{MSK} \leftarrow S$.

$\mathsf{CS.KG}(\mathsf{MSK}, f, i)$:

- Parse $S \leftarrow \mathsf{MSK}$.
- Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$ and $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
- Compute $\mathsf{1Key.sk}_f \leftarrow \mathsf{1Key.KG}(\mathsf{MSK}_i, f)$.
- Return $\mathsf{sk}_f \leftarrow (i, \mathsf{1Key.sk}_f)$.

$\mathsf{CS.Enc}(\mathsf{MSK}, \mathsf{tag}, m)$:

- Parse $S \leftarrow \mathsf{MSK}$.
- Generate $S_{\mathsf{Enc}} \xleftarrow{r} \{0,1\}^\lambda$.
- Return $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m])$. The circuit $\mathsf{E}_{\mathsf{1Key}}$ is defined in Figure 2.

$\mathsf{CS.Dec}(\mathsf{sk}_f, \mathsf{tag}, \mathsf{CT})$:

- Parse $(i, \mathsf{1Key.sk}_f) \leftarrow \mathsf{sk}_f$.
- Compute $\mathsf{CT}_i \leftarrow \mathsf{CT}(i)$.
- Return $y \leftarrow \mathsf{1Key.Dec}(\mathsf{1Key.sk}_f, \mathsf{tag}, \mathsf{CT}_i)$.

$\mathsf{CS.Punc}(\mathsf{MSK}, \mathsf{tag})$:

- Parse $S \leftarrow \mathsf{MSK}$.
- Generate $S_{\mathsf{Punc}} \xleftarrow{r} \{0,1\}^\lambda$.
- Compute $\widetilde{\mathsf{P}} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}}[S, S_{\mathsf{Punc}}, \mathsf{tag}])$. The circuit $\mathsf{P}_{\mathsf{1Key}}$ is defined in Figure 3.
- Return $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \widetilde{\mathsf{P}}$.

$\mathsf{CS.PEnc}(\mathsf{MSK}^*, \mathsf{tag}', m)$:

- Parse $\widetilde{\mathsf{P}} \leftarrow \mathsf{MSK}^*$.
- Generate $S_{\mathsf{Enc}} \xleftarrow{r} \{0,1\}^\lambda$.
- Return $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{PE}_{\mathsf{1Key}}[\widetilde{\mathsf{P}}, S_{\mathsf{Enc}}, \mathsf{tag}', m])$. The circuit $\mathsf{PE}_{\mathsf{1Key}}$ is defined in Figure 4.

Encryption circuit $\mathsf{E}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m](i)$:

Hardwired: Two PRF keys S and S_{Enc}, a tag tag, and a message m.
Input: An index $i \in [q]$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}} := \mathsf{pad}_{\mathsf{E}}(\lambda, n, s)$, which is determined in analysis.

1. Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$ and $r_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_{\mathsf{Enc}}}(i)$.
2. Compute $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
3. Return $\mathsf{CT}_i \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_i, \mathsf{tag}, m; r_{\mathsf{Enc}})$.

Figure 2: The description of $\mathsf{E}_{\mathsf{1Key}}$.

Then, we have the following theorem.

Punctured key generation circuit $\mathsf{P}_{\mathsf{1Key}}[S, S_{\mathsf{Punc}}, \mathsf{tag}](i)$:

Hardwired: Two PRF keys S and S_{Punc}, and a tag tag.
Input: An index $i \in [q]$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{P}} := \mathsf{pad}_{\mathsf{P}}(\lambda, n, s)$, which is determined in analysis.

1. Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$ and $r_{\mathsf{Punc}} \leftarrow \mathsf{F}_{S_{\mathsf{Punc}}}(i)$.
2. Compute $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
3. Return $\mathsf{MSK}^*_i\{\mathsf{tag}\} \leftarrow \mathsf{1Key.Punc}(\mathsf{MSK}_i, \mathsf{tag}; r_{\mathsf{Punc}})$.

Figure 3: The description of $\mathsf{P}_{\mathsf{1Key}}$.

Punctured encryption circuit $\mathsf{PE}_{\mathsf{1Key}}[\widetilde{\mathsf{P}}, S_{\mathsf{Enc}}, \mathsf{tag}, m](i)$:

Hardwired: A circuit $\widetilde{\mathsf{P}}$, a PRF key S_{Enc}, a tag tag, and a message m.
Input: An index $i \in [q]$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}} := \mathsf{pad}_{\mathsf{E}}(\lambda, n, s)$, which is determined in analysis.

1. Compute $\mathsf{MSK}^*_i \leftarrow \widetilde{\mathsf{P}}(i)$ and $r_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_{\mathsf{Enc}}}(i)$.
2. Return $\mathsf{CT}_i \leftarrow \mathsf{1Key.PEnc}(\mathsf{MSK}^*_i, \mathsf{tag}, m; r_{\mathsf{Enc}})$.

Figure 4: The description of $\mathsf{PE}_{\mathsf{1Key}}$.

Theorem 6.1. *Let $\delta(\cdot)$ be some negligible function. Let* OneKey *be a δ-secure single-key puncturable SKFE scheme constructed in Section 5. Let* sxi$\mathcal{O}$ *be a δ-secure γ-compressing SXIO, where $\gamma < 1$ is a constant. Let* PPRF *a δ-secure puncturable PRF. Then,* CollSuc *is a (δ, δ)-secure puncturable SKFE scheme with indistinguishability of functionality that is collusion-succinct with compression factor $\hat{\gamma}$, which is a constant smaller than* 1.

The concrete value of $\hat{\gamma}$ is determined in the efficiency analysis in the following proof of Theorem 6.1. As we will see, we can make $\hat{\gamma}$ arbitrarily small by using SXIO with appropriate compression factor.

Proof of Theorem 6.1. We first determine the size of padding parameters. Then, we analyze the efficiency. Finally, we complete the security proof.

Below, let s and n be the upper bound of size and input length of functions supported by CollSuc.

Padding Parameter. In order to complete this proof, we ensure that the encryption circuits $\mathsf{E}_{\mathsf{1Key}}$, $\mathsf{PE}_{\mathsf{1Key}}$, and $\mathsf{E}^{i^*}_{\mathsf{1Key}}$ for every $i^* \in [q]$ are indistinguishable when we obfuscate them by SXIO. Moreover, the obfuscated $\mathsf{P}_{\mathsf{1Key}}$ and $\mathsf{P}^{i^*}_{\mathsf{1Key}}$ also need to be indistinguishable. For this reason, we need appropriate size padding for these circuits. Below, we first analyze the size of padding for $\mathsf{P}_{\mathsf{1Key}}$ and $\mathsf{P}^{i^*}_{\mathsf{1Key}}$ because the description of $\mathsf{PE}_{\mathsf{1Key}}$ includes obfuscated $\mathsf{P}_{\mathsf{1Key}}$.

To guarantee the indistinguishability of $\mathsf{P}_{\mathsf{1Key}}$ and $\mathsf{P}^{i^*}_{\mathsf{1Key}}$ when we obfuscate them, we need to set

$$\mathsf{pad}_{\mathsf{P}} := \max(|\mathsf{P}_{\mathsf{1Key}}|, |\mathsf{P}^{i^*}_{\mathsf{1Key}}|) \ .$$

Both $\mathsf{P}_{\mathsf{1Key}}$ and $\mathsf{P}^{i^*}_{\mathsf{1Key}}$ includes two PRF evaluation over the domain $[q]$, and the key generation and puncturing procedure of OneKey. Since OneKey is a single-key scheme, and q is determined independently of OneKey, the

Encryption circuit $\mathsf{E}^{i^*}_{\mathsf{1Key}}[S^*, S^*_{\mathsf{Enc}}, \mathsf{tag}, m_0, m_1, \mathsf{CT}_{i^*}](i)$:

Hardwired: Two punctured PRF keys S^* and S^*_{Enc}, a tag tag, two messages m_0 and m_1, and a ciphertext CT_{i^*}.

Input: An index $i \in [q]$.

Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}} := \mathsf{pad}_{\mathsf{E}}(\lambda, n, s)$, which is determined in analysis.

1. If $i = i^*$ and $\mathsf{CT}_{i^*} \neq \bot$, return CT_{i^*}.
2. Else, compute as follows:
   - Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_{S^*}(i)$ and $r_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S^*_{\mathsf{Enc}}}(i)$.
   - Compute $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
   - If $i \leq i^*$, compute $\mathsf{CT}_i \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_i, \mathsf{tag}, m_1; r_{\mathsf{Enc}})$, and otherwise compute $\mathsf{CT}_i \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_i, \mathsf{tag}, m_0; r_{\mathsf{Enc}})$.
   - Return CT_i.

Figure 5: The description of $\mathsf{E}^{i^*}_{\mathsf{1Key}}$. The circuit is defined for every $i^* \in [q]$.

Punctured key generation circuit $\mathsf{P}^{i^*}_{\mathsf{1Key}}[S^*, S^*_{\mathsf{Punc}}, \mathsf{tag}, \mathsf{MSK}^*_{i^*}](i)$:

Hardwired: Two PRF keys S and S_{Punc}, a tag tag, and a punctured master secret key $\mathsf{MSK}^*_{i^*}$.

Input: An index $i \in [q]$.

Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{P}} := \mathsf{pad}_{\mathsf{P}}(\lambda, n, s)$, which is determined in analysis.

1. If $i = i^*$, return $\mathsf{MSK}^*_{i^*}$.
2. Else, compute as follows:
   - Compute $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_{S^*}(i)$ and $r_{\mathsf{Punc}} \leftarrow \mathsf{F}_{S^*_{\mathsf{Punc}}}(i)$.
   - Compute $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$.
   - Return $\mathsf{MSK}^*_i\{\mathsf{tag}\} \leftarrow \mathsf{1Key.Punc}(\mathsf{MSK}_i, \mathsf{tag}; r_{\mathsf{Punc}})$.

Figure 6: The description of $\mathsf{P}^{i^*}_{\mathsf{1Key}}$. The circuit is defined for every $i^* \in [q]$.

---

running time of each algorithm of OneKey is independent of q. Therefore, we have

$$\begin{aligned}\mathsf{pad}_{\mathsf{P}} &\leq \mathrm{poly}(\lambda, \log q) + \mathrm{poly}(\lambda, n, s) \\ &\leq \mathrm{poly}_{\mathsf{P}}(\lambda, n, s, \log q) \ ,\end{aligned}$$

where poly denotes an unspecified polynomial and $\mathrm{poly}_{\mathsf{P}}$ is some fixed polynomial.

Then, we move on to the analysis of the padding parameter for encryption algorithms.

We need to set $\mathsf{pad}_{\mathsf{E}}$ as

$$\mathsf{pad}_{\mathsf{E}} := \max(|\mathsf{E}_{\mathsf{1Key}}|, |\mathsf{PE}_{\mathsf{1Key}}|, |\mathsf{E}^{i^*}_{\mathsf{1Key}}|) \ .$$

$\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{E}^{i^*}_{\mathsf{1Key}}$ for every $i^* \in [q]$ consists of two PRF evaluation over the domain $[q]$, and the key generation and encryption procedure of OneKey. Therefore, we have

$$\max(|\mathsf{E}_{\mathsf{1Key}}|, |\mathsf{E}^{i^*}_{\mathsf{1Key}}|) \leq \mathrm{poly}(\lambda, n, s, \log q) \ , \tag{2}$$

where poly is an unspecified polynomial.

In addition, $\mathsf{PE}_{\mathsf{1Key}}$ includes one PRF evaluation over the domain $[q]$, the execution of $\widetilde{\mathsf{P}}$ that is obfuscated P by $\mathsf{sxi}\mathcal{O}$, and the punctured encryption procedure of OneKey. Then, since the non-trivial efficiency of $\mathsf{sxi}\mathcal{O}$, when we obfuscate a circuit C with input space $[N]$ by $\mathsf{sxi}\mathcal{O}$, we can bound the size of obfuscated C by $N^{\gamma} \cdot |C|^{c} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda)$, where $\gamma < 1$ and c are constants. Thus, we have

$$\begin{aligned} |\widetilde{\mathsf{P}}| &\leq q^{\gamma} \cdot |\mathsf{P}|^{c} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda) \\ &= q^{\gamma} \cdot |\mathrm{poly}_{\mathsf{P}}(\lambda, n, s, \log q)|^{c} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda) \\ &\leq q^{\gamma_1} \cdot \mathrm{poly}(\lambda, n, s) \ , \end{aligned}$$

where γ_1 is an arbitrary constant such that $\gamma < \gamma_1 < 1$. Hence, we obtain

$$\begin{aligned} |\mathsf{PE}_{\mathsf{1Key}}| &\leq \mathrm{poly}(\lambda, \log q) + q^{\gamma_1} \cdot \mathrm{poly}(\lambda, n, s) + \mathrm{poly}(\lambda, n, s) \\ &\leq q^{\gamma_1} \cdot \mathrm{poly}(\lambda, n, s) \ , \end{aligned} \tag{3}$$

where poly denotes an unspecified polynomial.

Therefore, from inequalities 2 and 3, we have

$$\mathsf{pad}_{\mathsf{E}} \leq q^{\gamma_1} \cdot \mathrm{poly}_{\mathsf{E}}(\lambda, n, s) \ , \tag{4}$$

where $\mathrm{poly}_{\mathsf{E}}$ is some fixed polynomial.

Efficiency. To simplify the efficiency analysis, we assume that we use two different SXIO $\mathsf{sxi}\mathcal{O}$ and $\mathsf{sxi}\mathcal{O}'$. We use $\mathsf{sxi}\mathcal{O}$ to obfuscate $\mathsf{P}_{\mathsf{1Key}}$. We use $\mathsf{sxi}\mathcal{O}'$ to obfuscate $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$.

We assume that when we obfuscate a circuit C with input space $[N]$ by $\mathsf{sxi}\mathcal{O}$ and $\mathsf{sxi}\mathcal{O}'$, we can bound the size of $\mathsf{sxi}\mathcal{O}(C)$ and $\mathsf{sxi}\mathcal{O}'(C)$ by

$$N^{\gamma} \cdot |C|^{c} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda) \ \text{ and } \ N^{\gamma'} \cdot |C|^{c'} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}'}(\lambda) \ ,$$

respectively, where γ and γ' are constants strictly smaller than 1, and c and c' are constants.

Then, from inequality 4, we can bound the running time of both CS.Enc and CS.PEnc by

$$q^{\gamma'} \cdot (\mathsf{pad}_{\mathsf{E}})^{c'} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda) \leq q^{\gamma'} \cdot (q^{\gamma_1} \cdot \mathrm{poly}_{\mathsf{E}}(\lambda, n, s))^{c'} \cdot \mathrm{poly}_{\mathsf{sxi}\mathcal{O}}(\lambda) \leq q^{\gamma' + c'\gamma_1} \cdot \mathrm{poly}(\lambda, n, s) \ ,$$

where poly denote an unspecified polynomial and γ_1 is an arbitrary constant such that $\gamma < \gamma_1 < 1$.

Therefore, if we have $\gamma' + c'\gamma_1 < 1$, we can conclude that CollSuc is collusion-succinct. From Theorem 3.15, using a collusion-resistant SKFE scheme, we can construct SXIO with arbitrary constant compression factor. Thus, we can use SXIO with compression factor smaller than $\frac{1-\gamma'}{c'}$ as $\mathsf{sxi}\mathcal{O}$, and ensure that $\hat{\gamma} := \gamma' + c'\gamma_1 < 1$ in our construction by assuming a collusion-resistant SKFE scheme. Note that γ, γ', and γ_1 are arbitrarily small constants such that $\gamma < \gamma_1$ and c' is a constant. Thus, $\hat{\gamma}$ could be an arbitrarily small constant by taking sufficiently small γ and γ'. This completes the efficiency analysis.

Indistinguishability of functionality under puncturing. A ciphertext output by the standard encryption algorithm is an obfuscated circuit of $\mathsf{E}_{\mathsf{1Key}}$. A ciphertext output by the punctured encryption algorithm is an obfuscated circuit of $\mathsf{PE}_{\mathsf{1Key}}$. Thus, if we prove that $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$ are functionally equivalent, δ-indistinguishability of functionality under puncturing of CollSuc holds due to the δ-security of $\mathsf{sxi}\mathcal{O}$.

Note that $\widetilde{\mathsf{P}}$ in $\mathsf{PE}_{\mathsf{1Key}}$ has the exactly same functionality as $\mathsf{P}_{\mathsf{1Key}}$ due to the functionality preserving property of $\mathsf{sxi}\mathcal{O}$. Thus, on input $i \in [q]$, $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$ basically compute the followings:

1. Compute $r^{i}_{\mathsf{Setup}} \leftarrow \mathsf{F}_{S}(i)$ and $r_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_{\mathsf{Enc}}}(i)$.
2. Compute $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^{\lambda}; r^{i}_{\mathsf{Setup}})$.
3. $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$ respectively computes CT_i as follows:
   - $\mathsf{E}_{\mathsf{1Key}}$ computes $\mathsf{CT}_i \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_i, \mathsf{tag}', m; r_{\mathsf{Enc}})$
   - $\mathsf{PE}_{\mathsf{1Key}}$ computes $\mathsf{CT}_i \leftarrow \mathsf{1Key.PEnc}(\mathsf{MSK}^{*}_i\{\mathsf{tag}\}, \mathsf{tag}', m; r_{\mathsf{Enc}})$ by using $\mathsf{MSK}^{*}_i\{\mathsf{tag}\} \leftarrow \mathsf{1Key.Punc}(\mathsf{MSK}_i, \mathsf{tag}; r_{\mathsf{Punc}})$ and $r_{\mathsf{Punc}} \leftarrow \mathsf{F}_{S_{\mathsf{Punc}}}(i)$.

4. Return CT_i.

Recall that OneKey satisfies functionality preserving under puncturing property defined in Definition 4.2. Thus, both $\mathsf{E}_{\mathsf{1Key}}$ and $\mathsf{PE}_{\mathsf{1Key}}$ compute the same CT_i as long as $\mathsf{tag}' \neq \mathsf{tag}$ holds and the same S_{Enc} is used in both circuits.

Thus, we can conclude that CollSuc satisfies δ-indistinguishability of functionality under puncturing by the δ-security of $\mathsf{sxi}\mathcal{O}$.

Semantic security at punctured tag Let $\mathcal{A}$ be a valid adversary that attacks the semantic security at punctured tag of CollSuc. We proof it via a sequence of games. Let $\mathtt{SUC}_j$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game j.

Game 0 This is the punctured semantic security game regarding CollSuc. Then, we have $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{CollSuc},\mathcal{A}}(\lambda) = 2|\Pr[\mathtt{SUC}_0] - \frac{1}{2}|$. The detailed description is as follows.

1. The challenger generates $S \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ and sets $\mathsf{MSK} \leftarrow S$. The challenger also chooses a challenge bit $b \xleftarrow{\mathsf{r}} \{0,1\}$. The challenger sends security parameter 1^λ to $\mathcal{A}$.
2. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$, $\mathsf{tag} \in \mathcal{T}$, and $\{f_i\}_{i\in[q]}$ to the challenger.
3. The challenger generates $S_{\mathsf{Enc}} \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ and computes $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m])$.
   Next, for every $i \in [q]$, the challenger computes as follows. The challenger computes $r^i_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i)$, $\mathsf{MSK}_i \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^i_{\mathsf{Setup}})$, and $\mathsf{1Key.sk}_{f_i} \leftarrow \mathsf{1Key.KG}(\mathsf{MSK}_i, f_i)$. Then, the challenger sets $\mathsf{sk}_{f_i} \leftarrow (i, \mathsf{1Key.sk}_{f_i})$.
   Then, the challenger generates $S_{\mathsf{Punc}} \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ and computes $\widetilde{\mathsf{P}} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}}[S, S_{\mathsf{Punc}}, \mathsf{tag}])$. Then, the challenger sets $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \widetilde{\mathsf{P}}$.
   The challenger returns $(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{CT}, \{\mathsf{sk}_{f_i}\}_{i\in[q]})$ to $\mathcal{A}$.
4. $\mathcal{A}$ outputs $b' \in \{0,1\}$.

Then, for every $i^* \in [q]$, we define the following games. We define Game $(6,0)$ as the same game as Game 0. Let $\mathtt{SUC}_{(\ell,i^*)}$ denote the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game (ℓ, i^*) for every $\ell \in \{1, \cdots, 6\}$ and $i^* \in [q]$.

Game $(1, i^*)$ Same as Game $(6, i^*-1)$ except the followings. The challenger generates $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}^{i^*}_{\mathsf{1Key}}[S^*\{i^*\}, S^*_{\mathsf{Enc}}\{i^*\}, \mathsf{tag}, m_b, m_1, \mathsf{CT}_{i^*}])$, where $\mathsf{CT}_{i^*} \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_{i^*}, \mathsf{tag}, m_b; r^{i^*}_{\mathsf{Enc}})$, $\mathsf{MSK}_{i^*} \leftarrow \mathsf{1Key.Setup}(1^\lambda; r^{i^*}_{\mathsf{Setup}})$, $r^{i^*}_{\mathsf{Setup}} \leftarrow \mathsf{F}_S(i^*)$, and $r^{i^*}_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_{\mathsf{Enc}}}(i^*)$.

The only difference between Game $(6, i^*-1)$ and $(1, i^*)$ is how CT is generated. In Game $(6, i^*-1)$, CT is generated by $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}^{i^*-1}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot])$. However, we see that $\mathsf{E}^{i^*-1}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot]$ and $\mathsf{E}^{i^*}_{\mathsf{1Key}}[S^*\{i^*\}, S^*_{\mathsf{Enc}}\{i^*\}, \mathsf{tag}, m_b, m_1, \mathsf{CT}_{i^*}]$ have exactly the same functionality. Therefore, by the indistinguishability guarantee of $\mathsf{sxi}\mathcal{O}$, we have $|\Pr[\mathtt{SUC}_{(6,i^*-1)}] - \Pr[\mathtt{SUC}_{(1,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

Game $(2, i^*)$ Same as Game $(1, i^*)$ except the followings. The challenger generates $\widetilde{\mathsf{P}} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{P}^{i^*}_{\mathsf{1Key}}[S^*\{i^*\}, S^*_{\mathsf{Punc}}\{i^*\}, \mathsf{tag}, \mathsf{MSK}^*_{i^*}\{\mathsf{tag}\}])$, where $\mathsf{MSK}^*_{i^*}\{\mathsf{tag}\} \leftarrow \mathsf{1Key.Punc}(\mathsf{MSK}_{i^*}, \mathsf{tag}; r^{i^*}_{\mathsf{Punc}})$.

Similarly to the analysis between Game $(6, i^*-1)$ and $(1, i^*)$, due to the indistinguishability guarantee of $\mathsf{sxi}\mathcal{O}$ and the fact that $\mathsf{P}_{\mathsf{1Key}}[S, S_{\mathsf{Punc}}, \mathsf{tag}]$ and $\mathsf{P}^{i^*}_{\mathsf{1Key}}[S^*\{i^*\}, S^*_{\mathsf{Punc}}\{i^*\}, \mathsf{tag}, \mathsf{MSK}^*_{i^*}\{\mathsf{tag}\}]$ have the same functionality, we have $|\Pr[\mathtt{SUC}_{(1,i^*)}] - \Pr[\mathtt{SUC}_{(2,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

Game $(3, i^*)$ Same as Game $(2, i^*)$ except that the challenger generates $r^{i^*}_{\mathsf{Setup}}$, $r^{i^*}_{\mathsf{Enc}}$, and $r^{i^*}_{\mathsf{Punc}}$ as truly random strings.

From the pseudorandomness of PPRF, it holds that $|\Pr[\mathtt{SUC}_{(2,i^*)}] - \Pr[\mathtt{SUC}_{(3,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

Game $(4, i^*)$ Same as Game $(3, i^*)$ except that the challenger generates $\mathsf{CT}_{i^*} \leftarrow \mathsf{1Key.Enc}(\mathsf{MSK}_{i^*}, \mathsf{tag}, m_1)$.

In both Game $(3, i^*)$ and $(4, i^*)$, all of MSK_{i^*}, $\mathsf{MSK}^*_{i^*}\{\mathsf{tag}\}$, and CT_{i^*} are generated under truly random strings. In addition, since $\mathcal{A}$ is a valid adversary, it holds that $f_{i^*}(m_0) = f_{i^*}(m_1)$. Therefore, from the semantic security at punctured tag of OneKey, we obtain $|\Pr[\mathtt{SUC}_{(3,i^*)}] - \Pr[\mathtt{SUC}_{(4,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

Game $(5, i^*)$ Same as Game $(4, i^*)$ except that the challenger generates $r^{i^*}_{\mathsf{Setup}}$, $r^{i^*}_{\mathsf{Enc}}$, and $r^{i^*}_{\mathsf{Punc}}$ using PPRF.

From the pseudorandomness of PPRF, it holds that $|\Pr[\mathtt{SUC}_{(4,i^*)}] - \Pr[\mathtt{SUC}_{(5,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

Game $(6, i^*)$ Same as Game $(5, i^*)$ except that the challenger generates $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}^{i^*}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot])$ and $\widetilde{\mathsf{P}} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{P}_{\mathsf{1Key}}[S, S_{\mathsf{Punc}}, \mathsf{tag}])$.

Similarly to the analysis between Game $(6, i^* - 1)$ and $(1, i^*)$, due to the indistinguishability guarantee of $\mathsf{sxi}\mathcal{O}$ and the fact that $\mathsf{E}^{i^*}_{\mathsf{1Key}}[S^*\{i^*\}, S^*_{\mathsf{Enc}}\{i^*\}, \mathsf{tag}, m_b, m_1, \mathsf{CT}_{i^*}]$ and $\mathsf{E}^{i^*}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot]$ have exactly the same functionality, we have $|\Pr[\mathtt{SUC}_{(5,i^*)}] - \Pr[\mathtt{SUC}_{(6,i^*)}]| \leq \delta^{\Omega(1)}$ for every $i^* \in [q]$.

We define one additional game.

Game 7 Same as Game $(6, q)$ except the followings. The challenger generates $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_1])$.

In Game $(6, q)$, CT is generated by $\mathsf{CT} \leftarrow \mathsf{sxi}\mathcal{O}(\mathsf{E}^q_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot])$. $\mathsf{E}^q_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot]$ always ignores m_b and outputs a ciphertext of m_1. Therefore, $\mathsf{E}^q_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_b, m_1, \bot]$ and $\mathsf{E}_{\mathsf{1Key}}[S, S_{\mathsf{Enc}}, \mathsf{tag}, m_1]$ have the same functionality. Therefore, from the indistinguishability guarantee of $\mathsf{sxi}\mathcal{O}$, we have $|\Pr[\mathtt{SUC}_{(6,q)}] - \Pr[\mathtt{SUC}_7]| \leq \delta^{\Omega(1)}$.

In Game 7, the choice of the challenge bit b is information theoretically hidden from the view of $\mathcal{A}$, and thus we have $|\Pr[\mathtt{SUC}_7] - \frac{1}{2}| = 0$. Then, we can estimate the advantage of $\mathcal{A}$ as

$$\begin{aligned}
\frac{1}{2}\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{CollSuc},\mathcal{A}}(\lambda) &= |\Pr[\mathtt{SUC}_0] - \frac{1}{2}| \\
&\leq |\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_7]| \\
&\leq |\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_{(1,1)}]| + \sum_{i^* \in [q]} \sum_{\ell=1}^{5} |\Pr[\mathtt{SUC}_{(\ell,i^*)}] - \Pr[\mathtt{SUC}_{(\ell+1,i^*)}]| \\
&\qquad + |\Pr[\mathtt{SUC}_{(6,q)}] - \Pr[\mathtt{SUC}_7]| \ . \qquad (5)
\end{aligned}$$

From the above argument, each term of the right side of inequality 5 is bounded by $\delta^{\Omega(1)}$. Therefore, we see that $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{CollSuc},\mathcal{A}}(\lambda) \leq \delta^{\Omega(1)}$. Since the choice of $\mathcal{A}$ is arbitrary, CollSuc satisfies δ-semantic security at punctured tag. □ (**Theorem 6.1**)

6.2 From Collusion-Succinct to Weakly-Succinct

In this section, we show how to construct a single-key weakly-succinct puncturable SKFE scheme from a collusion-succinct one.

This transformation is based on that proposed by Bitansky and Vaikuntanathan [BV15], and thus utilizes a decomposable randomized encoding. The difference is that we must consider puncturing and punctured encryption algorithms since we construct a puncturable SKFE scheme. In fact, we show their construction works for puncturable SKFE schemes. In addition, we consider semantic security defined in the weakly selective security manner while they considered selective security. Below, we give the construction.

We construct a single-key puncturable SKFE scheme $\mathsf{WeakSuc} = (\mathsf{WS.Setup}, \mathsf{WS.KG}, \mathsf{WS.Enc}, \mathsf{WS.Dec}, \mathsf{WS.Punc}, \mathsf{WS.PEnc})$. Let s and n be the maximum size and input length of functions supported by WeakSuc. Let RE be a c-local decomposable randomized encoding, where c is a constant. We suppose that the number of decomposed encodings of RE for a function of size s is μ. Then, μ is a polynomial bounded by $s \cdot \mathrm{poly}_{\mathsf{RE}}(\lambda, n)$, where $\mathrm{poly}_{\mathsf{RE}}(\lambda, n)$ is a fixed polynomial. We also suppose that the randomness space of RE is $\{0,1\}^\rho$, where ρ is a polynomial bounded by $s \cdot \mathrm{poly}_{\mathsf{RE}}(\lambda, n)$. Let $\mathsf{CollSuc} = (\mathsf{CS.Setup}, \mathsf{CS.KG}, \mathsf{CS.Enc}, \mathsf{CS.Dec}, \mathsf{CS.Punc}, \mathsf{CS.PEnc})$ be a puncturable SKFE scheme whose index space and tag space are $[\mu]$ and $\mathcal{T}$, respectively. Let $\mathsf{SKE} = (\mathsf{E}, \mathsf{D})$ be an SKE scheme and F a PRF. In the scheme, we use $\mathsf{F} : \{0,1\}^\lambda \times (\{0,1\}^\lambda \times [\rho]) \to \{0,1\}$. Using CollSuc, RE, SKE, and F, we construct WeakSuc as follows. The tag space of WeakSuc is $\mathcal{T}$.

$\mathsf{WS.Setup}(1^\lambda)$:

- Return $\mathsf{MSK} \leftarrow \mathsf{CS.Setup}(1^\lambda)$.

$\mathsf{WS.KG}(\mathsf{MSK}, f)$:

- Generate $K \xleftarrow{r} \{0,1\}^\lambda$ and $t \leftarrow \{0,1\}^\lambda$.
- Compute $\widehat{f} \leftarrow \mathsf{RE}(1^\lambda, f)$ and decomposed encodings $\widehat{f}_1, \cdots \widehat{f}_\mu$ together with sets of integers $(R_1, \cdots, R_\mu)$. R_i indicates which bit of a randomness $\widehat{f}_i$ depends on for every $i \in [\mu]$. Note that $R_i \subseteq [\rho]$ and $|R_i| = c$ for every $i \in [\mu]$.
- Generate $\mathsf{CT}_i^{\mathsf{ske}} \leftarrow \mathsf{E}(K, 0^{|\widehat{f}_i(\cdot,\cdot)|})$, and compute $\mathsf{sk}_{\mathsf{En}_i} \leftarrow \mathsf{CS.KG}(\mathsf{MSK}, \mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}_i^{\mathsf{ske}}], i)$ for every $i \in [\mu]$. $\mathsf{En}_{\mathsf{dre}}$ defined in Figure 7.
- Return $\mathsf{sk}_f \leftarrow (\mathsf{sk}_{\mathsf{En}_1}, \cdots, \mathsf{sk}_{\mathsf{En}_\mu})$.

$\mathsf{WS.Enc}(\mathsf{MSK}, \mathsf{tag}, m)$:

- Generate $S_{\mathsf{encd}} \leftarrow \{0,1\}^\lambda$.
- Return $\mathsf{CT} \leftarrow \mathsf{CS.Enc}(\mathsf{MSK}, \mathsf{tag}, (m, S_{\mathsf{encd}}, \bot))$.

$\mathsf{WS.Dec}(\mathsf{sk}_f, \mathsf{tag}, \mathsf{CT})$:

- Parse $(\mathsf{sk}_{\mathsf{En}_1}, \cdots, \mathsf{sk}_{\mathsf{En}_\mu}) \leftarrow \mathsf{sk}_f$.
- For every $i \in [\mu]$, compute $e_i \leftarrow \mathsf{CS.Dec}(\mathsf{sk}_{\mathsf{En}_i}, \mathsf{tag}, \mathsf{CT})$.
- Decode y from $(e_1, \cdots, e_\mu)$.
- Return y.

$\mathsf{WS.Punc}(\mathsf{MSK}, \mathsf{tag})$:

- Return $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{CS.Punc}(\mathsf{MSK}, \mathsf{tag})$.

$\mathsf{WS.PEnc}(\mathsf{MSK}^*, \mathsf{tag}', m)$:

- Generate $S_{\mathsf{encd}} \leftarrow \{0,1\}^\lambda$.
- Return $\mathsf{CT} \leftarrow \mathsf{CS.PEnc}(\mathsf{MSK}^*, \mathsf{tag}', (m, S_{\mathsf{encd}}, \bot))$.

Decomposable Randomized Encoding Circuit $\mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}_i^{\mathsf{ske}}](m, S_{\mathsf{encd}}, K)$

Hardwired: A randomized encoding $\widehat{f}_i$, a set R_i, a string t, and a ciphertext $\mathsf{CT}_i^{\mathsf{ske}}$.
Input: A message m, a PRF key S_{encd}, and an SKE secret key K.

1. If $m = \bot$, return $e_i \leftarrow \mathsf{D}(K, \mathsf{CT}_i^{\mathsf{ske}})$.
2. Else, compute as follows:
   - For $j \in R_i$, compute $r_j \leftarrow \mathsf{PRF}(S_{\mathsf{encd}}, t\|j)$, set $r_{R_i} \leftarrow \{r_j\}_{j \in R_i}$.
   - Return $e_i \leftarrow \widehat{f}_i(m; r_{R_i})$.

Figure 7: The description of $\mathsf{En}_{\mathsf{dre}}$.

Theorem 6.2. *Let $\delta(\cdot)$ be a negligible function. Let* CollSuc *be a (δ, δ)-secure puncturable SKFE scheme with indistinguishability of functionality that can issue μ functional keys and is collusion-succinct with compression factor γ, where $\gamma < 1$ is a constant. Let* RE, SKE, *and* F *be δ-secure decomposable RE, SKE scheme, and PRF, respectively. Then,* WeakSuc *be a (δ, δ)-secure single-key puncturable SKFE scheme with indistinguishability of functionality that is weakly-succinct with compression factor γ', where γ' is a constant such that $\gamma < \gamma' < 1$.*

Proof of Theorem 6.2. We start with analyzing the weak succinctness of WeakSuc, and then move on to the security proof.

Succinctness. Let $\mathsf{En}^i_{\mathsf{dre}}$ denote the circuit $\mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}^{\mathsf{ske}}_i]$. In order to issue one functional key, the construction needs to issue $1 \cdot \mu \leq s \cdot \mathrm{poly}_{\mathsf{RE}}(\lambda, n)$ keys of $\mathsf{CollSuc}$ since we consider functions of size s and n-bit input. Thus, we choose μ as the number of issuable keys of $\mathsf{CollSuc}$. The size of $\mathsf{En}^i_{\mathsf{dre}}$ is bounded by $\mathrm{poly}_{\mathsf{En}}(\lambda, n, \log s)$ since the size of $\widehat{f}_i, R_i$, and $\mathsf{CT}^{\mathsf{ske}}_i$ are independent of s from the decomposability of RE, t is a λ-bit string, and the running time of the PRF evaluation in $\mathsf{En}^i_{\mathsf{dre}}$ is logarithmic in s, where $\mathrm{poly}_{\mathsf{En}}$ is a polynomial. Since $\mathsf{CollSuc}$ is collusion-succinct, the encryption time of $\mathsf{WeakSuc}$ is bounded by

$$\mu^\gamma \cdot \mathrm{poly}(\lambda, n, |\mathsf{En}^i_{\mathsf{dre}}|) \leq (s \cdot \mathrm{poly}_{\mathsf{RE}}(\lambda, n))^\gamma \cdot \mathrm{poly}(\lambda, n, \mathrm{poly}_{\mathsf{En}}(\lambda, n, \log s)) \leq s^{\gamma'} \cdot \mathrm{poly}(\lambda, n) \ ,$$

where γ and γ' are constants such that $0 < \gamma < \gamma' < 1$, and poly denotes an unspecified polynomial. This implies that $\mathsf{WeakSuc}$ is weakly-succinct.

Indistinguishability of functionality under puncturing. $\mathsf{WS.Enc}$ and $\mathsf{WS.PEnc}$ just outputs a ciphertext output by $\mathsf{CS.Enc}$ and $\mathsf{CS.PEnc}$, respectively. Therefore, we can see that if $\mathsf{CollSuc}$ satisfies δ-indistinguishability of functionality under puncturing, then so does $\mathsf{WeakSuc}$.

Semantic security at punctured tag. Let $\mathcal{A}$ be an adversary that attacks the semantic security at punctured tag of $\mathsf{WeakSuc}$. We prove it via sequence of games. Below, for every $\ell \in \{0, \cdots, 4\}$, let $\mathtt{SUC}_\ell$ be the event that $\mathcal{A}$ succeeds in guessing the challenge bit b in Game ℓ.

Game 0**:** This is the punctured semantic security game regarding $\mathsf{WeakSuc}$. Then, we have $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{WeakSuc},\mathcal{A}}(\lambda) = 2|\Pr[\mathtt{SUC}_0] - \frac{1}{2}|$. The detailed description is as follows.

1. The challenger generate $S_{\mathsf{encd}} \leftarrow \{0,1\}^\lambda$ and computes $\mathsf{CT} \leftarrow \mathsf{CS.Enc}(\mathsf{MSK}, \mathsf{tag}, (m_b, S_{\mathsf{encd}}, \bot))$. The challenger sends security parameter 1^λ to $\mathcal{A}$.
2. $\mathcal{A}$ sends $(m_0, m_1) \in \mathcal{M} \times \mathcal{M}$, $\mathsf{tag} \in \mathcal{T}$, and a function f to the challenger.
3. The challenger generates $\mathsf{MSK} \leftarrow \mathsf{CS.Setup}(1^\lambda)$. The challenger also chooses a challenge bit $b \xleftarrow{\mathsf{r}} \{0,1\}$.
   Next, the challenger generates $K \xleftarrow{\mathsf{r}} \{0,1\}^\lambda$ and $t \leftarrow \{0,1\}^\lambda$, and computes $\widehat{f} \leftarrow \mathsf{RE}(1^\lambda, f)$ and decomposed encodings $\widehat{f}_1 \cdots \widehat{f}_\mu$ together with sets $(R_1, \cdots, R_\mu)$. Then, the challenger generates $\mathsf{CT}^{\mathsf{ske}}_i \leftarrow \mathsf{E}(K, 0^{|\widehat{f}_i(\cdot,\cdot)|})$, and computes $\mathsf{sk}_{\mathsf{En}_i} \leftarrow \mathsf{CS.KG}(\mathsf{MSK}, \mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}^{\mathsf{ske}}_i], i)$ for every $i \in [\mu]$. Moreover, the challenger sets $\mathsf{sk}_f \leftarrow (\mathsf{sk}_{\mathsf{En}_1}, \cdots, \mathsf{sk}_{\mathsf{En}_\mu})$.
   Then, the challenger computes $\mathsf{MSK}^*\{\mathsf{tag}\} \leftarrow \mathsf{CS.Punc}(\mathsf{MSK}, \mathsf{tag})$.
   The challenger returns $(\mathsf{MSK}^*\{\mathsf{tag}\}, \mathsf{CT}, \mathsf{sk}_f)$ to $\mathcal{A}$.
4. $\mathcal{A}$ outputs $b' \in \{0,1\}$.

Game 1 Same as Game 0 except that the challenger generates $\mathsf{CT}^{\mathsf{ske}}_i \leftarrow \mathsf{E}(K, e_i)$ for every $i \in [\mu]$, where $e_i \leftarrow \widehat{f}_i(m_b; r_{R_i})$.

In Game 0 and 1, $\mathcal{A}$ is not given any information of secret key K of SKE. Therefore, from the security guarantee of SKE, we have $|\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_1]| \leq \delta^{\Omega(1)}$.

Game 2 Same as Game 1 except that the challenger generates $\mathsf{CT} \leftarrow \mathsf{CS.Enc}(\mathsf{MSK}, \mathsf{tag}, (\bot, \bot, K))$.

We can see that for every $i \in [\mu]$, we have

$$\mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}^{\mathsf{ske}}_i](m_b, S_{\mathsf{encd}}, \bot) = \widehat{f}_i(m_b; r_{R_i}) = \mathsf{En}_{\mathsf{dre}}[\widehat{f}_i, R_i, t, \mathsf{CT}^{\mathsf{ske}}_i](\bot, \bot, K).$$

Therefore, from the semantic security at punctured tag of $\mathsf{CollSuc}$, it holds that $|\Pr[\mathtt{SUC}_1] - \Pr[\mathtt{SUC}_2]| \leq \delta^{\Omega(1)}$.

Game 3 Same as Game 2 except that the challenger generates r_j as a truly random string for every $j \in [\rho]$.

From the pseudorandomness of F, we have $|\Pr[\mathtt{SUC}_2] - \Pr[\mathtt{SUC}_3]| \leq \delta^{\Omega(1)}$.

Game 4 Same as Game 3 except that the challenger generates $\{e_i\}_{i \in [\mu]} \leftarrow \mathsf{Sim}(1^\lambda, s, y)$, where Sim is a simulator for RE and $y = f(m_0) = f(m_1)$.

In Game 3 and 4, for every $i \in [\mu]$, e_i hardwired into $\mathsf{En}_{\mathsf{dre}}$ after encrypted is generated with a truly random string. Therefore, from the security guarantee of RE, we have $|\Pr[\mathtt{SUC}_3] - \Pr[\mathtt{SUC}_4]| \leq \delta^{\Omega(1)}$.

In Game 4, the choice of the challenge bit b is information theoretically hidden from the view of $\mathcal{A}$, and thus we have $|\Pr[\mathtt{SUC}_4] - \frac{1}{2}| = 0$. Then, we can estimate the advantage of $\mathcal{A}$ as

$$\begin{aligned}\frac{1}{2}\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{WeakSuc},\mathcal{A}}(\lambda) &= |\Pr[\mathtt{SUC}_0] - \frac{1}{2}| \\ &\leq |\Pr[\mathtt{SUC}_0] - \Pr[\mathtt{SUC}_4]| \\ &\leq \sum_{\ell=0}^{3} |\Pr[\mathtt{SUC}_\ell] - \Pr[\mathtt{SUC}_{\ell+1}]| \ . \end{aligned} \tag{6}$$

From the above argument, each term of the right side of inequality 6 is bounded by $\delta^{\Omega(1)}$. Therefore, we see that $\mathsf{Adv}^{\mathsf{ss}}_{\mathsf{WeakSuc},\mathcal{A}}(\lambda) \leq \delta^{\Omega(1)}$. Since the choice of $\mathcal{A}$ is arbitrary, WeakSuc satisfies δ-semantic security at punctured tag. □ (**Theorem 6.2**)

7 Indistinguishability Obfuscation from Puncturable SKFE

In this section, we show how to construct IO from puncturable SKFE satisfying only indistinguishability of functionality under puncturing. Formally, we prove the following theorem.

Theorem 7.1. *Let $\delta(\lambda) = 2^{-\lambda^\epsilon}$, where $\epsilon < 1$ is a constant. Assuming there exists (δ, δ)-secure single-key weakly-succinct puncturable SKFE with indistinguishability of functionality for all circuits. Then, there exists secure IO for all circuits.*

In addition, by combining Theorems 3.15, 5.1, 6.1, and 6.2, we also obtain the following theorem.

Theorem 7.2. *Assuming there exists δ-secure collusion-resistant SKFE for all circuits, where $\delta(\cdot)$ is a negligible function. Then, there exists (δ, δ)-secure single-key weakly-succinct puncturable SKFE with indistinguishability of functionality for all circuits.*

In order to obtain Theorem 7.2, we also use δ-secure PRF, puncturable PRF, plain SKE, garbling scheme, and decomposable randomized encoding as building blocks. From Theorems 3.2, 3.4, 3.6, 3.8, and 3.10, all of these primitives are implied by δ-secure one-way functions thus implied by δ-secure collusion-resistant SKFE for all circuits.

By combining Theorems 7.1 and 7.2, we obtain the following main theorem.

Theorem 7.3. *Let $\delta(\lambda) = 2^{-\lambda^\epsilon}$, where $\epsilon < 1$ is a constant. Assuming there exists δ-secure collusion-resistant SKFE for all circuits. Then, there exists secure IO for all circuits.*

Remark 7.4 (IO for circuits with input of poly-logarithmic length). The security loss of our construction of IO is exponential in the input length of circuits, but is independent of the size of circuits. Thus, if the input length of circuits is poly-logarithmic in the security parameter, our construction of IO incurs only quasi-polynomial security loss regardless of the size of circuits. Therefore, we can obtain IO for circuits of *polynomial size* with input of poly-logarithmic length from *quasi-polynomially secure* collusion-resistant SKFE for all circuits. This is an improvement over the IO construction by Komargodski and Segev [KS17]. They showed that IO for circuits of *sub-polynomial size* with input of poly-logarithmic length is constructed from quasi-polynomially secure collusion-resistant SKFE for all circuits.

Komargodski and Segev also showed that the combination of their IO and sub-exponentially secure one-way functions yields succinct and collusion-resistant PKFE for circuits of *sub-polynomial size* with input of poly-logarithmic length. We also observe that our IO for circuits of polynomial size with input of poly-logarithmic length leads to succinct and collusion-resistant PKFE for circuits of *polynomial size* with input of poly-logarithmic length by combining sub-exponentially secure one-way functions from the result of Komargodski and Segev.

To prove Theorem 7.1, we first give the construction of IO based on puncturable SKFE. Then, we analyze its security and efficiency.

7.1 Construction

Our construction of IO is almost the same as that of Bitansky and Vaikuntanathan [BV15]. The notable difference is that we use the relaxed variant of puncturable SKFE in Definition 4.4 instead of PKFE or their puncturable SKFE. Thus, the security analysis of our IO is different from and more complex than that of Bitansky and Vaikuntanathan.

Let $\mathsf{pSKFE} = (\mathsf{Setup}, \mathsf{KG}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Punc}, \mathsf{PEnc})$ be a single-key weakly-succinct puncturable SKFE scheme. Let $\mathsf{SKE} = (\mathsf{E}, \mathsf{D})$ be an SKE scheme and $\mathsf{PPRF} = (\mathsf{F}, \mathsf{Punc}_{\mathsf{F}})$ a puncturable PRF. Below, let $\tilde{\lambda}$ denote the security parameter given to these building block schemes. Let $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^{\epsilon}}$, where $\epsilon < 1$ is a constant. We assume that pSKFE is a (δ, δ)-secure puncturable SKFE with indistinguishability of functionality under puncturing. In addition, we assume that SKE and PPRF are δ-secure. Note that the existence of such SKE and PPRF are implied by that of pSKFE. Using pSKFE, SKE, and PPRF, we construct an indistinguishability obfuscation $i\mathcal{O}$ as follows.

Given a circuit $C : \{0,1\}^n \to \{0,1\}^m$ and a security parameter λ, the obfuscator $i\mathcal{O}$ first sets the security parameter $\tilde{\lambda}$ for building block schemes as $\tilde{\lambda} = \omega((n^2 + \log \lambda)^{1/\epsilon})$. $i\mathcal{O}$ uses pSKFE whose tag space and message space is $\{0,1\}^n$ and $\{0,1\}^n \times \{0,1\}^{\tilde{\lambda}} \times \{0,1\}$, respectively. $i\mathcal{O}$ also uses PPRF whose domain is $\{0,1\}^n$. When a shorter string than expected is used as an input to these schemes, we always consider that it is fed after padded to the appropriate length. $i\mathcal{O}$ invokes the following recursive obfuscation procedure $\mathsf{r}i\mathcal{O}(1^{\tilde{\lambda}}, n, C)$ in order to obfuscate C.

$\mathsf{r}i\mathcal{O}(1^{\tilde{\lambda}}, i, C_i)$:

- If $i = 1$, return $\widetilde{C}_i \leftarrow (C_i(0), C_i(1))$.
- Else, runs as follows:
  - Generate $K_{i,0}, K_{i,1} \xleftarrow{\mathsf{r}} \{0,1\}^{\tilde{\lambda}}$ and compute $\mathsf{CT}^{\mathsf{ske}}_{i,0} \leftarrow \mathsf{E}(K_{i,0}, C_i)$ and $\mathsf{CT}^{\mathsf{ske}}_{i,1} \leftarrow \mathsf{E}(K_{i,1}, C_i)$.
  - Generate $\mathsf{MSK}_i \leftarrow \mathsf{Setup}(1^{\tilde{\lambda}})$ and compute $\mathsf{sk}_{\mathsf{Ev}_i} \leftarrow \mathsf{KG}(\mathsf{MSK}_i, \mathsf{Ev}[\mathsf{CT}^{\mathsf{ske}}_{i,0}, \mathsf{CT}^{\mathsf{ske}}_{i,1}])$. The circuit Ev_i is defined in Figure 8.
  - Generate $S_i \xleftarrow{\mathsf{r}} \{0,1\}^{\tilde{\lambda}}$ and compute $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{r}i\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}_{i-1,0}[\mathsf{MSK}_i, K_{i,0}, S_i])$. The circuit $\mathsf{E}_{i-1,0}$ is defined in Figure 9.
  - Return $\widetilde{C}_i \leftarrow (\mathsf{sk}_{\mathsf{Ev}_i}, \widetilde{\mathsf{E}}_{i-1})$.

The corresponding recursive evaluation procedure is as follows. We can evaluate $C(\boldsymbol{x}_n)$ by invoking $\mathsf{rEval}(n, \widetilde{C}, \boldsymbol{x}_n)$, where $\widetilde{C} \leftarrow \mathsf{r}i\mathcal{O}(1^{\tilde{\lambda}}, n, C)$ and $\boldsymbol{x}_n \in \{0,1\}^n$.

$\mathsf{rEval}(i, \widetilde{C}_i, \boldsymbol{x}_i)$:

- If $i = 1$, parse $(\mathsf{CT}_{1,0}, \mathsf{CT}_{1,1}) \leftarrow \widetilde{C}_i$ and return $\mathsf{CT}_{1,\boldsymbol{x}_i}$.
- Else, runs as follows:
  - Parse $(\mathsf{sk}_{\mathsf{Ev}_i}, \widetilde{\mathsf{E}}_{i-1}) \leftarrow \widetilde{C}_i$ and $\boldsymbol{x}_{i-1} \| x_i \leftarrow \boldsymbol{x}_i$.
  - Compute $(\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1}) \leftarrow \mathsf{rEval}(i-1, \widetilde{\mathsf{E}}_{i-1}, \boldsymbol{x}_{i-1})$.
  - Return $y \leftarrow \mathsf{Dec}(\mathsf{sk}_{\mathsf{Ev}_i}, \boldsymbol{x}_{i-1}, \mathsf{CT}_{i,x_i})$.

Evaluation Circuit $\mathsf{Ev}_i[\mathsf{CT}^{\mathsf{ske}}_{i,0}, \mathsf{CT}^{\mathsf{ske}}_{i,1}](\boldsymbol{x}_i, K, \alpha)$

Hardwired: Two ciphertexts $\mathsf{CT}^{\mathsf{ske}}_{i,0}$ and $\mathsf{CT}^{\mathsf{ske}}_{i,1}$.
Input: A string $\boldsymbol{x}_i \in \{0,1\}^i$, a SKE key K, and a bit $\alpha \in \{0,1\}$.

1. Compute $C_i \leftarrow \mathsf{D}(K, \mathsf{CT}^{\mathsf{ske}}_{i,\alpha})$.
2. Return $U(C_i, \boldsymbol{x}_i)$.

Figure 8: The description of Ev_i for every $i \in \{2, \cdots, n\}$. In the description, $U(\cdot, \cdot)$ is an universal circuit.

Encryption Circuit $\mathsf{E}_{i-1,\alpha}[\mathsf{MSK}_i, K_i, S_i](\boldsymbol{x}_{i-1})$

Hardwired: A master secret key MSK_i, a SKE key K_i, and a PRF key S_i.
Input: A string $\boldsymbol{x}_{i-1} \in \{0,1\}^{i-1}$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}}^{i-1} := \mathsf{pad}_{\mathsf{E}}^{i-1}(\lambda, n, s)$, which is determined in analysis.

1. For $x_i \in \{0,1\}$, compute as follows:
   - Compute $r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i} \leftarrow \mathsf{F}_{S_i}(\boldsymbol{x}_{i-1}\|x_i)$.
   - Compute $\mathsf{CT}_{i,x_i} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_i, \alpha); r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i})$.
2. Return $(\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})$.

Figure 9: The description of $\mathsf{E}_{i-1,\alpha}$ for every $i \in \{3, \cdots, n\}$ and $\alpha \in \{0,1\}$.

Remark 7.5 (On the parameter setting of $\tilde{\lambda}$). In the construction we set the security parameter $\tilde{\lambda}$ for building blocks as $\tilde{\lambda} = \omega\left((n^2 + \log\lambda)^{1/\epsilon}\right)$. In fact, this setting is the same as that of Bitansky and Vaikuntanathan [BV15]. However, the security loss is different between this work and the work by Bitansky and Vaikuntanathan. In our construction, $2^{O(n^2)}$ security loss occurs while the construction of Bitansky and Vaikuntanathan incurs $2^{O(n^2/2)}$ loss. The difference occurs due to our additional exponential hybrids that we need to complete the security proof while the building block puncturable SKFE scheme satisfies only indistinguishability of functionality under puncturing property. For the detailed security analysis, see Section 7.2.

Note that the size of padding for the encryption circuit $\mathsf{E}_{i-1,\alpha}$ is determined in the security analysis of our indistinguishability obfuscator $i\mathcal{O}$. We need to know the size of padding in order to analyze the efficiency of $i\mathcal{O}$. Therefore, we first analyze the security of $i\mathcal{O}$ in Section 7.2. Then, we analyze the efficiency of $i\mathcal{O}$ in Section 7.3. We complete the proof of Theorem 7.1 by completing the analysis of security and efficiency.

7.2 Security Analysis

Our goal is to prove that for any PPT distinguisher $\mathcal{D}$ and circuits C_0 and C_1 of the same functionality, we have

$$\begin{aligned}&|\Pr\left[\mathcal{D}(i\mathcal{O}(1^\lambda, C_0)) = 1\right] - \Pr\left[\mathcal{D}(i\mathcal{O}(1^\lambda, C_1)) = 1\right]| \\ &= |\Pr\left[\mathcal{D}(\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, n, C_0)) = 1\right] - \Pr\left[\mathcal{D}(\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, n, C_1)) = 1\right]| = \mathsf{negl}(\lambda) \ .\end{aligned}$$

In order to prove this, for every $i \in [n]$, we define

$$\delta_i := \max_{C_{i,0}, C_{i,1}} |\Pr\left[\mathcal{D}_i(\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i, C_{i,0})) = 1\right] - \Pr\left[\mathcal{D}_i(\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i, C_{i,1})) = 1\right]| \ ,$$

where $\mathcal{D}_i$ is a PPT distinguisher and $C_{i,0}$ and $C_{i,1}$ are pair of any circuits with i-bit input that are the same functionality. Then, our goal is restated to show that $\delta_n \leq 2^{-\omega(\log\lambda)}$ holds.

Note that we have $\delta_1 = 0$. This is because circuits with 1-bit input $C_{1,0}$ and $C_{1,1}$ of the same functionality are both obfuscated to the same truth table. Our goal is to prove the following lemma.

Lemma 7.6. *Let* $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\epsilon}$. *Assuming that* SKE *and* PPRF *are* δ*-secure and* pSKFE *is a* (δ, δ)*-secure puncturable SKFE with indistinguishability of functionality. It holds that*

$$\delta_i \leq 2^{2(i-1)} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \tag{7}$$

for every $i \in \{2, \cdots, n\}$.

By this lemma, we can estimate δ_n as

$$\begin{aligned}\delta_n &\leq 2^{2(n-1)} \cdot O(\delta_{n-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \\ &\leq 2^{2(n-1)} \cdot O(\delta_{n-1}) + 2^{2(n-1)} \cdot O(2^{-\Omega(\tilde{\lambda}^\epsilon)}) \\ &\leq \cdots \leq \left(\sum_{i=1}^{n} \prod_{j=1}^{i} 2^{2(n-j)}\right) \cdot O(2^{-\Omega(\tilde{\lambda}^\epsilon)}) \leq n \cdot 2^{n^2} \cdot O(2^{-\omega(n^2+\log\lambda)}) \leq 2^{-\omega(\log\lambda)} \ .\end{aligned}$$

This inequality shows that we complete the proof of Theorem 7.3.

Therefore, if we prove that inequality 7 holds for every $i \in \{2, \cdots, n\}$, that is Lemma 7.6, we can conclude that our $i\mathcal{O}$ is a secure indistinguishability obfuscator. In the rest of this section, we prove that inequality 7 holds for every $i \in \{2, \cdots, n\}$.

Let $i \in \{2, \cdots, n\}$. Let $\mathcal{D}_i$ be any PPT distinguisher again. In addition, let $C_{i,0}$ and $C_{i,1}$ be circuits with i-bit input of the same functionality that maximize the value of δ_i. First, we consider the following sequence of hybrid experiments.

$\mathcal{H}_0$: In this experiment, $\mathcal{D}_i$ is given an obfuscation of the circuit $C_{i,0}$, that is $\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i, C_{i,0})$.

$\mathcal{H}_1$: Same as $\mathcal{H}_0$ except that $\mathsf{CT}^{\mathsf{ske}}_{i,1}$ is generated as $\mathsf{CT}^{\mathsf{ske}}_{i,1} \leftarrow \mathsf{E}(K_{i,1}, C_{i,1})$. Note that in $\mathcal{H}_0$, $\mathsf{CT}^{\mathsf{ske}}_{i,1}$ is generated as $\mathsf{CT}^{\mathsf{ske}}_{i,1} \leftarrow \mathsf{E}(K_{i,1}, C_{i,0})$.

$\mathcal{H}_2$: Same as $\mathcal{H}_1$ except that $\widetilde{\mathsf{E}}_{i-1}$ is generated as $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}_{i-1,1}[\mathsf{MSK}_i, K_{i,1}, S_i])$. Note that in $\mathcal{H}_1$, $\widetilde{\mathsf{E}}_{i-1}$ is generated as $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}_{i-1,0}[\mathsf{MSK}_i, K_{i,0}, S_i])$.

$\mathcal{H}_3$: Same as $\mathcal{H}_2$ except that $\mathsf{CT}^{\mathsf{ske}}_{i,0}$ is generated as $\mathsf{CT}^{\mathsf{ske}}_{i,0} \leftarrow \mathsf{E}(K_{i,0}, C_{i,1})$. Note that in $\mathcal{H}_2$, $\mathsf{CT}^{\mathsf{ske}}_{i,0}$ is generated as $\mathsf{CT}^{\mathsf{ske}}_{i,0} \leftarrow \mathsf{E}(K_{i,1}, C_{i,0})$.

$\mathcal{H}_4$: Same as $\mathcal{H}_3$ except that $\widetilde{\mathsf{E}}_{i-1}$ is generated as $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}_{i-1,0}[\mathsf{MSK}_i, K_{i,0}, S_i])$. Note that in this experiment, the distribution of the input to $\mathcal{D}_i$ is exactly the same as an obfuscation of the circuit $C_{i,1}$, that is $\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i, C_{i,1})$.

For an experiment $\mathcal{H}$, we let $\mathcal{D}_i(\mathcal{H})$ denote the event that $\mathcal{D}_i$ outputs 1 in $\mathcal{H}$. Then, we can estimate δ_i as

$$\delta_i \leq \sum_{\ell=0}^{3} |\Pr[\mathcal{D}_i(\mathcal{H}_\ell)] - \Pr[\mathcal{D}_i(\mathcal{H}_{\ell+1})]| \ . \tag{8}$$

In the following, by estimating each term of the right hand side of inequality 8, we prove that inequality 7 holds for every $i \in \{2, \cdots, n\}$. We give relations of hybrid experiments in Figure 10 and 14 in order to see easily the dependences of hybrid experiments.

From $\mathcal{H}_0$ to $\mathcal{H}_1$ and From $\mathcal{H}_2$ to $\mathcal{H}_3$

First, we estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_0)] - \Pr[\mathcal{D}_i(\mathcal{H}_1)]|$ and $|\Pr[\mathcal{D}_i(\mathcal{H}_2)] - \Pr[\mathcal{D}_i(\mathcal{H}_3)]|$. In fact, we can easily bound these values by the security of SKE. Formally, we have the following lemma.

Lemma 7.7. *Let SKE be δ-secure, where $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\epsilon}$. Then, $|\Pr[\mathcal{D}_i(\mathcal{H}_0)] - \Pr[\mathcal{D}_i(\mathcal{H}_1)]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$ and $|\Pr[\mathcal{D}_i(\mathcal{H}_2)] - \Pr[\mathcal{D}_i(\mathcal{H}_3)]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$.*

The proof of this lemma is straightforward and thus we omit it.

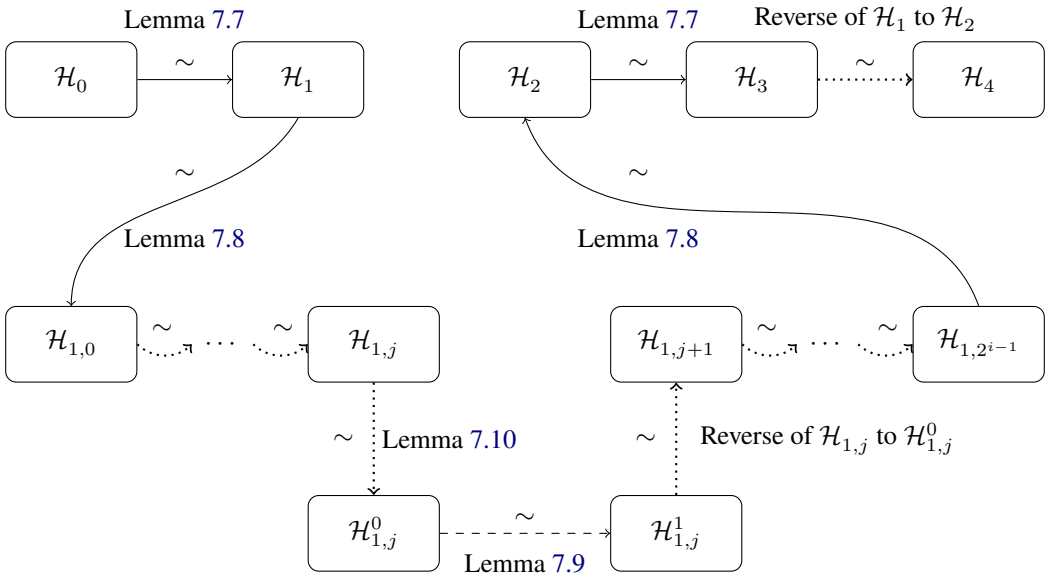

Figure 10: Relations of the hybrid experiments from $\mathcal{H}_0$ to $\mathcal{H}_4$ for the security of $ri\mathcal{O}$. Solid lines denote that the indistinguihability is proven by one step. Dashed lines denote that we use a few hybrid experiments to prove the indistinguishability. Dotted lines denote that we use many hybrid experiments to prove the indistinguishability (Figure 14 illustrates those of hybrid experiments for Lemma 7.10).

From $\mathcal{H}_1$ to $\mathcal{H}_2$ and From $\mathcal{H}_3$ to $\mathcal{H}_4$

Next, we estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]|$ and $|\Pr[\mathcal{D}_i(\mathcal{H}_3)] - \Pr[\mathcal{D}_i(\mathcal{H}_4)]|$. Since the difference between $\mathcal{H}_1$ and $\mathcal{H}_2$, and that of $\mathcal{H}_3$ and $\mathcal{H}_4$ are almost symmetric, we focus on estimating $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]|$ here. We can apply the following arguments for the estimation of $|\Pr[\mathcal{D}_i(\mathcal{H}_3)] - \Pr[\mathcal{D}_i(\mathcal{H}_4)]|$.

In order to accomplish the estimation of $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]|$, we first introduce intermediate hybrid experiments $\mathcal{H}_{1,j}$ between $\mathcal{H}_1$ and $\mathcal{H}_2$, where $j \in \{0, \cdots, 2^{i-1}\}$. In the following, let $\boldsymbol{j} \in \{0,1\}^{i-1} \cup \{1\|0^{i-1}\}$ be the binary representation of j.

$\mathcal{H}_{1,j}$: In this experiment, $\widetilde{\mathsf{E}}_{i-1}$ is computed as $\widetilde{\mathsf{E}}_{i-1} \leftarrow ri\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}^j_{i-1}[\mathsf{MSK}_i, K_{i,0}, K_{i,1}, S_i])$. The circuit E^j_{i-1} is defined in Figure 11.

Then, we have

$$|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \leq |\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,0})]| + \sum_{j=1}^{2^{i-1}-1} |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j+1})]| + |\Pr[\mathcal{D}_i(\mathcal{H}_{1,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \ .$$

If we use a PKFE scheme instead of our puncturable SKFE scheme, we can directly prove the indistinguishability between $\mathcal{H}_{1,j}$ and $\mathcal{H}_{1,j+1}$. However, to estimate each term of the right hand size of the above inequality, we need introduce the following additional hybrid experiments for every $j \in \{1, \cdots, 2^{i-1}\}$ since we use a puncturable SKFE scheme.

$\mathcal{H}^0_{1,j}$: $\widetilde{\mathsf{E}}_{i-1}$ is computed as $\widetilde{\mathsf{E}}_{i-1} \leftarrow ri\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{PE}^j_{i-1}[\mathsf{MSK}^*_i\{\boldsymbol{j}\}, K_{i,0}, K_{i,1}, S_i\{\boldsymbol{j}\|0, \boldsymbol{j}\|1\}, u_{i,0}, u_{i,1}])$ in this experiment, where $u_{i,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{j}, (\boldsymbol{j}\|b, K_{i,0}, 0); r^{\boldsymbol{j}\|b}_{\mathsf{Enc}})$ and $r^{\boldsymbol{j}\|b}_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_i}(\boldsymbol{j}\|b)$ for every $b \in \{0,1\}$. The circuit PE^j_{i-1} is defined in Figure 12. The other part of this experiment is same as $\mathcal{H}_{1,j}$.

Encryption Circuit $\mathsf{E}^{j}_{i-1}[\mathsf{MSK}_i, K_{i,0}, K_{i,1}, S_i](\boldsymbol{x}_{i-1})$

Hardwired: A master secret key MSK_i, two SKE keys $K_{i,0}$ and $K_{i,1}$, and a PRF key S_i.
Input: A string $\boldsymbol{x}_{i-1} \in \{0,1\}^{i-1}$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}}^{i-1} := \mathsf{pad}_{\mathsf{E}}^{i-1}(\lambda, n, s)$, which is determined in analysis.

1. If $\boldsymbol{x}_{i-1} < \boldsymbol{j}$, set $\alpha \leftarrow 1$, and otherwise, set $\alpha \leftarrow 0$.
2. For $x_i \in \{0,1\}$, compute as follows:
   - Compute $r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i} \leftarrow \mathsf{F}_{S_i}(\boldsymbol{x}_{i-1}\|x_i)$.
   - Compute $\mathsf{CT}_{i,x_i} \leftarrow \mathsf{Enc}(\mathsf{MSK}, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_{i,\alpha}, \alpha); r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i})$.
3. Return $(\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})$.

Figure 11: The description of E^{j}_{i-1}. The red underline is the difference from $\mathsf{E}_{i-1,\alpha}$.

$\mathcal{H}^{1}_{1,j}$: Same as $\mathcal{H}^{0}_{1,j}$ except that $u_{i,b}$ is computed by $u_{i,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}, \boldsymbol{j}, (\boldsymbol{j}\|b, K_{i,1}, 1); r_{\mathsf{Enc}}^{\boldsymbol{j}\|b})$ and $r_{\mathsf{Enc}}^{\boldsymbol{j}\|b} \leftarrow \mathsf{F}_{S_i}(\boldsymbol{j}\|b)$ for every $b \in \{0,1\}$.

Punctured Encryption Circuit $\mathsf{PE}^{j}_{i-1}[\mathsf{MSK}^{*}_i\{\boldsymbol{j}\}, K_{i,0}, K_{i,1}, S^{*}_i\{\boldsymbol{j}\|0, \boldsymbol{j}\|1|\}, u_{i,0}, u_{i,1}](\boldsymbol{x}_{i-1})$

Hardwired: A punctured master secret key $\mathsf{MSK}^{*}_i\{\boldsymbol{j}\}$, two SKE keys $K_{i,0}$ and $K_{i,1}$, a punctured PRF key $S^{*}_i\{\boldsymbol{j}\|0, \boldsymbol{j}\|1\}$, and two ciphertexts $u_{i,0}$ and $u_{i,1}$.
Input: A string $\boldsymbol{x}_{i-1} \in \{0,1\}^{i-1}$.
Padding: This circuit is padded to size $\mathsf{pad}_{\mathsf{E}}^{i-1} := \mathsf{pad}_{\mathsf{E}}^{i-1}(\lambda, n, s)$, which is determined in analysis.

1. If $\boldsymbol{x}_{i-1} = \boldsymbol{j}$, return $(u_{i,0}, u_{i,1})$.
2. Else, set $\alpha \leftarrow 1$ if $\boldsymbol{x}_{i-1} < \boldsymbol{j}$ and $\alpha \leftarrow 0$ otherwise.
3. For $x_i \in \{0,1\}$, compute as follows:
   - Compute $r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i} \leftarrow \mathsf{F}_{S^{*}_i\{\boldsymbol{j}\|0,\boldsymbol{j}\|1|\}}(\boldsymbol{x}_{i-1}\|x_i)$.
   - Compute $\mathsf{CT}_{i,x_i} \leftarrow \mathsf{PEnc}(\mathsf{MSK}^{*}_i\{\boldsymbol{j}\}, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_{i,\alpha}, \alpha); r_{\mathsf{Enc}}^{\boldsymbol{x}_{i-1}\|x_i})$.
4. Return $(\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})$.

Figure 12: The description of PE^{j}_{i-1}. Red underlines are differences from E^{j}_{i-1}.

Then, we can estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]|$ in more detail and obtain

$$\begin{aligned}
|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \le{}& |\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,0})]| \\
&+ \sum_{j=0}^{2^{i-1}-1} |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^{0}_{1,j})]| \\
&+ \sum_{j=0}^{2^{i-1}-1} |\Pr[\mathcal{D}_i(\mathcal{H}^{0}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^{1}_{1,j})]| \\
&+ \sum_{j=0}^{2^{i-1}-1} |\Pr[\mathcal{D}_i(\mathcal{H}^{1}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j+1})]| \\
&+ |\Pr[\mathcal{D}_i(\mathcal{H}_{1,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \ . \qquad (9)
\end{aligned}$$

In the rest of this section, we estimate each term of the right side of inequality 9 by Lemma 7.8, 7.9, and 7.10.

From $\mathcal{H}_1$ to $\mathcal{H}_{1,0}$ and from $\mathcal{H}_{1,2^{i-1}}$ to $\mathcal{H}_2$. We have the following lemma.

Lemma 7.8. *$|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,0})]| \leq \delta_{i-1}$ and $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \leq \delta_{i-1}$ hold.*

Proof of Lemma 7.8. We focus on proving $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,0})]| \leq \delta_{i-1}$ here.

The only difference between $\mathcal{H}_1$ and $\mathcal{H}_{1,0}$ is how $\widetilde{\mathsf{E}}_{i-1}$ is generated. In $\mathcal{H}_1$, $\widetilde{\mathsf{E}}_{i-1}$ is generated by $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}_{i-1,0}[\mathsf{MSK}_i, K_{i,0}, S_i])$. On the other hand, in $\mathcal{H}_{1,0}$, $\widetilde{\mathsf{E}}_{i-1}$ is generated by $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{E}^0_{i-1}[\mathsf{MSK}_i, K_{i,0}, K_{i,1}, S_i])$. We can see that circuits obfuscated in each experiment have the same functionality. Moreover, both circuits are padded to the same size $\mathsf{pad}^{i-1}_{\mathsf{E}}$. Therefore, we have $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,0})]| \leq \delta_{i-1}$.

By analyzing similarly, we also obtain $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| \leq \delta_{i-1}$. □ (**Lemma 7.8**)

From $\mathcal{H}^0_{1,j}$ to $\mathcal{H}^1_{1,j}$. Next, we estimate $|\Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})]|$ for every $j \in \{0, \cdots, 2^{i-1}-1\}$.

We can estimate $|\Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})]|$ by the semantic security at punctured tag of pSKFE and the security of PPRF. Formally, we have the following lemma.

Lemma 7.9. *Let $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\epsilon}$. Let pSKFE satisfy δ-semantic security at punctured tag. Let PPRF be δ-secure. Then, $|\Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$ holds for every $i \in \{0, \cdots, 2^{i-1}-1\}$.*

Proof of Lemma 7.9. In order to use the semantic security at punctured tag of pSKFE, we change the randomness $r^{\boldsymbol{j}\|b}_{\mathsf{Enc}}$ used in $u_{i,b}$ into truly random for every $b \in \{0,1\}$. Thus, we introduce the following intermediate hybrid experiments $\mathcal{H}^{0,\mathsf{rnd}}_{1,j}$ and $\mathcal{H}^{1,\mathsf{rnd}}_{1,j}$ between $\mathcal{H}^0_{1,j}$ and $\mathcal{H}^1_{1,j}$.

$\mathcal{H}^{0,\mathsf{rnd}}_{1,j}$: Same as $\mathcal{H}^0_{1,j}$ except that $r^{\boldsymbol{j}\|b}_{\mathsf{Enc}}$ is generated as a truly random string for every $b \in \{0,1\}$.

By the pseudorandomness of PPRF, we have $|\Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^{0,\mathsf{rnd}}_{1,j})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

$\mathcal{H}^{1,\mathsf{rnd}}_{1,j}$: Same as $\mathcal{H}^1_{1,j}$ except that $r^{\boldsymbol{j}\|b}_{\mathsf{Enc}}$ is generated as a truly random string for every $b \in \{0,1\}$.

Note that the only difference between $\mathcal{H}^{0,\mathsf{rnd}}_{1,j}$ and $\mathcal{H}^{1,\mathsf{rnd}}_{1,j}$ is how $u_{i,b}$ is generated for every $b \in \{0,1\}$. In $\mathcal{H}^{0,\mathsf{rnd}}_{1,j}$, $u_{i,b}$ is generated by $u_{i,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{j}, (\boldsymbol{j}\|b, K_{i,0}, 0))$. On the other hand, in $\mathcal{H}^{1,\mathsf{rnd}}_{1,j}$, $u_{i,b}$ is generated by $u_{i,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{j}, (\boldsymbol{j}\|b, K_{i,1}, 1))$. Now, in both experiments, the ciphertext $u_{i,b}$ is generated using a truly random string for every $b \in \{0,1\}$. In addition, in both experiments, $\mathsf{CT}^{\mathsf{ske}}_{i,\alpha}$ is a ciphertext of $C_{i,\alpha}$ under the SKE key $K_{i,\alpha}$ for every $\alpha \in \{0,1\}$. Hence, we have

$$\mathsf{Ev}_i[\mathsf{CT}^{\mathsf{ske}}_{i,0}, \mathsf{CT}^{\mathsf{ske}}_{i,1}](\boldsymbol{j}\|b, K_{i,0}, 0) = C_{i,0}(\boldsymbol{j}\|b) = C_{i,1}(\boldsymbol{j}\|b) = \mathsf{Ev}_i[\mathsf{CT}^{\mathsf{ske}}_{i,0}, \mathsf{CT}^{\mathsf{ske}}_{i,1}](\boldsymbol{j}\|b, K_{i,1}, 1),$$

since $C_{i,0}$ and $C_{i,1}$ are functionally equivalent.

Therefore, from the semantic security at punctured tag of pSKFE, we obtain $|\Pr[\mathcal{D}_i(\mathcal{H}^{0,\mathsf{rnd}}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^{1,\mathsf{rnd}}_{1,j})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

By the pseudorandomness of PPRF, we also have $|\Pr[\mathcal{D}_i(\mathcal{H}^{1,\mathsf{rnd}}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

From these, we see that $|\Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$. □ (**Lemma 7.9**)

From $\mathcal{H}_{1,j}$ to $\mathcal{H}^0_{1,j}$ and from $\mathcal{H}^1_{1,j}$ to $\mathcal{H}_{1,j+1}$. In the rest of this proof, we estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]|$ and $|\Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j+1})]|$ for every $j \in \{0, \cdots, 2^{i-1}-1\}$. Since the difference between $\mathcal{H}_{1,j}$ and $\mathcal{H}^0_{1,j}$, and that of $\mathcal{H}^1_{1,j}$ and $\mathcal{H}_{1,j+1}$ are almost symmetric, we focus on evaluating $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]|$ here. More precisely, we prove the following lemma.

Lemma 7.10. *Let $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\epsilon}$. Let pSKFE satisfy δ-indistinguishability of functionality under puncturing. Let PPRF be δ-secure. Then, for every $\{0, \cdots, 2^{i-1}-1\}$, we have*

$$|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| \leq 2^{i-1} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \ .$$

We can apply the following arguments for the evaluation of $|\Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j+1})]|$.

Proof of Lemma 7.10. If the underlying puncturable SKFE scheme satisfies functionality preserving under puncturing property, we can directly estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]|$ for every $j \in \{0, \cdots, 2^{i-1}-1\}$ by using the property. However, our pSKFE satisfies only indistinguishability of functionality under puncturing property. Thus, we need more hybrid experiments between $\mathcal{H}_{1,j}$ and $\mathcal{H}^0_{1,j}$ defined for every $k \in \{0, \cdots, 2^{i-1}\}$ as follows. Below, let $\boldsymbol{k} \in \{0,1\}^{i-1} \cup \{1\|0^{i-1}\}$ be the binary representation of k.

$\mathcal{H}_{1,j,k}$: $\widetilde{\mathsf{E}}_{i-1}$ is computed as $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{HE}^{j,k}_{i-1}[\mathsf{MSK}_i, \mathsf{MSK}^*_i\{\boldsymbol{j}\}, K_{i,0}, K_{i,1}, S_i, \bot, \bot])$ in this experiment. The circuit $\mathsf{HE}^{j,k}_{i-1}$ is defined in Figure 13.

Figure 14 illustrates an overview of hybrid experiments from $\mathcal{H}_{1,j-1}$ to $\mathcal{H}^0_{1,j}$.

Hybrid Encryption Circuit $\mathsf{HE}^{j,k}_{i-1}[\mathsf{MSK}_i, \mathsf{MSK}^*_i\{\boldsymbol{j}\}, K_{i,0}, K_{i,1}, S_i, v_{k,0}, v_{k,1}](\boldsymbol{x}_{i-1})$

Hardwired: A master secret key MSK_i, punctured master secret key $\mathsf{MSK}^*_i\{\boldsymbol{j}\}$, two SKE keys $K_{i,0}$ and $K_{i,1}$, PRF key S_i, and two ciphertexts $v_{k,0}$ and $v_{k,1}$.
Input: A string $\boldsymbol{x}_{i-1} \in \{0,1\}^{i-1}$.
Padding: This circuit is padded to size $\mathsf{pad}^{i-1}_{\mathsf{E}} := \mathsf{pad}^{i-1}_{\mathsf{E}}(\lambda, n, s)$, which is determined in analysis.

1. If $(v_{i,0}, v_{i,1}) \neq (\bot, \bot)$ and $\boldsymbol{x}_{i-1} = \boldsymbol{k}$, return $(v_{k,0}, v_{k,1})$.
2. Else, compute as follows:
   - If $\boldsymbol{x}_{i-1} < \boldsymbol{j}$, set $\alpha \leftarrow 1$, and otherwise, set $\alpha \leftarrow 0$.
   - For $x_i \in \{0,1\}$, compute as follows:
     - Compute $r^{\boldsymbol{x}_{i-1}\|x_i}_{\mathsf{Enc}} \leftarrow \mathsf{F}_{S_i}(\boldsymbol{x}_{i-1}\|x_i)$.
     - If $\boldsymbol{x}_{i-1} < \boldsymbol{k}$ and $\boldsymbol{x}_{i-1} \neq \boldsymbol{j}$, then compute $\mathsf{CT}_{i,x_i} \leftarrow \mathsf{PEnc}(\mathsf{MSK}^*_i\{\boldsymbol{j}\}, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_{i,\alpha}, \alpha); r^{\boldsymbol{x}_{i-1}\|x_i}_{\mathsf{Enc}})$. Otherwise, compute $\mathsf{CT}_{i,x_i} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{x}_{i-1}, (\boldsymbol{x}_{i-1}\|x_i, K_{i,\alpha}, \alpha); r^{\boldsymbol{x}_{i-1}\|x_i}_{\mathsf{Enc}})$.
   - Return $(\mathsf{CT}_{i,0}, \mathsf{CT}_{i,1})$.

Figure 13: The description of $\mathsf{HE}^{j,k}_{i-1}$. Red underlines are differences from E^j_{i-1}.

Then, we have

$$\begin{aligned}|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| \leq\ & |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,0})]| \\ &+ \sum_{k=0}^{2^{i-1}-1} |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k+1})]| \\ &+ |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| \ .\end{aligned}$$

To estimate each term of the right hand size of the above inequality, we introduce the following hybrid experiments for $k \in \{0, \cdots, 2^{i-1}-1\} \setminus \{j\}$.

$\mathcal{H}^{\mathsf{enc}}_{1,j,k}$: $\widetilde{\mathsf{E}}_{i-1}$ is computed as $\widetilde{\mathsf{E}}_{i-1} \leftarrow \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, i-1, \mathsf{HE}^{j,k}_{i-1}[\mathsf{MSK}_i, \mathsf{MSK}^*_i\{\boldsymbol{j}\}, K_{i,0}, K_{i,1}, S^*_i\{\boldsymbol{k}\|0, \boldsymbol{k}\|1\}, v_{k,0}, v_{k,1}])$ in this experiment, where $v_{k,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{k}, (\boldsymbol{k}\|b, K_{i,\alpha}, \alpha); r^{\boldsymbol{k}\|b}_{\mathsf{Enc}})$ for every $b \in \{0,1\}$.

$\mathcal{H}^{\mathsf{penc}}_{1,j,k}$: In this experiment, $v_{k,b}$ is computed by $v_{k,b} \leftarrow \mathsf{PEnc}(\mathsf{MSK}^*_i\{\boldsymbol{j}\}, \boldsymbol{k}, (\boldsymbol{k}\|b, K_{i,\alpha}, \alpha); r^{\boldsymbol{k}\|b}_{\mathsf{Enc}})$ for every $b \in \{0,1\}$.

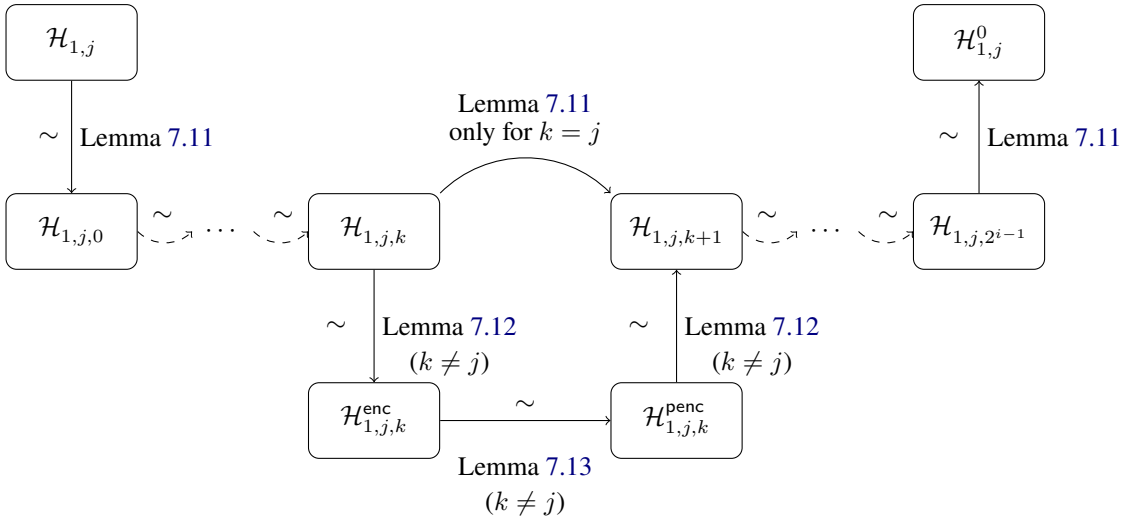

Figure 14: Relations of the hybrid experiments from $\mathcal{H}_{1,j-1}$ to $\mathcal{H}^0_{1,j}$ for Lemma 7.10. Solid lines denote that the indistinguishability is proven by one step. Dashed lines denote that we use for loop with variable k. Note that, only for $k = j$, we do not need $\mathcal{H}^{\mathsf{enc}}_{1,j,k}$ and $\mathcal{H}^{\mathsf{penc}}_{1,j,k}$ ($\mathcal{H}^{\mathsf{penc}}_{1,j,j}$ is not well-defined).

Then, we can estimate $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]|$ in more detail and obtain

$$\begin{aligned}
|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| \leq\ & |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,0})]| \\
&+ \sum_{k\in\{0,\cdots,2^{i-1}-1\}\setminus\{j\}} |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{enc}}_{1,j,k})]| \\
&+ \sum_{k\in\{0,\cdots,2^{i-1}-1\}\setminus\{j\}} |\Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{enc}}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{penc}}_{1,j,k})]| \\
&+ \sum_{k\in\{0,\cdots,2^{i-1}-1\}\setminus\{j\}} |\Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{penc}}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k+1})]| \\
&+ |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,j+1})]| \\
&+ |\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| \ . \qquad (10)
\end{aligned}$$

To bound the right hand side of inequality 10, we prove Lemma 7.11, 7.12, and 7.13.

Lemma 7.11. *It holds that*

$$\begin{aligned}
|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,0})]| &\leq \delta_{i-1} \ , \\
|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,j+1})]| &\leq \delta_{i-1} \ , \\
|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,2^{i-1}})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| &\leq \delta_{i-1} \ .
\end{aligned}$$

Lemma 7.12. *For $k \in \{0,\cdots,2^{i-1}-1\}\setminus\{j\}$, it holds that*

$$\begin{aligned}
|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{enc}}_{1,j,k})]| &\leq \delta_{i-1} \ , \\
|\Pr[\mathcal{D}_i(\mathcal{H}^{\mathsf{penc}}_{1,j,k})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k+1})]| &\leq \delta_{i-1} \ .
\end{aligned}$$

Proof of Lemma 7.11. The only difference between two experiments in each inequality is how $\widetilde{\mathsf{E}}_{i-1}$ is generated. Thus, we verify that circuits of the same functionality are obfuscated to generate $\widetilde{\mathsf{E}}_{i-1}$ in those two experiments related to each inequality.

For the first inequality, it is easy to verify that E^j_{i-1} in $\mathcal{H}_{1,j}$ is equivalent to $\mathsf{HE}^{j,0}_{i-1}$ in $\mathcal{H}_{1,j,0}$ since $\mathsf{HE}^{j,0}_{i-1}$ does not compute PEnc. Thus, the first inequality holds due to the security of $\mathsf{ri}\mathcal{O}$.

We next show the second inequality. For $\boldsymbol{x}_{i-1} \leq \boldsymbol{j}-1$ and $\boldsymbol{x}_{i-1} \geq \boldsymbol{j}+1$, the behavior of $\mathsf{HE}_{i-1}^{j,j}$ and $\mathsf{HE}_{i-1}^{j,j+1}$ are the same. On input $\boldsymbol{x}_{i-1} = \boldsymbol{j}$, $\mathsf{HE}_{i-1}^{j,k}$ always computes output ciphertexts by using Enc with MSK_i regardless of the value of k, and thus $\mathsf{HE}_{i-1}^{j,j}$ and $\mathsf{HE}_{i-1}^{j,j+1}$ behave in exactly the same way on input $\boldsymbol{x}_{i-1} = \boldsymbol{j}$. Thus, $\mathsf{HE}_{i-1}^{j,j}$ and $\mathsf{HE}_{i-1}^{j,j+1}$ are functionally equivalent and the second inequality holds due to the security of $\mathsf{ri}\mathcal{O}$.

We finally show the third inequality. On input $\boldsymbol{x}_{i-1}$, if $\boldsymbol{x}_{i-1} = \boldsymbol{j}$, PE_{i-1}^{j} in $\mathcal{H}_{1,j}^{0}$ outputs hardwired $u_{i,0}$ and $u_{i,1}$ that are ciphertexts of $\boldsymbol{j}$ generated by using Enc with MSK_i. Otherwise, it generates ciphertexts of $\boldsymbol{x}_{i-1}$ using PEnc and $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$, and outputs them. On input $\boldsymbol{x}_{i-1}$, $\mathsf{HE}_{i-1}^{j,2^{i-1}}$ in $\mathcal{H}_{1,j,2^{i-1}}^{0}$ computes output ciphertexts by using Enc with MSK_i if $\boldsymbol{x}_{i-1} = \boldsymbol{j}$, and by using PEnc with $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$ otherwise. From this fact, we see that $\mathsf{HE}_{i-1}^{j,2^{i-1}}$ in $\mathcal{H}_{1,j,2^{i-1}}$ is functionally equivalent to PE_{i-1}^{j} in $\mathcal{H}_{1,j}^{0}$. Thus, the third inequality holds due to the security of $\mathsf{ri}\mathcal{O}$. □ (**Lemma 7.11**)

Proof of Lemma 7.12. We first show the first inequality. On input $\boldsymbol{x}_{i-1} \neq \boldsymbol{k}$, $\mathsf{HE}_{i-1}^{j,k}$ in $\mathcal{H}_{1,j,k}$ runs in exactly the same way as $\mathsf{HE}_{i-1}^{j,k}$ in $\mathcal{H}_{1,j,k}^{\mathsf{enc}}$. On input $\boldsymbol{x}_{i-1} = \boldsymbol{k}$, $\mathsf{HE}_{i-1}^{j,k}$ in $\mathcal{H}_{1,j,k}$ generate ciphertexts of k by using Enc with MSK_i, and outputs them. On input $\boldsymbol{x}_{i-1} = \boldsymbol{k}$, $\mathsf{HE}_{i-1}^{j,k}$ in $\mathcal{H}_{1,j,k}^{\mathsf{enc}}$ outputs hardwired $v_{k,0}$ and $v_{k,1}$ that are ciphertexts of $\boldsymbol{k}$ generated by using Enc with MSK_i. Thus, these circuits are functionally equivalent and the inequality holds due to the security of $\mathsf{ri}\mathcal{O}$.

We next show the second inequality. In $\mathcal{H}_{1,j,k}^{\mathsf{penc}}$, $\widetilde{\mathsf{E}}_{i-1}$ is generated by obfuscating $\mathsf{HE}_{i-1}^{j,k}$ that has hardwired ciphertexts of $\boldsymbol{k}$, that is $v_{k,0}$ and $v_{k,1}$ generated by using PEnc with $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$. In $\mathcal{H}_{1,j,k+1}$, $\widetilde{\mathsf{E}}_{i-1}$ is generated by obfuscating $\mathsf{HE}_{i-1}^{j,k+1}$ that does not have hardwired ciphertexts. On input $\boldsymbol{x}_{i-1} \neq \boldsymbol{k}$, these two circuits runs in exactly the same way. On input $\boldsymbol{x}_{i-1} = \boldsymbol{k}$, $\mathsf{HE}_{i-1}^{j,k}$ in $\mathcal{H}_{1,j,k}^{\mathsf{penc}}$ outputs hardwired $v_{k,0}$ and $v_{k,1}$. On input $\boldsymbol{x}_{i-1} = \boldsymbol{k}$, $\mathsf{HE}_{i-1}^{j,k+1}$ in $\mathcal{H}_{1,j,k+1}$ generates ciphertexts of $\boldsymbol{k}$ by using PEnc with $\mathsf{MSK}_i^*\{\boldsymbol{j}\}$, and outputs them. Thus, two circuits are functionally equivalent and the second inequality holds due to the security of $\mathsf{ri}\mathcal{O}$. □ (**Lemma 7.12**)

Finally, we estimate the term $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc}})]|$ for every $k \in \{0, \cdots, 2^{i-1}-1\} \setminus \{j\}$. We can bound this term by using the indistinguishability of functionality under puncturing of pSKFE. Formally, we have the following lemma.

Lemma 7.13. *Let $\delta(\tilde{\lambda}) = 2^{-\tilde{\lambda}^\epsilon}$. Let pSKFE satisfy δ-indistinguishability of functionality under puncturing. Let PPRF be δ-secure. Then, $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc}})]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$ holds for every $j \in \{0, \cdots, 2^{i-1}-1\} \setminus \{j\}$.*

Proof of Lemma 7.13. Let k be any integer in $\{0, \cdots, 2^{i-1}-1\} \setminus \{j\}$. In order to use the indistinguishability of functionality under puncturing of pSKFE, we change the randomness $r_{\mathsf{Enc}}^{\boldsymbol{k}\|b}$ used in $v_{k,b}$ into truly random for every $b \in \{0,1\}$. Thus, we introduce the following intermediate hybrid experiments $\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}}$ and $\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}}$ between $\mathcal{H}_{1,j,k}^{\mathsf{enc}}$ and $\mathcal{H}_{1,j,k}^{\mathsf{penc}}$.

$\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}}$: Same as $\mathcal{H}_{1,j,k}^{\mathsf{enc}}$ except that $r_{\mathsf{Enc}}^{\boldsymbol{k}\|b}$ is generated as a truly random string for every $b \in \{0,1\}$.

By the pseudorandomness of PPRF, we have $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

$\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}}$: Same as $\mathcal{H}_{1,j,k}^{\mathsf{penc}}$ except that $r_{\mathsf{Enc}}^{\boldsymbol{k}\|b}$ is generated as a truly random string for every $b \in \{0,1\}$.

Note that the only difference between $\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}}$ and $\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}}$ is how $v_{k,b}$ is generated for every $b \in \{0,1\}$. In $\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}}$, $v_{k,b}$ is generated by $v_{k,b} \leftarrow \mathsf{Enc}(\mathsf{MSK}_i, \boldsymbol{k}, (\boldsymbol{k}\|b, K_{i,\alpha}, \alpha))$. On the other hand, in $\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}}$, $v_{k,b}$ is generated by $v_{k,b} \leftarrow \mathsf{PEnc}(\mathsf{MSK}_i^*\{\boldsymbol{j}\}, \boldsymbol{k}, (\boldsymbol{k}\|b, K_{i,\alpha}, \alpha))$. Now, in both experiments, the ciphertext $v_{k,b}$ is generated using a truly random string for every $b \in \{0,1\}$.

Therefore, from the indistinguishability of functionality under puncturing of pSKFE, we obtain $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc,rnd}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

By the pseudorandomness of PPRF, we alos have $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc,rnd}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc}})]| \leq 2^{-\tilde{\lambda}^\epsilon}$.

From these, we obtain $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{enc}})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j,k}^{\mathsf{penc}})]| \leq 2^{-\Omega(\tilde{\lambda}^\epsilon)}$ for every $k \in \{0, \cdots, 2^{i-1}-1\} \setminus \{j\}$.
□ (**Lemma 7.13**)

From inequality 10, and Lemma 7.11, 7.12, and 7.13, it holds that

$$\begin{aligned}|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]| &\leq 3 \cdot \delta_{i-1} + 2(2^{i-1}-1) \cdot \delta_{i-1} + (2^{i-1}-1) \cdot 2^{-\Omega(\tilde{\lambda}^\epsilon)} \\ &\leq 2^{i-1} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \ . \end{aligned} \tag{11}$$

This completes the estimation of $|\Pr[\mathcal{D}_i(\mathcal{H}_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}^0_{1,j})]|$ for every $j \in \{0, \cdots, 2^{i-1}-1\}$, that is, completes the proof of Lemma 7.10. □ (**Lemma 7.10**)

From the symmetry of the difference of $\mathcal{H}_{1,j}$ and $\mathcal{H}^0_{1,j}$, and that of $\mathcal{H}^1_{1,j}$ and $\mathcal{H}_{1,j+1}$, by analyzing similarly, we obtain

$$|\Pr[\mathcal{D}_i(\mathcal{H}^1_{1,j})] - \Pr[\mathcal{D}_i(\mathcal{H}_{1,j+1})]| \leq 2^{i-1} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \tag{12}$$

for every $j \in \{0, \cdots, 2^{i-1}-1\}$.

From inequality 11 and 12, inequality 9, and Lemma 7.8 and 7.9, we have

$$\begin{aligned}|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]| &\leq 2 \cdot \delta_{i-1} + 2 \cdot 2^{i-1} \cdot 2^{i-1} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) + 2^{i-1} \cdot 2^{-\Omega(\tilde{\lambda}^\epsilon)} \\ &\leq 2^{2(i-1)} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \ . \end{aligned} \tag{13}$$

From the symmetry of the difference of $\mathcal{H}_1$ and $\mathcal{H}_2$, and that of $\mathcal{H}_3$ and $\mathcal{H}_4$, by analyzing similarly, we also have

$$|\Pr[\mathcal{D}_i(\mathcal{H}_3)] - \Pr[\mathcal{D}_i(\mathcal{H}_4)]| \leq 2^{2(i-1)} \cdot O(\delta_{i-1} + 2^{-\Omega(\tilde{\lambda}^\epsilon)}) \ . \tag{14}$$

This completes the estimation of $|\Pr[\mathcal{D}_i(\mathcal{H}_1)] - \Pr[\mathcal{D}_i(\mathcal{H}_2)]|$ and $|\Pr[\mathcal{D}_i(\mathcal{H}_3)] - \Pr[\mathcal{D}_i(\mathcal{H}_4)]|$.

By combining inequality 13 and 14, Lemma 7.7, and inequality 8, we obtain inequality 7 in Lemma 7.6.

This completes the security analysis for $i\mathcal{O}(1^\lambda, C) = \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, n, C)$ in Section 7.1.

7.3 Efficiency Analysis

In this section, we prove that $i\mathcal{O}(1^\lambda, C) = \mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, n, C)$ runs in polynomial time. In this analysis, let s and n be the upper bound of the size and input length of circuits supported by $i\mathcal{O}$. When we invoke $\mathsf{ri}\mathcal{O}(1^{\tilde{\lambda}}, n, C)$ for some circuit C with n-bit input, $i\mathcal{O}$ generates total $n-1$ master secret keys $\mathsf{MSK}_2, \cdots, \mathsf{MSK}_n$. In other words, $i\mathcal{O}$ includes $n-1$ instances of pSKFE. If all of these $n-1$ instances runs in polynomial of $\tilde{\lambda}, n$ and s, then so does $i\mathcal{O}$. Below, we show it.

In order to accomplish the above task, we clearly distinguish each of these instances. Below, let pSKFE_i denote the instance of pSKFE with respect to MSK_i. Especially, let Enc_i denote the encryption circuit Enc corresponding to MSK_i, that is $\mathsf{Enc}(\mathsf{MSK}_i, \cdot)$. Moreover, let PEnc_i denote the punctured encryption circuit PEnc with respect to MSK^*_i, that is $\mathsf{PEnc}(\mathsf{MSK}^*_i, \cdot)$, where MSK^*_i is the punctured master secret key generated by puncturing MSK_i. Note that which tag is punctured does not affect the running time of PEnc, and thus we omit to write the tag.

We start with the estimation of the size of padding $\mathsf{pad}^{i-1}_{\mathsf{E}}$ of encryption circuits for every $i \in \{3, \cdots, n\}$. Then, using the bound of $\mathsf{pad}^{i-1}_{\mathsf{E}}$, we complete the efficiency analysis of $i\mathcal{O}$.

In the above security analysis, in addition to the encryption circuits $\mathsf{E}_{i-1,\alpha}$, we introduce encryption circuits PE^j_{i-1}, E^j_{i-1}, and $\mathsf{HE}^{j,k}_{i-1}$, where $j, k \in \{0, \cdots, 2^{i-1}\}$. We need to ensure that all of them have the same size, and thus we set

$$\mathsf{pad}^{i-1}_{\mathsf{E}} := \max(|\mathsf{E}_{i-1,\alpha}|, |\mathsf{PE}^j_{i-1}|, |\mathsf{E}^j_{i-1}|, |\mathsf{HE}^{j,k}_{i-1}|) \ .$$

All of above circuits evaluates a puncturable PRF over the domain $\{0,1\}^n$, and then computes either Enc_i or PEnc_i. Without loss of generality, we assume that Enc_i and PEnc_i have the same size for every $i \in \{2, \cdots, n\}$. This can be done by appropriate size padding. In this case, for every $i \in \{3, \cdots, n\}$, we can bound $\mathsf{pad}^{i-1}_{\mathsf{E}}$ as

$$\mathsf{pad}^{i-1}_{\mathsf{E}} \leq |\mathsf{Enc}_i| \cdot \mathrm{poly}_{\mathsf{pad}}(\tilde{\lambda}, n) \ , \tag{15}$$

where $\mathrm{poly}_{\mathsf{pad}}$ is a fixed polynomial.

We move on to the analysis of the running time of $i\mathcal{O}$. Especially, as stated earlier, we show that $|\mathsf{Enc}_i|$ is polynomial of $\tilde{\lambda}, n$ and s for every $i \in \{2, \cdots, n\}$.

First, for every $i \in \{2, \cdots, n\}$, we specify the upper bound of the size s_i of circuits that pSKFE_i has to support.

Before analysis, we introduce some notations. For every $i \in \{2, \cdots, n\}$ and circuit C_i with i-bit input, let $\mathsf{Ev}_i[C_i]$ denote the evaluation circuit Ev_i defined in Figure 8 into which C_i is hardwired after encrypted by SKE. In addition, for every $i \in \{2, \cdots, n\}$, let E_{i-1} denote the encryption circuit $\mathsf{E}_{i-1,0}[\mathsf{MSK}_i, K_i, S_i]$ defined in Figure 9.

Using this notation, we see that pSKFE_i has to support $\mathsf{Ev}_i[\mathsf{E}_i]$ for every $i \in \{2, \cdots, n-1\}$. In addition, pSKFE_n has to support $\mathsf{Ev}_n[C]$. Below, we bound the size of these circuits.

From inequality 15, for every $i \in \{3, \cdots, n\}$, we have

$$|\mathsf{E}_{i-1}| \leq |\mathsf{Enc}_i| \cdot \mathrm{poly}_{\mathsf{pad}}(\tilde{\lambda}, n) \ . \tag{16}$$

In addition, we can analyze the size of $\mathsf{Ev}_i[C_i]$ as follows. $\mathsf{Ev}_i[C_i]$ includes a decryption procedure of encrypted C_i by SKE, and evaluation of an universal circuit $U(C_i, \boldsymbol{x}_i)$, where $\boldsymbol{x}_i \in \{0,1\}^i$. Decryption procedure of SKE is done in linear time in $|C_i|$. In addition, we can perform the evaluation of an universal circuit in quasi-linear time in $|C_i|$ [Val76]. Thus, for every $i \in \{2, \cdots, n\}$, we have

$$|\mathsf{Ev}_i[C_i]| \leq |C_i| \cdot \log |C_i| \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) \leq |C_i|^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) \ , \tag{17}$$

where $\mathrm{poly}_{\mathsf{Ev}}$ is a fixed polynomial and $\gamma_1 < 1$ is an arbitrary constant. By combining inequalities 16 and 17, for every $i \in \{3, \cdots, n\}$, we obtain

$$\begin{aligned} |\mathsf{Ev}_{i-1}[\mathsf{E}_{i-1}]| &\leq |\mathsf{E}_{i-1}|^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) \\ &\leq \left(|\mathsf{Enc}_i| \cdot \mathrm{poly}_{\mathsf{pad}}(\tilde{\lambda}, n)\right)^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) \\ &\leq |\mathsf{Enc}_i|^{1+\gamma_1} \cdot \mathrm{poly}_1(\tilde{\lambda}, n) \ , \end{aligned} \tag{18}$$

where poly_1 is some fixed polynomial.

Therefore, for every $i \in \{3, \cdots, n\}$, we have

$$s_{i-1} \leq |\mathsf{Enc}_i|^{1+\gamma_1} \cdot \mathrm{poly}_1(\tilde{\lambda}, n) \ . \tag{19}$$

Moreover, from inequality 17, we have

$$s_n \leq |C|^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) = s^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n) \ . \tag{20}$$

Using the bound of s_i, we complete the efficiency analysis by estimating $|\mathsf{Enc}_i|$ for every $i \in \{2, \cdots, n\}$.

Recall that the building block pSKFE is weakly succinct, and for every $i \in \{2, \cdots, n\}$, pSKFE_i encrypts $n + \tilde{\lambda} + 1$ bit plaintexts in the construction. Therefore, for every $i \in \{2, \cdots, n\}$, we have

$$\begin{aligned} |\mathsf{Enc}_i| &\leq s_i^{\gamma} \cdot \mathrm{poly}(\tilde{\lambda}, n + \tilde{\lambda} + 1) \\ &\leq s_i^{\gamma} \cdot \mathrm{poly}_{\mathsf{E}}(\tilde{\lambda}, n) \ , \end{aligned} \tag{21}$$

where $\gamma < 1$ is some constant, poly denotes unspecified polynomials, and $\mathrm{poly}_{\mathsf{E}}$ denotes some fixed polynomial. By substituting inequality 19 into 21, for every $i \in \{3, \cdots, n\}$, we obtain

$$\begin{aligned} |\mathsf{Enc}_{i-1}| &\leq \left(|\mathsf{Enc}_i|^{1+\gamma_1} \cdot \mathrm{poly}_1(\tilde{\lambda}, n)\right)^{\gamma} \cdot \mathrm{poly}_{\mathsf{E}}(\tilde{\lambda}, n) \\ &\leq |\mathsf{Enc}_i|^{\gamma_2} \cdot \mathrm{poly}_2(\tilde{\lambda}, n) \ , \end{aligned} \tag{22}$$

where γ is a constant such that $(1+\gamma_1)\gamma < \gamma_2 < 1$ and poly_2 is some fixed polynomial. Note that since we can choose γ_1 as an arbitrary small constant and $\gamma < 1$, by setting $\gamma_1 < \frac{1-\gamma}{\gamma}$, γ_2 satisfying the above condition exists. In addition, from inequality 20, we also have

$$\begin{aligned} |\mathsf{Enc}_n| &\leq \left(s^{1+\gamma_1} \cdot \mathrm{poly}_{\mathsf{Ev}}(\tilde{\lambda}, n)\right)^{\gamma} \cdot \mathrm{poly}_{\mathsf{E}}(\tilde{\lambda}, n) \\ &\leq s^{\gamma_2} \cdot \mathrm{poly}_3(\tilde{\lambda}, n) \ , \end{aligned} \tag{23}$$

where poly_3 is some fixed polynomial.

Then, from inequalities 22 and 23, for every $i \in \{2, \cdots, n\}$, it holds that

$$\begin{aligned}
|\mathsf{Enc}_i| &\leq |\mathsf{Enc}_n|^{\gamma_2^{n-i}} \cdot \prod_{j=1}^{n-i} \mathrm{poly}_2(\tilde{\lambda}, n)^{\gamma_2^{j-1}} \\
&\leq \left(s^{\gamma_2} \cdot \mathrm{poly}_3(\tilde{\lambda}, n)\right)^{\gamma_2^{n-i}} \cdot \prod_{j=1}^{n-i} \mathrm{poly}_2(\tilde{\lambda}, n)^{\gamma_2^{j-1}} \\
&\leq s^{\gamma_2} \cdot \mathrm{poly}_3(\tilde{\lambda}, n) \cdot \mathrm{poly}_2(\tilde{\lambda}, n)^{\frac{1}{1-\gamma_2}} \quad .
\end{aligned}$$

The third inequality follows from the fact that $\gamma_2 < 1$. The above inequality means that the encryption algorithm of pSKFE_i runs in polynomial time of $\tilde{\lambda}, n$ and s for every $i \in \{2, \cdots, n\}$.

In that case, all of the algorithms of pSKFE_i runs in polynomial of $\tilde{\lambda}, n$ and s for every $i \in \{2, \cdots, n\}$. Thus, we conclude that $i\mathcal{O}$ runs in polynomial of $\tilde{\lambda}, n$ and s.

We completed the analysis of security and efficiency. Thus, we completed the proof of Theorem 7.1 and proved that we can construct IO for all circuits solely from SKFE for all circuits.

References

[AB15] Benny Applebaum and Zvika Brakerski. Obfuscating circuits via composite-order graded encoding. In Dodis and Nielsen [DN15], pages 528–556. doi:10.1007/978-3-662-46497-7_21. (Cited on page 1.)

[ADGM17] Daniel Apon, Nico Döttling, Sanjam Garg, and Pratyay Mukherjee. Cryptanalysis of indistinguishability obfuscations of circuits over GGH13. In *44rd International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland (to appear)*, 2017. (Cited on page 1.)

[AGIS14] Prabhanjan Vijendra Ananth, Divya Gupta, Yuval Ishai, and Amit Sahai. Optimizing obfuscation: Avoiding Barrington's theorem. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *ACM CCS 14*, pages 646–658. ACM Press, November 2014. (Cited on page 1.)

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Computationally private randomizing polynomials and their applications. *Computational Complexity*, 15(2):115–162, 2006. URL: http://dx.doi.org/10.1007/s00037-006-0211-8, doi:10.1007/s00037-006-0211-8. (Cited on page 13.)

[AJ15] Prabhanjan Ananth and Abhishek Jain. Indistinguishability obfuscation from compact functional encryption. In Gennaro and Robshaw [GR15], pages 308–326. doi:10.1007/978-3-662-47989-6_15. (Cited on page 1, 3.)

[AJS15] Prabhanjan Ananth, Abhishek Jain, and Amit Sahai. Indistinguishability obfuscation from functional encryption for simple functions. Cryptology ePrint Archive, Report 2015/730, 2015. http://eprint.iacr.org/2015/730. (Cited on page 1.)

[AS15] Gilad Asharov and Gil Segev. Limits on the power of indistinguishability obfuscation and functional encryption. In Guruswami [Gur15], pages 191–209. doi:10.1109/FOCS.2015.21. (Cited on page 1, 2, 3.)

[AS17] Prabhanjan Ananth and Amit Sahai. Projective arithmetic functional encryption and indistinguishability obfuscation from degree-5 multilinear maps. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, pages 152–181, 2017. (Cited on page 1.)

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2001. (Cited on page 1, 5.)

[BGI14] Elette Boyle, Shafi Goldwasser, and Ioana Ivan. Functional signatures and pseudorandom functions. In Hugo Krawczyk, editor, *PKC 2014*, volume 8383 of *LNCS*, pages 501–519. Springer, Heidelberg, March 2014. doi:10.1007/978-3-642-54631-0_29. (Cited on page 12.)

[BGK+14] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. In Nguyen and Oswald [NO14], pages 221–238. doi:10.1007/978-3-642-55220-5_13. (Cited on page 1.)

[BGL+15] Nir Bitansky, Sanjam Garg, Huijia Lin, Rafael Pass, and Sidharth Telang. Succinct randomized encodings and their applications. In Servedio and Rubinfeld [SR15], pages 439–448. (Cited on page 1.)

[BGMS15] Dan Boneh, Divya Gupta, Ilya Mironov, and Amit Sahai. Hosting services on an untrusted cloud. In Oswald and Fischlin [OF15], pages 404–436. doi:10.1007/978-3-662-46803-6_14. (Cited on page 1.)

[BHR12] Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM CCS 12*, pages 784–796. ACM Press, October 2012. (Cited on page 13.)

[BMSZ16] Saikrishna Badrinarayanan, Eric Miles, Amit Sahai, and Mark Zhandry. Post-zeroizing obfuscation: New mathematical tools, and the case of evasive circuits. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part II*, volume 9666 of *LNCS*, pages 764–791. Springer, Heidelberg, May 2016. doi:10.1007/978-3-662-49896-5_27. (Cited on page 1.)

[BNPW16] Nir Bitansky, Ryo Nishimaki, Alain Passelègue, and Daniel Wichs. From cryptomania to obfustopia through secret-key functional encryption. In Hirt and Smith [HS16], pages 391–418. (Cited on page 1, 3, 6, 7, 15.)

[BPR+08] Dan Boneh, Periklis A. Papakonstantinou, Charles Rackoff, Yevgeniy Vahlis, and Brent Waters. On the impossibility of basing identity based encryption on trapdoor permutations. In *49th FOCS*, pages 283–292. IEEE Computer Society Press, October 2008. (Cited on page 2.)

[BPW16] Nir Bitansky, Omer Paneth, and Daniel Wichs. Perfect structure on the edge of chaos - trapdoor permutations from indistinguishability obfuscation. In Eyal Kushilevitz and Tal Malkin, editors, *TCC 2016-A, Part I*, volume 9562 of *LNCS*, pages 474–502. Springer, Heidelberg, January 2016. doi:10.1007/978-3-662-49096-9_20. (Cited on page 1.)

[BR14] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 1–25. Springer, Heidelberg, February 2014. doi:10.1007/978-3-642-54242-8_1. (Cited on page 1.)

[BS15] Zvika Brakerski and Gil Segev. Function-private functional encryption in the private-key setting. In Dodis and Nielsen [DN15], pages 306–324. doi:10.1007/978-3-662-46497-7_12. (Cited on page 4.)

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In Yuval Ishai, editor, *TCC 2011*, volume 6597 of *LNCS*, pages 253–273. Springer, Heidelberg, March 2011. (Cited on page 1.)

[BV15] Nir Bitansky and Vinod Vaikuntanathan. Indistinguishability obfuscation from functional encryption. In Guruswami [Gur15], pages 171–190. doi:10.1109/FOCS.2015.20. (Cited on page 1, 2, 3, 5, 6, 8, 9, 11, 15, 17, 26, 30, 31.)

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. In Kazue Sako and Palash Sarkar, editors, *ASIACRYPT 2013, Part II*, volume 8270 of *LNCS*, pages 280–300. Springer, Heidelberg, December 2013. doi:10.1007/978-3-642-42045-0_15. (Cited on page 12.)

[CGH+15] Jean-Sébastien Coron, Craig Gentry, Shai Halevi, Tancrède Lepoint, Hemanta K. Maji, Eric Miles, Mariana Raykova, Amit Sahai, and Mehdi Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In Gennaro and Robshaw [GR15], pages 247–266. doi:10.1007/978-3-662-47989-6_12. (Cited on page 1.)

[CGH17] Yilei Chen, Craig Gentry, and Shai Halevi. Cryptanalyses of candidate branching program obfuscators. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pages 278–307, 2017. (Cited on page 1.)

[CHJV15] Ran Canetti, Justin Holmgren, Abhishek Jain, and Vinod Vaikuntanathan. Succinct garbling and indistinguishability obfuscation for RAM programs. In Servedio and Rubinfeld [SR15], pages 429–437. (Cited on page 1.)

[CHN+16] Aloni Cohen, Justin Holmgren, Ryo Nishimaki, Vinod Vaikuntanathan, and Daniel Wichs. Watermarking cryptographic capabilities. In Daniel Wichs and Yishay Mansour, editors, *48th ACM STOC*, pages 1115–1127. ACM Press, June 2016. (Cited on page 1.)

[CLLT17] Jean-Sébastien Coron, Moon Sung Lee, Tancrède Lepoint, and Mehdi Tibouchi. Zeroizing attacks on indistinguishability obfuscation over CLT13. In *Public-Key Cryptography - PKC 2017 - 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28-31, 2017, Proceedings, Part I*, pages 41–58, 2017. (Cited on page 1.)

[CLTV15] Ran Canetti, Huijia Lin, Stefano Tessaro, and Vinod Vaikuntanathan. Obfuscation of probabilistic circuits and applications. In Dodis and Nielsen [DN15], pages 468–497. doi:10.1007/978-3-662-46497-7_19. (Cited on page 4, 9.)

[DN15] Yevgeniy Dodis and Jesper Buus Nielsen, editors. *TCC 2015, Part II*, volume 9015 of *LNCS*. Springer, Heidelberg, March 2015. (Cited on page 41, 42, 43.)

[FRS16] Rex Fernando, Peter M. R. Rasmussen, and Amit Sahai. Preventing CLT attacks on obfuscation with linear overhead. *IACR Cryptology ePrint Archive*, 2016:1070, 2016. (Cited on page 1.)

[GG14] Juan A. Garay and Rosario Gennaro, editors. *CRYPTO 2014, Part I*, volume 8616 of *LNCS*. Springer, Heidelberg, August 2014. (Cited on page 45.)

[GGH+13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th FOCS*, pages 40–49. IEEE Computer Society Press, October 2013. (Cited on page 1, 3.)

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. (Cited on page 11, 12.)

[GMM+16] Sanjam Garg, Eric Miles, Pratyay Mukherjee, Amit Sahai, Akshayaram Srinivasan, and Mark Zhandry. Secure obfuscation in a weak multilinear map model. In Hirt and Smith [HS16], pages 241–268. (Cited on page 1.)

[GPSZ17] Sanjam Garg, Omkant Pandey, Akshayaram Srinivasan, and Mark Zhandry. Breaking the subexponential barrier in obfustopia. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pages 156–181, 2017. (Cited on page 2.)

[GR15] Rosario Gennaro and Matthew J. B. Robshaw, editors. *CRYPTO 2015, Part I*, volume 9215 of *LNCS*. Springer, Heidelberg, August 2015. (Cited on page 41, 43.)

[Gur15] Venkatesan Guruswami, editor. *56th FOCS*. IEEE Computer Society Press, October 2015. (Cited on page 41, 42.)

[HJK+16] Dennis Hofheinz, Tibor Jager, Dakshita Khurana, Amit Sahai, Brent Waters, and Mark Zhandry. How to generate and use universal samplers. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 715–744. Springer, Heidelberg, December 2016. doi:10.1007/978-3-662-53890-6_24. (Cited on page 1.)

[HS16] Martin Hirt and Adam D. Smith, editors. *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part II*, volume 9986 of *Lecture Notes in Computer Science*, 2016. (Cited on page 42, 43.)

[HSW14] Susan Hohenberger, Amit Sahai, and Brent Waters. Replacing a random oracle: Full domain hash from indistinguishability obfuscation. In Nguyen and Oswald [NO14], pages 201–220. doi:10.1007/978-3-642-55220-5_12. (Cited on page 1.)

[Imp95] Russell Impagliazzo. A personal view of average-case complexity. In *Proceedings of the Tenth Annual Structure in Complexity Theory Conference, Minneapolis, Minnesota, USA, June 19-22, 1995*, pages 134–147. IEEE Computer Society, 1995. URL: http://dx.doi.org/10.1109/SCT.1995.514853, doi:10.1109/SCT.1995.514853. (Cited on page 2.)

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *21st ACM STOC*, pages 44–61. ACM Press, May 1989. (Cited on page 2.)

[KLW15] Venkata Koppula, Allison Bishop Lewko, and Brent Waters. Indistinguishability obfuscation for turing machines with unbounded memory. In Servedio and Rubinfeld [SR15], pages 419–428. (Cited on page 1.)

[KMN+14] Ilan Komargodski, Tal Moran, Moni Naor, Rafael Pass, Alon Rosen, and Eylon Yogev. One-way functions and (im)perfect obfuscation. In *55th FOCS*, pages 374–383. IEEE Computer Society Press, October 2014. doi:10.1109/FOCS.2014.47. (Cited on page 2.)

[KPTZ13] Aggelos Kiayias, Stavros Papadopoulos, Nikos Triandopoulos, and Thomas Zacharias. Delegatable pseudorandom functions and applications. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *ACM CCS 13*, pages 669–684. ACM Press, November 2013. (Cited on page 12.)

[KS17] Ilan Komargodski and Gil Segev. From minicrypt to obfustopia via private-key functional encryption. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part I*, pages 122–151, 2017. (Cited on page 2, 3, 29.)

[Lin16a] Huijia Lin. Indistinguishability obfuscation from constant-degree graded encoding schemes. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 28–57. Springer, Heidelberg, May 2016. doi:10.1007/978-3-662-49890-3_2. (Cited on page 1.)

[Lin16b] Huijia Lin. Indistinguishability obfuscation from DDH on 5-linear maps and locality-5 PRGs. *IACR Cryptology ePrint Archive*, 2016:1096, 2016. (Cited on page 1.)

[LP09] Yehuda Lindell and Benny Pinkas. A proof of security of yao's protocol for two-party computation. *J. Cryptology*, 22(2):161–188, 2009. (Cited on page 13.)

[LPST16] Huijia Lin, Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation with non-trivial efficiency. In Chen-Mou Cheng, Kai-Min Chung, Giuseppe Persiano, and Bo-Yin Yang, editors, *PKC 2016, Part II*, volume 9615 of *LNCS*, pages 447–462. Springer, Heidelberg, March 2016. doi:10.1007/978-3-662-49387-8_17. (Cited on page 6, 7, 20.)

[LR88] Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM J. Comput.*, 17(2):373–386, 1988. (Cited on page 12.)

[LV16] Huijia Lin and Vinod Vaikuntanathan. Indistinguishability obfuscation from ddh-like assumptions on constant-degree graded encodings. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 11–20, 2016. (Cited on page 1.)

[MSZ16] Eric Miles, Amit Sahai, and Mark Zhandry. Annihilation attacks for multilinear maps: Cryptanalysis of indistinguishability obfuscation over GGH13. In Matthew Robshaw and Jonathan Katz, editors, *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 629–658. Springer, Heidelberg, August 2016. doi:10.1007/978-3-662-53008-5_22. (Cited on page 1.)

[NO14] Phong Q. Nguyen and Elisabeth Oswald, editors. *EUROCRYPT 2014*, volume 8441 of *LNCS*. Springer, Heidelberg, May 2014. (Cited on page 42, 44.)

[OF15] Elisabeth Oswald and Marc Fischlin, editors. *EUROCRYPT 2015, Part II*, volume 9057 of *LNCS*. Springer, Heidelberg, April 2015. (Cited on page 42, 45.)

[O'N10] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/2010/556. (Cited on page 1.)

[PST14] Rafael Pass, Karn Seth, and Sidharth Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In Garay and Gennaro [GG14], pages 500–517. doi:10.1007/978-3-662-44371-2_28. (Cited on page 1.)

[SR15] Rocco A. Servedio and Ronitt Rubinfeld, editors. *47th ACM STOC*. ACM Press, June 2015. (Cited on page 42, 43, 44.)

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10*, pages 463–472. ACM Press, October 2010. (Cited on page 5, 18.)

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005. (Cited on page 1.)

[SW14] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In David B. Shmoys, editor, *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014. (Cited on page 1, 8.)

[Val76] Leslie G. Valiant. Universal circuits (preliminary report). In Ashok K. Chandra, Detlef Wotschke, Emily P. Friedman, and Michael A. Harrison, editors, *Proceedings of the 8th Annual ACM Symposium on Theory of Computing, May 3-5, 1976, Hershey, Pennsylvania, USA*, pages 196–203. ACM, 1976. URL: http://doi.acm.org/10.1145/800113.803649, doi:10.1145/800113.803649. (Cited on page 40.)

[Yao86] Andrew Chi-Chih Yao. How to generate and exchange secrets (extended abstract). In *27th FOCS*, pages 162–167. IEEE Computer Society Press, October 1986. (Cited on page 13.)

[Zim15] Joe Zimmerman. How to obfuscate programs directly. In Oswald and Fischlin [OF15], pages 439–467. doi:10.1007/978-3-662-46803-6_15. (Cited on page 1.)