博士論文

Public-Key Cryptosystems Resilient to Computationally Hard-to-Invert Leakage

（計算量的逆変換困難漏洩耐性を持つ
公開鍵系暗号技術）

Masahito Ishizaka

石坂 理人

Public-Key Cryptosystems Resilient to Computationally Hard-to-Invert Leakage

（計算量的逆変換困難漏洩耐性を持つ公開鍵系暗号技術）

Masahito Ishizaka

石坂 理人

Acknowledgements

I, the author of this thesis, would lilke to express my sincere gratitude to everyone who supported me to complete this thesis.

Firstly, I would like to express the greatest gratitude to my supervisor Prof. Kanta Matsuura. From his advices, statements, attitudes, etc., I have learned how I should be as an active researcher at the front line in the research field. I have attended many international/domestic conferences thanks to the financial supports by him, and through the experiences, I must have grown not only as a researcher, but also as a man. It is also thanks to him that we could communicate with many globally known researchers, namely Prof. Adi Shamir, at the regular meetings.

I sincerely appreciate Prof. Hideki Imai, Prof. Miodrag Mihaljevic and Project Associate Prof. Rie Shigetomi Yamaguchi giving me beneficial advices at the regular meeting from cryptology or information security specialists' points of view. I also greatly appreciate my advisers during the doctoral course, Associate Prof. Yoshihiro Kawahara and Associate Prof. Hidetsugu Irie, giving me a lot of useful advices helping me to complete this thesis.

I would like to express my thanks to Takuro Hosoi who is a technical associate of our laboratory. Thanks to him, we could always agreeably use its room and equipments. I also received many beneficial advices from him regarding my research. I also would like to express my thanks to every secretary of the laboratory, namely Sae Nakano, Kayoko Ogura, Maki Saeki and Yoko Tsuruyama, for supporting me a lot mainly when I attended international/domestic conferences. I also would like to express my thanks to every present member of the laboratory or past one during the doctoral course. Thanks to them, I could acquire broad knowledge on information security.

Finally, I thank my family who have been and will continue to be my spiritual pillar.

Abstract

Cryptographic technologies are essential to make our daily lives more secure and/or more convenient. For instance, in an environment where any sent message can be eavesdropped, by using public-key encryption (PKE) or symmetric-key encryption (SKE), we can send any message without giving no information about the message to the eavesdroppers. And, in an environment where any sent message can be maliciously modified, by using digital signature or message authentication code (MAC), we can send any message in a way such that any modification of the message is detected. Especially, public-key cryptosystems such as PKE and digital signature are convenient because the key-change in advance is no necessary.

The current cryptographic schemes are required to be theoretically secure or provably secure. A reason behind the fact is that there are at least a few examples of cryptographic schemes such that although they are assessed to be secure from specialists' empirical points of view, standardized and widely used, later they were shown to be theoretically insecure. When we do the proof of security, we formally construct a security model, and then reduce the hardness breaking the security model to the hardness solving a mathematical problem which are believed to be hard to solve. For instance, the most desirable security notion of digital signature is one named *strongly existentially unforgeability against chosen message attack*, or *sEUF-CMA* in short. Intuitively, this formalization states that any adversary, who is given a signature-verification-key (public-key) in advance, is successful in forging a signature only with an extremely small probability (i.e., negligible probability), even if he can adaptively use *signing oracle* which takes a message and returns a signature on the message.

In the real world, cryptographic devices are always threatened by secret-information leakage caused by malicious entities. The most serious threat is *Side-Channel Attack (SCA)* which utilizes physical information observed from the device such as power consumption, electromagnetic radiation, acoustic emanation and temperature to identify the secret-key of the device. The standard security models, including sEUF-CMA of digital signature mentioned above, do not consider such information leakage at all. Thus, as soon as 1 bit of such information leaked, they lose their security guarantees.

Leakage-Resilience guarantees that even if some partial related information of the secret-information are leaked, the security is maintained. In security models considering leakage-resilience, we model the information leakage by an efficiently computable function f. The function should be restricted, and various security models which differ in such a restriction have been proposed. Especially, a security model named *Hard-to-Invert Leakage Model (HL Model)* which requires f to be *computationally hard-to-invert* is considered to be theoretically/practically meaningful.

We present solutions to the following three open problems concerning the hard-to-invert leakage-resilience.

The first one is regarding *Identity-Based Encryption (IBE)* with HL-resilience. The IBE scheme proposed by Yuen et al. at EUROCRYPT'12 has been known as the only one claimed to be correctly proven to be secure in a security model with HL-resilience. Firstly, to show that their security proof is defective, we present some concrete counterexamples of adversaries which can be the evidence of the deficiency. Moreover, we propose an original IBE construction and prove that it is secure in a security model considering HL-resilience. As a result, our IBE scheme is the first one whose HL-resilience is correctly proven.

The second one is regarding *Digital Signature* with HL-resilience. We propose a generic construction of digital signature, and show that it is strongly unforgeable, i.e., sEUF-CMA secure, and resilient to polynomially hard-to-invert leakage. Then, we show that it can be instantiated under a standard assumption, namely the decisional linear (DLIN) assumption. Currently, there are some signature schemes proven to be secure in a model considering HL-resilience. We emphasize that our instantiation of signature scheme is not only the first one resilient to polynomially hard-to-invert leakage under standard assumptions, but also the first one proven to be secure in a strong unforgeability model considering HL-resilience.

The third one is regarding *Attribute-Based Signature (ABS)* and *Identity-Based Signature (IBS)* with HL-resilience. Currently, a lot of ABS/IBS schemes secure in a model with no leakage-resilience have been known. However, no scheme proven to be resilient to some leakage under standard assumptions has been known. We propose generic constructions of ABS/IBS schemes and prove that they are secure in HL model. Then, we show that they can be instantiated under the DLIN assumption and the symmetric external Diffie-Hellman (SXDH) assumption. It should be noted that our schemes are the first ones with HL-resilience under standard assumptions, and more generally, the first ones with leakage-resilience under standard assumptions.

Contents

Chapter 1

Introduction

1.1 Background

In the modern society, cryptographic technologies are used everywhere in our daily lives. Without them, current level of security and convenience of the society cannot be achieved. For instance, public-key encryption (PKE) and symmetric-key encryption are the core technologies to realize the secrecy or confidentiality of communication. Digital signature and message authentication code (MAC) are the core primitives to realize the authenticity of digital messages. Not only such fundamental cryptosystems, but also various applied cryptosystems are widely used.

In the past, security of cryptosystems were empirically evaluated. That means that cryptosystems can be assessed to be secure if they cannot be broken by trying various known cryptanalysis techniques. However, there are a few schemes which had been assessed to be empirically secure, standardized and widely used in the society, but later, were found their cryptanalyses, e.g., PKCS #1 v1.5 encryption [Ble98], ISO/IEC 9796-1 signature [CNS99] and ISO/IEC 9796-2 signature [CNS99]. Through the bitter experiences, the current cryptosystems are required to be theoretically secure. The security is generally called as provable security. Intuitively, we do the proof of security by formally constructing a security model, and then proving that breaking the security model is at least as hard as solving a mathematical problem believed to be computationally hard to solve, e.g., Factoring, Discrete Logarithm (DL).

For the case of digital signature, the security proof is done as follows. Practically-ideal security of digital signature is that any entity without the true signature-generation-key (secret-key) cannot forge any valid signature. The most desirable (or the strongest) theoretical model capturing the security is a security notion named *strongly existentially unforgeability against chosen messages attack* or *sEUF-CMA* in short. Its formal definition is given in Sect. 2.9. Informally, the notion means that any adversary, who is given the signature-verification-key (public-key) in advance, succeeds to forge a valid signature only with an extremely small probability, i.e., negligible probability, even if he can adaptively use a signing oracle which takes a message and returns a signature on the message. Here, the adversary is assumed to be *Probabilistic Polynomial Time Algorithm (PPTA)*. And then, we prove that if we assume that there exists an adversary breaking the model, i.e., succeeding forging a signature with a non-negligible probability, we can construct another algorithm solving a mathematical problem such as the DL problem. In this case, the signature scheme remains to be secure as long as the DL problem is computationally hard to solve.

Practically-desirable security of PKE must be something like that any entity without the decryption-key (secret-key) cannot get any information about the original plaintext from its ciphertext. A theoretical model capturing the security is a notion called *Semantic Security (SS)*. However, since SS is difficult to prove, we do not use it for the actual proof, but another notion called *Indistinguishability (IND)* which was proven to be equivalent to SS. Rigorously, based on the strength of adversary, there are some IND-notions. The strongest one is ciphertext-indistinguishability against adaptively chosen ciphertexts attack (IND-CCA) [RS91]. For its formal definition, refer to Sect. 2.6. Informally, that means that any PPT adversary who receives the encryption-key (public-key), chooses two plaintexts, then receives a challenge-ciphertext of randomly chosen one of the two plaintexts, cannot correctly guess the actual plaintext of the ciphertext, even if he can adaptively use decryption oracle, which takes a ciphertext and returns its decryption result, before and after receiving the challenge-ciphertext. For cryptographic schemes other than digital signature and PKE, their security models are properly formalized in a practically-ideal manner.

The security models given above are *blackbox*-wise, which means that any adversary cannot observe the secret-information such as the secret-key. Specifically, in sEUF-CMA notion, although the adversary can acquire *indirect* information about the secret-key through the public-key or the signatures generated on the signing oracle, he can acquire neither *direct* information about the secret-key nor *direct* information about the randomnesses used to generate the signatures. However, in the real world, such information can be leaked because of various causes or attacks. Thus, practically, security models without such assumptions are more desirable. Specifically, leakage-pattern of secret-information can be categorized into *Full Leakage* which is a pessimistic pattern and *Related-Information Leakage* which is a more realistic pattern.

Full leakage can occur because of various causes. For instance, virus (malware) infection, non-thorough key-management, and intention by malicious insiders, can be the cause. There are some security models considering the situation where the secret-key is fully leaked. For instance, *Threshold Security* [Des87, DF89] divides a secret-key into multiple shares and guarantees that the security is maintained as long as the number of shares fully leaked is less than a constant number. *Forward Security* [And97, BM99] considers a situation where secret-keys are periodically updated and during an arbitrary period t, the secret-key is fully leaked, and guarantees that the security for the previous periods $i \in [1, t-1]$ is maintained. *Key-Insulated Security* [DKXY02] assumes that information needed to update the secret-key is stored in a *leak-free* device and guarantees that even if the secret-key, stored in a non-leak-free device, is fully leaked at arbitrary periods t, the security for the other periods $\neq t$ is maintained.

In the real world, related-information leakage is more possible and more hard to prevent than full leakage. Of course, related-information leakage can occur because of the same reasons as full leakage, such as virus infection and intention by malicious insiders. Another serious cause is *Side-Channel Attack (SCA)* which acquires information about the secret-key of the device through the side-channels. Specifically, many SCAs observe physical information from the device including related-information about the secret-key, and utilize the information to identify the secret-key. More specifically, *Timing Attack* [Koc96, Sch00] utilizes the running time consumed to process a cryptographic operation such as decrypting or signing. *Power Analysis Attack* [KJJ99, Cor99, MDS99, GPT14] utilizes the power consumption of the device during processing a cryptographic operation. The other known SCAs utilizes physical information such as electromagnetic (EM) radiation [GMO01, AARR02, GPT14], acoustic emanation [ST04, GST14] and temperature [HS13] of the device. In addition to SCA, *Cold-Boot Attack* [HSH⁺08], which identifies the secret-key by utilizing a device's characteristic such that, for some time after its power down, information about the secret-key remains in its memory, is also a serious threat of related-information leakage.

For some SCAs, countermeasures against the attacks are known. Some countermeasures are algorithmic or software-based, e.g., masking, blinding. Some other ones are physical or hardware-based, e.g., shielding. It should be noted that any countermeasure accompanies some negative effects such as degradation of efficiency such as computational cost and increasing of monetary cost. Although various SCAs were proposed, it must be true that new SCAs are continuously proposed in the future. Thus, even if we take every known countermeasure to realize a cryptosystem, there is no guarantee that it is secure against even SCAs which will be found in the future. Moreover, some physical information are almost impossible to completely prevent by any countermeasure. Thus, we obtain the following conclusion: Even if we take every known countermeasure to realize a cryptosystem with accompanying some efficiency degradation and monetary cost increasing, it must be impossible for us to perfectly prevent *all* SCAs including currently known SCAs and SCAs found in the future, i.e., impossible to make the leakage caused by *all* SCAs zero.

1.2 Leakage-Resilient Cryptosystems

Leakage-Resilience is a property that guarantees that even if some related-information about secret-information such as the secret-key is leaked because of various causes such as side-channel attack (SCA), the security is maintained. Our common ultimate goal in the community is that we achieve the security against as many types of possible related-information leakage as possible.

In security models considering leakage-resilience, it is common that a side-channel attack is modelled as an efficiently computable (i.e., polynomial time computable) function f. Precisely, each adversary in a security model can arbitrarily choose such a function f and learn leakage information $f(sk)$, then utilizes it to try to break the model. Obviously, if the adversary is allowed to choose a function which reveals the correct secret-key such as the identity map, he never fails to break the model. So, a restriction on the function f is required. Such a restriction differs in each one of the models. Our goal is making a scheme secure in a model with a looser restriction on f .

We are mainly interested in a leakage-resilient security model called *hard-to-invert leakage (HL) model*. We give a explanation for the model in Subsect. 1.2.4. Before that, we broadly categorize existing models which are closely related to HL model into three models, namely *bounded leakage model*, *continual leakage model* and *noisy leakage model*, and introduce each model in Subsect. 1.2.1, Subsect. 1.2.2 and Subsect. 1.2.3, respectively. Specifically speaking, for each model in the subsection, we explain the *restriction* on the leakage function f, relation among some models including the model, and some important known results on the model.

1.2.1 Bounded Leakage (BL) Model

Bounded Leakage (BL) Model was firstly presented by Akavia et al. at TCC'09 [AGV09]. In BL model, the output's bit-length of the function f is restricted by a variable $l(k)$ called *leakage bound* [1] which is smaller than the actual bit-length of the secret-key $|sk|$. Formally, each adversary can choose as f only function which satisfies a condition that $f : \{0, 1\}^{|sk|} \to \{0, 1\}^{l(k)}$, where $l(k) < |sk|$. The cold-boot attack [HSH+08] which had been presented right before the work [AGV09] motivated the authors of [AGV09] to invent the model. *Only Computation Leaks Information (OCLI)* security model [MR04] cannot deal with the cold-boot attack [HSH+08] since the model assumes that only computation leaks information and no leakage occurs from memory unrelated to the computation. On the other hand, BL model can be secure against wider class of side-channel attacks including the cold-boot attack.

Given a cryptographic scheme secure in BL model, we define its *leakage-ratio* of secret-key as $l(sk)/|sk|$. We say that the leakage-ratio is *optimal* if it is written as $1 - o(1)$. The leakage-ratio of secret-key is an information whom we use to compare some schemes secure in BL model each other. Also, for instance, the following information can be used for the comparison. Firstly, if it is a scheme such as IBE, ABE, IBS or ABS, whether the master-key can be leaked, or more generally, how large is the leakage-ratio for it. Secondly, whether leakage from the randomness used to generate the secret-key and/or master-key is allowed, or more generally, how large is the leakage-ratio for each one of them. Thirdly, if it is a digital signature scheme such as digital signature, IBS or ABS, whether the scheme is *Fully Leakage-Resilient (FLR) [KV09]* [2].

Akavia et al. [AGV09] proved that PKE schemes such as [Reg05] and [GPV08] are secure in BL model. Because of the simple or easy-to-understand definition of the model, until the present time, (at least) hundreds of schemes proven to be secure in the model have been proposed. Some examples, including schemes secure in *bounded retrieval (BR) model* or *continual leakage (CL) model*, are given below.

PKE [AGV09, NS09, BG10, BKKV10, DHLAW10b, LLW11, QL13, QL14, CQX18], digital signature [KV09, BKKV10, DHLAW10a, DHLAW10b, BSW11, GJS11, KKS11, LLW11, MTVY11, FNV15], message authentication code (MAC) [CQX18], IBE [CDRW10, LRW11, KP13, LTZY16], ABE [LRW11, ZSW+13, Zha14], functional encryption (FE) [YAX+16, WCLH18], inner-product encryption (IPE) [KP13], zero-knowledge proof [GJS11, Pan14, Kiy15], identification (ID) [DHLAW10a], authenticated key agreement (AKA) [DHLAW10a], multiparty computation [BGJK12], secret sharing [BDIR18] and fully homomorphic encryption [BL14] have been proposed.

[1] In Sect. 1.2, k denotes the minimum entropy of the secret-key sk.

[2] In a broad sense, we say that a signature scheme is fully leakage-resilient (FLR), if it permits not only leakage from the secret-key, but also leakage from the randomnesses used to generate signatures on the signing oracle. In a narrow sense, full leakage-resilience (FLR) means that the schemes allows all leakage from the secret-key, the randomnesses used to generate signatures *and* the randomness used to generate the secret-key.

We obtain the following notable facts. Firstly, as schemes achieving the optimal leakage-ratio, PKE schemes [QL13, QL14, CQX18], IBE schemes [KP13] and digital signature schemes [BSW11] were proposed. Secondly, as schemes achieving the strongest security, IND-CCA secure PKE schemes [NS09, QL13, QL14, CQX18], adaptive IND-CCA secure IBE scheme [LTZY16] and sEUF-CMA secure digital signature schemes [WT14, HHP16] are known. Thirdly, FLR signature schemes were proposed in [BSW11, GJS11, MTVY11, FNV15]. Fourthly, as schemes secure against master-key leakage, IBE scheme and ABE scheme were proposed in [LRW11].

Bounded Retrieval (BR) Model. The terminology *Bounded Retrieval (BR) Model* initially appeared in the works [DCLW06, Dzi06] which proposed protocols secure against intrusion attacks. The first public-key cryptosystem secure in BR model was proposed by Alwen et al. at CRYPTO'09 [ADW09]. Specifically, they proposed a few schemes including digital signature secure in the random oracle model. Subsequently, Alwen et al. [ADN$^+$10] proposed IBE schemes secure in BR model at Eurocrypt'10.

Alwen et al. [ADW09, ADN$^+$10] defined public-key cryptographic schemes secure in BR model as bounded leakage-resilient schemes which satisfy the condition such that we can increase its leakage bound $l(k)$ only by increasing the size of secret-key, where no degradation of the other efficiency measures, such as public-key size, ciphertext size, signature size, decryption cost and signing cost, occurs. Also, they mentioned that schemes secure in BR model are useful, since they can be secure against hacking or malware attacks.

IBE schemes secure in BR model were proposed in [ADN$^+$10, CZLC16, HLAWW16]. Among the IBE schemes, there is no one which simultaneously achieves the optimal leakage-ratio and security in the standard model. So, presenting such a scheme is an open problem. Signature schemes secure in BR model were proposed in [ADW09, FNV15]. Among the signature schemes, there is no one which simultaneously achieves the optimal leakage-ratio and FLR in the standard model. Thus, presenting such a scheme is also an open problem.

1.2.2 Continual Leakage (CL) Model

Continual Leakage (CL) Model was independently proposed by Brakerski et al. [BKKV10] and Dodis et al. [DHLAW10a] at FOCS'10 as a generalization of BL model.

In BL model, total amount of leakage is bounded by the leakage bound $l(k)$. However, as soon as more information than the bound is leaked, its security guarantee is lost. On the other hand, in CL model, total amount of leakage is unbounded. In the model, we consider a situation where the secret-key is updated periodically. We assume that there are t periods in total and express the secret-key at i-th period as sk_i, where $i \in [1, t]$. For every $i \in [1, t]$, the model permits sk_i to leak any information about the key as long as bit-length of total amount of the leakage is less than $l(k)$. Note that total number of update of secret-key is unbounded. As a result, total amount of leakage is also unbounded.

In Subsect. 1.2.1, we introduced some indexes which are useful when we compare some schemes secure in BL model each other, such as whether its leakage-ratio is optimal. Every one of the indexes is also useful when we decide whether a scheme secure in CL model is superior to the other one. Additionally, the following indexes are also useful. The first one is whether the master-key can be updated. The second one is whether leakage from the randomness used to update the secret-key and/or the master-key is considered.

Various schemes secure in CL model have been proposed. Specifically, PKE [BKKV10, LLW11], digital signature [BKKV10, DHLAW10a, BSW11, GJS11, KKS11, LLW11, WT15], IBE [BKKV10, LRW11, KP13], ABE [LRW11, ZCG$^+$18], functional encryption (FE) [YAX$^+$16], IPE [KP13], Identification [DHLAW10a], Authenticated Key Agreement [DHLAW10a] and IBS [WTH16] have been proposed.

Floppy Model, a.k.a. Invisible Key-Update Model. In a broad sense, *Floppy Model*, a.k.a. *Invisible Key-Update Model* [ADW09], is categorized as a CL model. Rigorously, the model is based on a stronger assumption than the normal CL model. Specifically, in the model, it is assumed that we can prepare a *leak-free* device in whom some information needed to update the secret-key are stored.

Although the strong assumption of the model can be a big disadvantage for schemes secure in the model, there have been proposed some schemes secure in the model with good properties whom any previous scheme secure in CL model has not achieved. For instance, PKE schemes by Agrawal et al. [ADVW13] proven to be secure in the floppy model under DDH assumption and achieve the optimal leakage-ratio are schemes which do not utilize bilinear groups (pairing).

Introducing Forward Security into CL Model. CL model has a disadvantage. Specifically, the model assumes that the situation where the secret-key is entirely leaked at a period t never occurs. However, in the real world, such a situation can happen. As soon as such a situation occurs, any scheme secure in the model loses its security guarantee for all periods.

Forward Security [And97, BM99] guarantees that for every polynomial integer $t \in \mathbb{N}$, even if secret-key at period t is revealed entirely, its security guarantee from period 1 to $t-1$ is maintained. The model also has a disadvantage such that it assumes that all secret-keys during period 1 to $t-1$ do not leak any information about the keys. Thus, the model loses security guarantee in a case where at least one of the secret-keys during period 1 to $t-1$ leaks some information about the keys and the secret-key at period t leaks its entire information.

Bellare et al. [BOS17] introduced forward security into CL model to make a model without each one of such disadvantages. Specifically, their model guarantees that for every polynomial integer $t \in \mathbb{N}$, even if every secret-key during period 1 to $t-1$ leaks its partial information whose bit-length is less than $l(k)$ and the secret-key at period t leaks its full information, the security is maintained. By the way, they proposed PKE and digital signature schemes secure in the model.

1.2.3 Noisy Leakage (NL) Model

We remind us that BL model and CL model assume that bit-length of leakage from the secret-key is smaller than the leakage bound $l(k)$ s.t. $l(k) < |sk|$. We consider whether the assumption is realistic. It must be reasonable to conjecture that most of real side-channel attacks, especially ones utilizing physical information such as running time [Koc96, Sch00], power consumption [KJJ99, Cor99, MDS99, GPT14], electromagnetic radiation [GMO01, AARR02, GPT14], acoustic emanation [ST04, GST14] and temperature [HS13], leaks a great amount of information, whose bit-length can be a lot bigger than $|sk|$. According to [Sta11], some side-channel attacks such as differential power analysis [KJJ99] can leak information on the order of gigabits per second. Because of the reasons, we cannot help saying that the assumption of BL model is unrealistically strong.

Noisy Leakage (NL) Model was invented as a generalization of BL model. The model removes the restriction of BL model on bit-length of f's output. Formally, in NL model, the function f must be a function $f : \{0,1\}^{|sk|} \rightarrow \{0,1\}^*$ such that the loss of minimum entropy of sk caused by f is smaller than $l(k) < k$ [3]. Obviously, $l(k)$ cannot be $l(k) \geq k$ since that means that f can be a function revealing sk such as the identity map. Thus, as BL model, every function f in NL model does not information-theoretically reveal sk.

The concept of NL model was initially presented by Dziembowski and Pietrzak at FOCS'08 [DP08], and symmetric-key encryption scheme secure in the model was given. Subsequently, Naor and Segev at CRYPTO'09 [NS09] presented the first PKE scheme secure in NL model by proving that their PKE scheme proven to be secure in BL model is also secure in NL model. Signature schemes secure in NL model and BL model were proposed in [GJS11, FNV15]. Another signature scheme whose security in NL model and BR model was guaranteed in the random oracle model was proposed in [FNV15].

1.2.4 Hard-to-Invert Leakage (HL) Model

We remind us that in NL model [DP08, NS09] (or BL model [AGV09]), f cannot information-theoretically determine the secret-key sk. Based on the assumption, we evaluate NL model (and BL model) from practical/theoretical points of view.

First, we consider whether the assumption is reasonable from practical point of view. According to [Sta11], some practical side-channel attacks can information-theoretically determine the secret-key sk. In that sense, the assumption is unrealistic.

Next, we theoretically evaluate the assumption. It is obvious that, because of the assumption, functions which information-theoretically reveal sk such as *one-way permutation (OWP)* are prohibited for the adversary to query. Theoretically, a model with a looser restriction on f than NL model (and BL model), such that such functions like OWP are allowed for the adversary to query, is more desirable.

To eliminate such an assumption of NL model (or BL model), or to generalize the models, *Hard-to-Invert Leakage (HL) Model*, a.k.a. *Auxiliary (Input) Leakage (AL) Model*, was invented by Dodis et al. at STOC'09 [DKL09]. The model requires f to be *hard-to-invert*. Formally, f must satisfy a condition that no probabilistic polynomial time algorithm, given $f(sk)$, can find sk with a probability larger than $\mu(k)$, where $\mu(\cdot)$, whom we call *auxiliary parameter*, is a negligible function satisfying $\mu(k) > 2^{-k}$. Note that the larger the parameter $\mu(k)$ is, the larger the set of leakage-functions is. Based on the parameter $\mu(k)$, we consider three categories for the function, namely *polynomially/sub-exponentially/exponentially* hard-to-invert functions, whose formal definitions are given in Sect. 2.4.

[3]Note that meaning of the function $l(\cdot)$ differs between BL model and NL model. In BL model, it denotes maximum bit-length of f. In NL model, it denotes maximum loss of minimum entropy caused by f.

Dodis et al. [DKL09] proposed symmetric-key encryption schemes secure in HL model. Subsequently, the first PKE schemes secure in the model were proposed by Dodis et al. at TCC'10 [DGK$^+$10]. The other PKE schemes were proposed in [BG10, SHGL16]. Note that the above PKE schemes were proven to be secure against *chosen plaintext attacks (CPA)*. A methodology to transform any CPA-secure PKE scheme in HL model into one secure against chosen ciphertext attacks (CCA) was proposed in [ZCQ12].

The first IBE scheme (claimed to be correctly proven) secure in HL model was proposed by Yuen et al. at Eurocrypt'12 [YCZY12]. ABE schemes have been proposed by Zhang et al. [ZWTM13] and [WY15]. The first digital signature scheme secure in HL model was proposed by Faust et al. at Asiacrypt'12 [FHN$^+$12]. The other digital signature schemes have been proposed in [YYH12, WMHT16].

1.3 Contributions

We can fairly say that it has been our common knowledge that HL (hard-to-invert leakage) model is practically and theoretically the most desirable one among various models considering leakage-resilience. In this thesis, we give concrete solutions for some especially meaningful open problems regarding the model, whose details are described in the latter half of this section. In the research field on leakage-resilient cryptography, until the present time, BL (bounded leakage) model has been the hottest topic. It is just a fact that at the current time, papers on BL model greatly outnumber ones on the other models. One of the reasons behind the fact must be that the simplicity of the definition of the model allows us to use it the most easily. As we explained in the previous sections, HL model and BL model are neither independent nor unrelated. The former one is a generalization of the latter one as said in [DGK$^+$10], and actually, most of schemes proven to be secure in the former one, including some schemes introduced in this thesis, are proven to be secure in the latter one. We think that we can fairly say that our works are very meaningful, because it is highly possible that they further develop not only the research field on hard-to-invert leakage-resilience, but also the research field on leakage-resilience on the whole.

Specifically, we consider three open problems and present concrete solutions for them, as follows.

IBE with HL-Resilience [B2, C2]. One of the identity-based encryption (IBE) schemes proposed by Yuen et al. at Eurocrypt'12 [YCZY12] has been believed to be the only one whose hard-to-invert leakage-resilience was correctly proven. Firstly, we show that their security proof is defective. Specifically, we introduce some concrete counterexamples which can be the evidence for the deficiency. Thus, constructing IBE schemes secure in HL model is an open problem. Actually, we construct a concrete IBE scheme which is a modified variant of the IBE scheme proposed by Kurosawa and Phong [KP13] and proven to be secure in BL model. Then, we prove that our IBE scheme is secure in HL model. As a result, our IBE scheme is the first one which is correctly proven to be secure in HL model. A talk on this work was presented at CANS2018 [B2], and a talk on a work closely related to this work was presented at SCIS2017 [C2].

Digital Signature with HL-Resilience [B1, C6, D1, E2]. Digital signature schemes secure in HL model have been proposed by Faust et al. at Asiacrypt'12 [FHN$^+$12] and the other ones [YYH12, WMHT16]. By modifying the result by Faust et al., we generically construct a signature, and prove that it simultaneously achieves the strong unforgeability and polynomially hard-to-invert leakage-resilience. Then, we instantiate it to give a concrete construction secure under a standard assumption, namely the decisional linear (DLIN) assumption. We emphasize that the instantiation of digital signature is *not only* the first one which is resilient to polynomially hard-to-invert leakage under standard assumptions, *but also* the first one which is strongly unforgeable in HL model. A talk on this work was presented at ISC2018 [B1], and a talk on a work closely related to this work is going to be presented at SCIS2019 [C6].

ABS/IBS with HL-Resilience [C3, C5, E1]. Although various identity-based signature (IBS) schemes [PS06] and attribute-based signature (ABS) schemes [MPR11, OT11, SAH16] secure in the non-leakage setting have been proposed, IBS or ABS scheme proven to have some leakage-resilience under standard assumptions has not been proposed. We generically construct IBS scheme, and prove that it is existentially unforgeable in HL model. We generically construct ABS scheme whose predicate is represented as a general circuit, and prove that it is existentially unforgeable in HL model and *computationally* signer-private. Then, we instantiate them under standard assumptions, namely the DLIN assumption and the symmetric external Diffie-Hellman (SXDH) assumption. We emphasize that they are the first ones secure in HL model under standard assumptions, and more generally, the first leakage-resilient ones under standard assumptions. Talks on works closely related to this work were presented at CSS2017 [C3] and CSS2018 [C5].

1.4 Organization of This Thesis

Except for the fist chapter, this thesis has five chapters. As we introduced in the last section, we have three contributions. The details of the contributions are given in Chapter 3, Chapter 4 and Chapter 5, respectively. Chapter 2 is *Preliminaries*, where some pieces of information which are necessary or useful for the reader to comprehend our works are given. Specifically, notations whose meanings are not necessarily obvious for the reader, and syntaxes and definitions of security notions or properties of some cryptographic schemes such as digital signature, IBE, IBS and ABS, are given. Chapter 6 is the conclusion of this thesis.

Chapter 2

Preliminaries

2.1 Basic Notations

For $a, b \in \mathbb{N}$, $[a, b]$ denotes $\{x \in \mathbb{N} \mid a \leq x \leq b\}$. For $\lambda \in \mathbb{N}$, 1^λ denotes a security parameter. We say that a function $h : \mathbb{N} \to \mathbb{R}$ is negligible if for every $c \in \mathbb{N}$, there exists $x_0 \in \mathbb{N}$ such that $h(x) \leq x^{-c}$ for every $x \geq x_0$. $\mathcal{G}$ is a function which takes 1^λ as input, and randomly outputs $(p, \mathbb{G}, g)$, where p is a prime number whose bit-size is λ, $\mathbb{G}$ is a multiplicative cyclic group whose order is p, and g is a generator of $\mathbb{G}$. PPTA means probabilistic polynomial time algorithm. For a set A, $a \overset{\text{U}}{\leftarrow} A$ indicates a procedure which we extract an element a from A uniformly at random. Given a matrix $\mathbf{A}$ of size $m \times n$ whose (i, j)-th element is denoted by $a_{i,j}$, $g^{\mathbf{A}}$ denotes a matrix of size $m \times n$ whose (i, j)-th element is $g^{a_{i,j}}$. $\mathbf{I}_m$ denotes the identity matrix of size m. For a set A, let $|A|$ denote the bit-length of each element of the set. For $n \in \mathbb{N}$, 0^n denotes the bit-string composed of n number of 0.

2.2 Bilinear Groups of Prime Order

$\mathcal{G}_{pg}$ denotes a generator of (asymmetric) bilinear pairing with groups of prime order. $\mathcal{G}_{pg}$ takes 1^λ, where $\lambda \in \mathbb{N}$, as input, and outputs $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g})$, where p is a prime whose bit-length is λ, $\mathbb{G}$, $\tilde{\mathbb{G}}$ and $\mathbb{G}_T$ are multiplicative groups of order p, g is a generator of $\mathbb{G}$, $\tilde{g}$ is a generator of $\tilde{\mathbb{G}}$, and $\hat{e} : \mathbb{G} \times \tilde{\mathbb{G}} \to \mathbb{G}_T$ is a map which is computable in polynomial time and satisfies the following conditions: $\hat{e}(g, \tilde{g})$ generates $\mathbb{G}_T$. For every $a, b \in \mathbb{Z}_p$, $\hat{e}(g^a, \tilde{g}^b) = \hat{e}(g, \tilde{g})^{ab}$.

2.3 Hardness Assumptions

2.3.1 Discrete Logarithm (DL) Assumption

For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. DL assumption holds (on the group $\mathbb{G}$), if there exists a negligible function $negl(\lambda)$ s.t. for every PPT $\mathcal{A}$, it holds that $\Pr[x \leftarrow \mathcal{A}(p, \mathbb{G}, g, g^x) \mid x \overset{\text{U}}{\leftarrow} \mathbb{Z}_p] < negl(\lambda)$.

2.3.2 Decisional Linear (DLIN) Assumption [BBS04]

For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. DLIN assumption holds (on the group $\mathbb{G}$), if there exists a negligible function $negl(\lambda)$ s.t. for every PPT $\mathcal{A}$, it holds that

$$\Big|\Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{r_1+r_2}) \mid g_1, g_2, g_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}, r_1, r_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p]$$
$$- \Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{u}) \mid g_1, g_2, g_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}, r_1, r_2, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p]\Big| < negl(\lambda).$$

2.3.3 Decisional Diffie-Hellman (DDH) Assumption

For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. DDH assumption holds (on the group $\mathbb{G}$), if there exists a negligible function $negl(\lambda)$ s.t. for every PPT $\mathcal{A}$, it holds that

$$|\Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g, g^{r_1}, g^{r_2}, g^{r_1 r_2}) \mid r_1, r_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p]$$
$$- \Pr[1 \leftarrow \mathcal{A}(p, \mathbb{G}, g, g^{r_1}, g^{r_2}, g^{u}) \mid r_1, r_2, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p]| < negl(\lambda).$$

2.3.4 Symmetric External Diffie-Hellman (SXDH) Assumption

For $\lambda \in \mathbb{N}$, let $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$. SXDH assumption holds (on the groups $\mathbb{G}$ and $\tilde{\mathbb{G}}$), if the DDH assumption holds on both of the groups $\mathbb{G}$ and $\tilde{\mathbb{G}}$.

2.4 Polynomially/Exponentially Hard-to-Invert Functions

Following [FHN+12], we define *polynomially/(sub-)exponentially* hard-to-invert functions. We consider an efficiently computable function $h : \mathcal{R} \to \{0,1\}^*$. Let $x \in \mathcal{R}$ denote input variable of h. Let k denote minimum entropy of x which is randomly chosen from $\mathcal{R}$.

Definition 1. *h is a polynomially hard-to-invert leakage function, if there exists a negligible function $negl(\cdot)$ s.t. for every PPT $\mathcal{B}$, it holds that* $\Pr[x \leftarrow \mathcal{B}(h(x)) \mid x \overset{\mathrm{R}}{\leftarrow} \mathcal{R}] < negl(k)$.

Definition 2. *h is a sub-exponentially hard-to-invert leakage function, if there exists a constant $1 > c > 0$ s.t. for every PPT $\mathcal{B}$, it holds that* $\Pr[x \leftarrow \mathcal{B}(h(x)) \mid x \overset{\mathrm{R}}{\leftarrow} \mathcal{R}] < 2^{-k^c}$.

Definition 3. *h is an exponentially hard-to-invert leakage function, if there exists a constant $c > 0$ s.t. for every PPT $\mathcal{B}$, it holds that* $\Pr[x \leftarrow \mathcal{B}(h(x)) \mid x \overset{\mathrm{R}}{\leftarrow} \mathcal{R}] < 2^{-c \cdot k}$.

2.5 Goldreich-Levin Theorem for Large Fields [DGK+10]

The following Goldreich-Levin theorem for large fields was proven by Dodis et al. [DGK+10].

Theorem 2.5.1. *Let p denote a prime number. Let H denote an arbitrary subset of $GF(p)$. $f : H^{m\times 1} \to \{0,1\}^*$ is an arbitrary function. If there is a distinguisher $\mathcal{D}$ running in time t such that*

$$\begin{aligned}\delta \ := \ \Big| &\Pr\Big[\mathcal{D}(f(\mathbf{s}), \mathbf{r}, \mathbf{rs}) = 1 \mid \mathbf{s} \overset{\text{U}}{\leftarrow} H^{m\times 1}, \mathbf{r} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}\Big] - \\ &\Pr\Big[\mathcal{D}(f(\mathbf{s}), \mathbf{r}, u) = 1 \mid \mathbf{s} \overset{\text{U}}{\leftarrow} H^{m\times 1}, \mathbf{r} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\text{U}}{\leftarrow} \mathbb{Z}_p\Big]\Big|,\end{aligned}$$

then there is an inverter $\mathcal{B}$ running in time $t' = t \cdot \mathrm{poly}(m, |H|, 1/\delta)$ such that

$$\Pr\Big[\mathcal{B}(f(\mathbf{s})) = \mathbf{s} \mid \mathbf{s} \overset{\text{U}}{\leftarrow} H^{m\times 1}\Big] \geq \frac{\delta^3}{512 \cdot m \cdot p^2}.$$

2.6 Public-Key Encryption (PKE)

Syntax of PKE. Public-key encryption (PKE) consists of the following 3 polynomial-time algorithms, where Gen and Enc are probabilistic and Dec is deterministic.

$\mathsf{Gen}(1^\lambda) \to (pk, sk)$. Key-generation algorithm[1] takes a security parameter 1^λ as an input, and outputs a public-key pk and a secret-key sk. pk determines plaintext space $\mathcal{M}$ uniquely.

$\mathsf{Enc}(pk, M) \to C$. Encryption algorithm takes pk and a plaintext $m \in \mathcal{M}$ as inputs, and outputs a ciphertext C.

$\mathsf{Dec}(sk, C) \to M \ / \perp$. Decryption algorithm[2] takes sk and a ciphertext C as inputs, and outputs a plaintext M or $\perp$, where $\perp$ indicates a failure of decryption.

Every PKE scheme must be correct. An PKE scheme $\Sigma_{PKE} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$ is correct, if for every $\lambda \in \mathbb{N}$, every $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, every $M \in \mathcal{M}$ and every $C \leftarrow \mathsf{Enc}(pk, M)$, it holds that $\Pr[M \leftarrow \mathsf{Dec}(sk, C)] = 1$.

Standard Security Notions of PKE. To define ciphertext indistinguishability against adaptively chosen ciphertexts attack (IND-CCA) for a PKE scheme $\Sigma_{PKE} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$, we need the following game which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$.

Key-Generation. $\mathcal{CH}$ runs $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, and sends pk to $\mathcal{A}$.

Query. $\mathcal{A}$ is allowed to use the decryption oracle **Decrypt** adaptively.

Decrypt(C): $\mathcal{A}$ queries a ciphertext $C \in \mathcal{C}$. $\mathcal{CH}$ returns $M \ / \perp \leftarrow \mathsf{Dec}(sk, C)$.

[1] In this paper, the pair of keys (pk, sk) can be substituted by another pair of keys (ek, dk), where ek is an encryption-key and dk is a decryption-key.

[2] Although Dec needs the public-key pk or the encryption-key ek as an input since the key includes information such as the prime p and the group $\mathbb{G}$, we often omit the key as the input.

Challenge(M_0, M_1)**.** $\mathcal{A}$ sends two plaintexts $M_0, M_1 \in \mathcal{M}$. $\mathcal{CH}$ sets $b \overset{U}{\leftarrow} \{0, 1\}$, then returns $C^* \leftarrow \mathsf{Enc}(pk, M_b)$.

Query 2. $\mathcal{A}$ is allowed to use the decryption oracle **Decrypt** adaptively.

> **Decrypt**(C)**:** $\mathcal{A}$ queries a ciphertext $C \in \mathcal{C}$ which is not the challenge ciphertext C^*. $\mathcal{CH}$ returns M / $\perp \leftarrow \mathsf{Dec}(sk, C)$.

Guess(b')**.** $\mathcal{A}$ sends $b' \in \{0, 1\}$ to $\mathcal{CH}$.

Definition 4. *A PKE scheme Σ_{PKE} is IND-CCA secure if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{IND\text{-}CCA}_{\mathcal{A},\Sigma_{PKE}}(\lambda) = |2 \cdot \Pr[b' = b] - 1|$ is negligible.*

Remark. Weaker security notion named *ciphertext indistinguishability against chosen plaintexts attack (IND-CPA)* is defined basically in the same manner as IND-CCA except that the adversary in the security game cannot use the decryption oracle in each one of the phases **Query** and **Query 2**.

2.7 Labeled Public-Key Encryption (LPKE)

Syntax of LPKE. Labeled public key encryption (LPKE) consists of three polynomial time algorithms. Gen and Enc are probabilistic. Dec is deterministic.

$\mathsf{Gen}(1^\lambda) \to (pk, sk)$**.** Key-generation algorithm[1] takes 1^λ as input, and outputs a public-key pk, and a secret-key sk. Plaintext space $\mathcal{M}$, ciphetext space $\mathcal{C}$, and label space $\mathcal{L}$ are uniquely determined by pk.

$\mathsf{Enc}(pk, M, L) \to C$**.** Encryption algorithm takes pk, a plaintext $M \in \mathcal{M}$, and a label $L \in \mathcal{L}$ as inputs, and outputs a ciphertext C.

$\mathsf{Dec}(sk, C, L) \to M$ / $\perp$**.** Decryption algorithm[2] takes sk, a ciphetext $C \in \mathcal{C}$, and a label $L \in \mathcal{L}$ as inputs, and outputs a plaintext M or $\perp$.

Every LPKE scheme must be correct. An LPKE scheme $\Sigma_{LPKE} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$ is correct, if for every $\lambda \in \mathbb{N}$, every $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, every $M \in \mathcal{M}$, every $L \in \mathcal{L}$, and every $C \leftarrow \mathsf{Enc}(pk, M, L)$, it holds that $\Pr[M \leftarrow \mathsf{Dec}(sk, C, L)] = 1$.

Standard Security Notions of LPKE. To define weak ciphertext indistinguishability against adaptively chosen label and ciphertexts attacks (IND-wLCCA) for an LPKE scheme $\Sigma_{LPKE} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$, we use the following game which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$.

Key-Generation. $\mathcal{CH}$ runs $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, and sends pk to $\mathcal{A}$.

Query. $\mathcal{A}$ is allowed to use the decryption oracle **Decrypt** adaptively.

$\mathtt{Decrypt}(C, L)$: $\mathcal{A}$ queries a ciphertext $C \in \mathcal{C}$ and a label $L \in \mathcal{L}$. $\mathcal{CH}$ returns M / $\perp \leftarrow \mathsf{Dec}(sk, C, L)$.

$\mathtt{Challenge}(M_0, M_1, L^*)$. $\mathcal{A}$ sends two plaintexts $M_0, M_1 \in \mathcal{M}$, and a label $L^* \in \mathcal{L}$. $\mathcal{CH}$ sets $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$, then returns $C^* \leftarrow \mathsf{Enc}(pk, M_b, L^*)$.

$\mathtt{Query\ 2}$. $\mathcal{A}$ is allowed to use the decryption oracle **$\mathtt{Decrypt}$** adaptively.

$\mathtt{Decrypt}(C, L)$: $\mathcal{A}$ queries a ciphertext $C \in \mathcal{C}$ and a label $L \in \mathcal{L}$ such that $L \neq L^*$. $\mathcal{CH}$ returns M / $\perp \leftarrow \mathsf{Dec}(sk, C, L)$.

$\mathtt{Guess}(b')$. $\mathcal{A}$ sends $b' \in \{0, 1\}$ to $\mathcal{CH}$.

Definition 5. *An LPKE scheme Σ_{LPKE} is IND-wLCCA secure if for any PPT adversary $\mathcal{A}$, $Adv^{\mathtt{IND\text{-}wLCCA}}_{\mathcal{A},\Sigma_{LPKE}}(\lambda) = |2 \cdot \Pr[b' = b] - 1|$ is negligible.*

Remark. Stronger notion named *strong ciphertext indistinguishability against adaptively chosen label and ciphertexts attack (IND-LCCA)* is defined basically in the same manner as IND-wLCCA except that the adversary can query (C, L) to the decryption oracle at **$\mathtt{Query\ 2}$** if $C \neq C^* \vee L \neq L^*$.

2.8 Identity-Based Encryption (IBE)

Syntax of IBE. Identity-based encryption (IBE) consists of the following 4 polynomial-time algorithms, where Dec is deterministic and the others are probabilistic.

$\mathsf{Setup}(1^\lambda) \rightarrow (pk, mk)$. Setup algorithm takes a security parameter 1^λ as an input, and outputs a system public-key pk and a master-key mk. pk determines the ID space $\mathcal{I}$ and plaintext space $\mathcal{M}$ uniquely.

$\mathsf{KeyGen}(pk, mk, ID) \rightarrow sk$. User's secret-key generation algorithm takes pk, mk, and $ID \in \mathcal{I}$ as inputs, and outputs an secret-key sk for ID.

$\mathsf{Enc}(pk, M, ID) \rightarrow C$. Encryption algorithm takes pk, a plaintext $M \in \mathcal{M}$, and $ID \in \mathcal{I}$ as inputs, and outputs a ciphertext C.

$\mathsf{Dec}(pk, C, sk) \rightarrow M$ / $\perp$. Decryption algorithm[3] takes pk, a ciphertext C, and a secret-key sk as inputs, and outputs a plaintext M or $\perp$, where $\perp$ indicates a failure of decryption.

Every IBE scheme must be correct. An IBE scheme $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ is correct, if for every $\lambda \in \mathbb{N}$, every $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, every $ID \in \mathcal{I}$, every $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, every $M \in \mathcal{M}$ and every $C \leftarrow \mathsf{Enc}(pk, M, ID)$, it holds that $\Pr[M \leftarrow \mathsf{Dec}(pk, C, sk)] = 1$.

[3]For some IBE schemes, the decryption algorithm takes not only (pk, C, sk), but also an ID, as input.

Standard Security Notions of IBE. To define ciphertext indistinguishability against adaptively chosen ID/plaintexts attack (IND-ID-CPA) for an IBE scheme $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$, we need the following game which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$.

$\mathtt{Setup}$. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, and sends pk to $\mathcal{A}$.

$\mathtt{Query}$. $\mathcal{A}$ is allowed to use the key-revelation oracle $\mathtt{Reveal}$ adaptively.

> **$\mathtt{Reveal}(ID)$:** $\mathcal{A}$ queries an ID $ID \in \mathcal{I}$. $\mathcal{CH}$ returns $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$.

$\mathtt{Challenge}(M_0, M_1, ID^*)$. $\mathcal{A}$ sends two plaintexts $M_0, M_1 \in \mathcal{M}$ and an ID $ID^* \in \mathcal{I}$. It is required that ID^* was never queried to the key-revelation oracle. $\mathcal{CH}$ sets $b \overset{\text{U}}{\leftarrow} \{0, 1\}$, then returns $C^* \leftarrow \mathsf{Enc}(pk, M_b)$.

$\mathtt{Query\ 2}$. $\mathcal{A}$ is allowed to use the key-revelation oracle $\mathtt{Reveal}$ adaptively.

> **$\mathtt{Reveal}(ID)$:** $\mathcal{A}$ queries an ID $ID \in \mathcal{I}$ which is not the target ID ID^*. $\mathcal{CH}$ returns $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$.

$\mathtt{Guess}(b')$. $\mathcal{A}$ sends $b' \in \{0, 1\}$ to $\mathcal{CH}$.

Definition 6. *A IBE scheme Σ_{IBE} is said to be IND-ID-CPA secure or adaptively secure if for any PPT adversary $\mathcal{A}$, $\mathsf{Adv}^{\mathtt{IND\text{-}ID\text{-}CPA}}_{\mathcal{A},\Sigma_{IBE}}(\lambda) = |2 \cdot \Pr[b' = b] - 1|$ is negligible.*

Remark. Weaker notion named *ciphertext indistinguishability against selectively chosen ID and adaptively chosen plaintexts attack (IND-CPA)* or selective security is defined basically in the same manner as IND-ID-CPA or the adaptive security except that the adversary is forced to choose the target ID ID^* before receiving the public-key pk.

2.9 Digital Signature

Syntax of Digital Signature. Digital signature consists of the polynomial time algorithms $\{\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver}\}$. Gen and Sig are probabilistic, and Ver is deterministic.

$\mathsf{Gen}(1^\lambda) \to (pk, sk)$. Key-generation algorithm takes 1^λ, where $\lambda \in \mathbb{N}$, as an input, and outputs a public-key pk and a secret-key sk. The message space $\mathcal{M}$ is uniquely determined by pk.

$\mathsf{Sig}(pk, m, sk) \to \sigma$. Signing algorithm takes the public-key pk, a message $m \in \mathcal{M}$, and the secret-key sk as inputs, and outputs a signature σ.

$\mathsf{Ver}(pk, m, \sigma) \to 1 \ / \ 0$. Signature-verification algorithm takes the public-key pk, a message $m \in \mathcal{M}$, and a signature σ as inputs, and outputs 1 or 0.

Any signature scheme must be correct. A signature scheme $\Sigma_{\mathrm{SIG}} = \{\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver}\}$ is correct if for every $(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$, every $m \in \mathcal{M}$, and every $\sigma \leftarrow \mathsf{Sig}(pk, m, sk)$, it holds that $\Pr[1 \leftarrow \mathsf{Ver}(pk, m, \sigma)]$.

Standard Security Notions of Digital Signature. We consider strong existential unfrogeability (in non-leakage setting) for signature schemes. Specifically, we define strong existential unforgeability against adaptively chosen messages attack (sEUF-CMA) for signature schemes.

At first, we define a game w.r.t. a signature scheme $\Sigma_{\mathrm{SIG}} = \{\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver}\}$, which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$ as follows.

`Key-Generation`. $\mathcal{CH}$ runs $(pk, sk) \leftarrow \mathrm{SIG}.\mathsf{Gen}(1^\lambda)$. $\mathcal{CH}$ sends pk to $\mathcal{A}$. $\mathcal{CH}$ initializes the list $\mathcal{L}_S$, i.e., $\mathcal{L}_S := \emptyset$.

`Query`. $\mathcal{A}$ is allowed to use signing oracle **`Sign`**, adaptively.

> **`Sign`**$(m \in \mathcal{M})$**:** $\mathcal{CH}$ generates $\sigma \leftarrow \mathrm{SIG}.\mathsf{Sig}(pk, m, sk)$, then sends σ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \sigma)\}$.

`Forgery`(m^*, σ^*)**.** $\mathcal{CH}$ receives (m^*, σ^*) sent by $\mathcal{A}$. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathrm{SIG}.\mathsf{Ver}(pk, m^*, \sigma^*)] \wedge [(m^*, \sigma^*) \notin \mathcal{L}_S]$. We define his advantage $\mathsf{Adv}^{sEUF-CMA}_{\Sigma_{\mathrm{SIG}},\mathcal{A}}(\lambda)$ as $\Pr[\mathcal{A}\ wins.]$.

Definition 7. Σ_{SIG} *is sEUF-CMA-secure, if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{sEUF-CMA}_{\Sigma_{\mathrm{SIG}},\mathcal{A}}(\lambda)$ *is negligible.*

Remark. Its weaker version, i.e., *weak* existential unforgeability against adaptively chosen messages attack (wEUF-CMA), is defined in the same manner as sEUF-CMA except for the winning condition of the adversary $\mathcal{A}$ in the game. The adversary is said to win the game if the signature σ^* is a valid signature on the message m^*, i.e., $[1 \leftarrow \mathrm{SIG}.\mathsf{Ver}(pk, m^*, \sigma^*)]$, and the message m^* has not been queried to the signing oracle **`Sign`**.

2.10 Identity-Based Signature (IBS)

Syntax of IBS. Identity-based signature (IBS) consists of polynomial time algorithms $\{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$. Ver is deterministic and the others are probabilistic.

$\mathsf{Setup}(1^\lambda, 1^l) \rightarrow (pk, mk)$**.** 1^λ, where $\lambda \in \mathbb{N}$, denotes a security parameter. $l \in \mathbb{N}$ denotes bit-length of an ID. Thus, the ID space is defined as $\mathcal{I} := \{0, 1\}^l$. Setup algorithm takes 1^λ and 1^l as inputs, and outputs a system public-key pk and a master-key mk. The message space $\mathcal{M}$ is uniquely determined by pk.

$\mathsf{KeyGen}(pk, mk, ID) \rightarrow sk$**.** Key-generation algorithm takes pk, mk and an ID $ID \in \mathcal{I}$ as inputs, and outputs a secret-key sk.

$\mathsf{Sig}(pk, m, ID, sk) \rightarrow \sigma$**.** Signing algorithm takes pk, a message $m \in \mathcal{M}$, an ID $ID \in \mathcal{I}$, and a secret-key sk as inputs, and outputs a signature σ.

$\mathsf{Ver}(pk, m, ID, \sigma) \rightarrow 1\ /\ 0$**.** Signature-verification algorithm takes pk, a message $m \in \mathcal{M}$, an ID $ID \in \mathcal{I}$ and a signature σ as inputs, and outputs 1 or 0.

Any IBS scheme must be correct. An IBS scheme $\Sigma_{\text{IBS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ is correct if every $\lambda \in \mathbb{N}$, every $l \in \mathbb{N}$, every $(pk, sk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$, every $ID \in \mathcal{I}$, every $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, every $m \in \mathcal{M}$ and every $\sigma \leftarrow \mathsf{Sig}(pk, m, ID, sk)$, it holds that $\Pr[1 \leftarrow \mathsf{Ver}(pk, m, ID, \sigma)] = 1$.

Standard Security Notions of IBS. We define weak existential unforgeability under adaptively chosen ID/messages attack (in non-leakage setting) (EUF-CMA) for IBS schemes. We consider the following game for an IBS scheme $\Sigma_{\text{IBS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ which is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$. $\mathcal{CH}$ initializes a list $\mathcal{L}_G$ as en empty set $\emptyset$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-generation oracle **Generate**, secret-key-revelation oracle **Reveal**, and signature-generation oracle **Sign** as follows.

Generate$(ID \in \mathcal{I})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$. If a list $\mathcal{L}_{ID}$ for the ID has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such a list $\mathcal{L}_{ID}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Reveal$(ID \in \mathcal{I}, i \in \mathbb{N})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{ID}|]$. $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{ID}$, then returns the secret-key.

Sign$(ID \in \mathcal{I}, i \in \mathbb{N}, m \in \mathcal{M})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$, $m \in \mathcal{M}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{ID}|]$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{ID}$, then generates $\sigma \leftarrow \mathsf{SIG.Sig}(pk, m, ID, sk)$. After that, $\mathcal{CH}$ returns σ and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Forgery$(m^* \in \mathcal{M}, \sigma^*, ID^* \in \mathcal{I})$**.** $\mathcal{A}$ sends a message $m^* \in \mathcal{M}$, a signature σ^* and an ID $ID^* \in \mathcal{I}$. Here, ID^* must be an ID which was never queried to **Reveal**. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{Ver}(pk, m^*, ID^*, \sigma^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]$. Its advantage $\mathsf{Adv}^{EUF-CMA}_{\Sigma_{\text{IBS}},\mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 8. Σ_{IBS} *is EUF-CMA secure (or adaptively secure), if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{EUF-CMA}_{\Sigma_{\text{IBS}},\mathcal{A}}(\lambda)$ *is negligible.*

2.11 Attribute-Based Signature (ABS)

General Circuit [SAH16]. A general circuit is constructed by NAND-gates of fan-in two. Formally, a circuit is determined by integers L, N and functions I_1, I_2. L denotes total number of input wires. N denotes total number of (NAND-)gates. Total number of wires in the circuit is $L + N$. We number each wire. Precisely, we give the input wires the numbers $1, \cdots, L$, the internal wires $L+1, \cdots, L+N-1$, and the output wire $L+N$. $I_1 : [L+1, L+N] \to [1, L+N-1]$ (resp. $I_2 : [L+1, L+N] \to [1, L+N-1]$) takes an internal wire or the output wire $i \in [L+1, L+N]$ as input and outputs left (resp. right) input wire of the gate which generates the wire i.

Syntax of ABS. We give syntax of attribute-based signature (ABS) whose predicate is represented as a general circuit. It consists of polynomial time algorithms $\{\mathsf{Setup}, \mathsf{KeyGen}$ $, \mathsf{Sig}, \mathsf{Ver}\}$. Setup, KeyGen and Sig are probabilistic, and Ver is deterministic.

$\mathsf{Setup}(1^\lambda, 1^L) \rightarrow (pk, mk)$**.** 1^λ, where $\lambda \in \mathbb{N}$, denotes a security parameter. $L \in \mathbb{N}$ denotes the bit-length of an attribute, and the universal set of attributes is denoted by $\mathcal{U} = \{0, 1\}^L$. The setup algorithm takes 1^λ and 1^L as inputs, and outputs a system public-key pk and a master-key mk. The message space $\mathcal{M}$ is uniquely determined by pk.

$\mathsf{KeyGen}(pk, mk, \mathbb{W} \in \mathcal{U}) \rightarrow sk$**.** The key-generation algorithm takes pk, mk and an attribute $\mathbb{W}$ as inputs, and outputs a secret-key sk.

$\mathsf{Sig}(pk, m, \phi = \{L, N, I_1, I_2\}, sk) \rightarrow \sigma$**.** The signing algorithm takes pk, a message $m \in \mathcal{M}$, a predicate ϕ represented as a general circuit $\{L, N, I_1, I_2\}$, and a secret-key sk for an attribute $\mathbb{W}$ such that $\phi(\mathbb{W}) = 1$ as inputs, and outputs a signature σ.

$\mathsf{Ver}(pk, m, \phi = \{L, N, I_1, I_2\}, \sigma) \rightarrow 1 / 0$**.** The signature-verification algorithm takes pk, a message $m \in \mathcal{M}$, a predicate ϕ represented as a general circuit $\{L, N, I_1, I_2\}$, and a signature σ as inputs, and outputs 1 or 0.

Any ABS scheme must be correct. An ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ is correct if for every $\lambda \in \mathbb{N}$, every $L \in \mathbb{N}$, every $(pk, sk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$, every $\mathbb{W} \in \mathcal{U}$, every $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W})$, every $m \in \mathcal{M}$, every ϕ s.t. $\phi(\mathbb{W}) = 1$ and every $\sigma \leftarrow \mathsf{Sig}(pk, m, \phi, sk)$, it holds that $\Pr[1 \leftarrow \mathsf{Ver}(pk, m, \phi, \sigma)] = 1$.

Standard Security Notions of ABS. In this paragraph, we give definitions of two standard security notions of ABS. The first notion is related to unforgeability. The second one is related to signer-privacy.

Firstly, we give definition of a security notion of weak existential unforgeability under adaptively chosen predicate/messages attack (in non-leakage setting) (EUF-CMA) for ABS schemes. We consider the following game for an ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ which is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

`Setup`. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$. The universal set of attributes is set as $\mathcal{U} = \{0, 1\}^L$. A list $\mathcal{L}_S$ is set as a set $\emptyset$.

`Query`. $\mathcal{A}$ is allowed to adaptively use secret-key-generation oracle **`Generate`**, secret-key-revelation oracle **`Reveal`**, and signature-generation oracle **`Sign`** as follows.

- **`Generate`**$(\mathbb{W} \in \mathcal{U})$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W})$. If a list $\mathcal{L}_{\mathbb{W}}$ for the attribute has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such list $\mathcal{L}_{\mathbb{W}}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.
- **`Reveal`**$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N})$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{W}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{\mathbb{W}}$, then returns it.
- **`Sign`**$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$, $m \in \mathcal{M}$, a predicate ϕ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{W}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$, then generates $\sigma \leftarrow \mathrm{SIG}.\mathsf{Sig}(pk, m, \phi, sk)$. After that, $\mathcal{CH}$ returns σ, and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

$\texttt{Forgery}(m^* \in \mathcal{M}, \sigma^*, \phi^*)$. $\mathcal{A}$ sends a message m^*, a signature σ^* and a predicate ϕ^*. The predicate must be one such that every attribute $\mathbb{W}$ queried to **Reveal** satisfies $\phi^*(\mathbb{W}) = 0$. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{Ver}(pk, m^*, \phi^*, \sigma^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$. The advantage $\texttt{Adv}^{EUF-CMA}_{\Sigma_{\mathrm{ABS}}, \mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 9. Σ_{ABS} *is EUF-CMA secure (or adaptively secure), if for every PPT* $\mathcal{A}$*,* $\texttt{Adv}^{EUF-CMA}_{\Sigma_{\mathrm{ABS}}, \mathcal{A}}(\lambda)$ *is negligible.*

Secondly, we give definition of perfect signer-privacy [MPR11, SAH16].

Definition 10. Σ_{ABS} *is perfectly signer-private , if for every* $\lambda, L \in \mathbb{N}$*, every* $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$*, every* $\mathbb{W}_1 \in \mathcal{U}$*, every* $\mathbb{W}_2 \in \mathcal{U}$*, every* $sk_1 \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W}_1)$*, every* $sk_2 \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W}_2)$*, every* $m \in \mathcal{M}$ *and every* ϕ *s.t.* $\phi(\mathbb{W}_1) = \phi(\mathbb{W}_2) = 1$*, distribution of* $\sigma_1 \leftarrow \mathsf{Sig}(pk, m, \phi, sk_1)$ *and distribution of* $\sigma_2 \leftarrow \mathsf{Sig}(pk, m, \phi, sk_2)$ *are identical.*

2.12 Non-Interactive Zero-Knowledge Proof (NIZK)

Syntax of NIZK. Non-interactive zero-knowledge proof (NIZK) Σ_{NIZK} for a language $\mathbb{L}$ consists of three polynomial time algorithms $\{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$. Each one of Gen and Pro is probabilistic. Ver is deterministic. R_L denotes the witness relation.

$\mathsf{Gen}(1^\lambda) \rightarrow crs$. Common Reference String (CRS) generation algorithm takes 1^λ as an input, and outputs a CRS crs.

$\mathsf{Pro}(crs, x, w) \rightarrow \pi$. The proof-generation algorithm takes the CRS crs, a statement x, and a witness w as inputs, and outputs a proof π.

$\mathsf{Ver}(crs, x, \pi) \rightarrow 1\ /\ 0$. The proof-verification algorithm takes the CRS crs, a statement x, and a proof π as inputs, and outputs 1 or 0.

Any NIZK scheme must be correct. An NIZK scheme $\Sigma_{\mathrm{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ is correct if for every $\lambda \in \mathbb{N}$, every $crs \leftarrow \mathsf{Gen}(1^\lambda)$, every (x, w) such that $(x, w) \in R_{\mathbb{L}}$, and every $\pi \leftarrow \mathsf{Pro}(crs, x, w)$, it holds that $\Pr[1 \leftarrow \mathsf{Ver}(crs, x, \pi)] = 1$.

Standard Security Notions of NIZK. We give the definitions of soundness and zero-knowledge for an NIZK scheme.

Definition 11. *An NIZK scheme* $\Sigma_{\mathrm{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ *is sound if for every* $\lambda \in \mathbb{N}$*, every* $crs \leftarrow \mathsf{Gen}(1^\lambda)$*, and every PPT* $\mathcal{A}$*,*

$$\Pr\left[\mathcal{A}(crs) \rightarrow (x, \pi)\ s.t.\ [\mathsf{Ver}(crs, x, \pi) \rightarrow 1] \wedge [x \notin \mathbb{L}]\right]$$

is negligible.

Definition 12. *An NIZK scheme* $\Sigma_{\text{NIZK}} = \{\mathsf{Gen}, \mathsf{Pro}, \mathsf{Ver}\}$ *is zero-knowledge if for every* $\lambda \in \mathbb{N}$ *and every PPT* $\mathcal{A}$*, there exists a PPT* $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ *such that*

$$\Big| \Pr\Big[\mathcal{A}^{O_0^{crs}(x,w)}(crs) \to 1 \mid \mathsf{Gen}(1^\lambda) \to crs\Big] - \\ \Pr\Big[\mathcal{A}^{O_1^{crs,td}(x,w)}(crs) \to 1 \mid \mathcal{S}_1(1^\lambda) \to (crs, td)\Big]\Big|$$

is negligible, where $O_0^{crs}(x, w)$ *returns* $\mathsf{Pro}(crs, x, w)$ *(resp.* $\perp$*), if* $(x, w) \in R_{\mathbb{L}}$ *(resp.* $(x, w) \notin R_{\mathbb{L}}$*), and* $O_1^{crs,td}(x, w)$ *returns* $\mathcal{S}_2(crs, x, td)$ *(resp.* $\perp$*), if* $(x, w) \in R_{\mathbb{L}}$ *(resp.* $(x, w) \notin R_{\mathbb{L}}$*).*

Chapter 3

IBE with Hard-to-Invert Leakage-Resilinece

3.1 Introduction for Chapter 3

3.1.1 Background

Identity-based encryption (IBE) is one type of public-key encryption (PKE), and has an advantage such that we can use an identifier $ID \in \{0, 1\}^*$ (e.g., a mail address, telephone number) as the public key. The idea of IBE was presented by Shamir [Sha84] at Crypto'84. The first concrete IBE scheme which utilizes a bilinear map was proposed by Boneh et al. [BF01]. Although the IBE scheme by Boneh et al. was proven to be secure in the random oracle model, large number of IBE schemes have been proven to be secure in the standard model, e.g., [CHK03, BB04a, BB04b, Wat05, Wat09]. Large number of lattice-based IBE schemes have been proposed, e.g., [GPV08, ABB10].

3.1.2 Related Work

Several schemes secure in auxiliary leakage (AL) model (or hard-to-invert leakage (HL) model) have been proposed. Symmetric-key encryption scheme [DKL09] and digital signature scheme [FHN+12, YYH12, WMHT16] have been proposed. PKE scheme has been proposed by Dodis et al. [DGK+10]. An IBE scheme has been proposed by Yuen et al. [YCZY12] at Eurocrypt'12. They [YCZY12] proposed not only the IBE scheme secure in AL model, but also an IBE scheme secure in continual auxiliary leakage (CAL) model which combines AL model and CL model. ABE schemes (secure in AL model) have been proposed by Zhang et al. [ZWTM13] and Wang et al. [WY15].

3.1.3 Our Results

Firstly, we present an IBE construction and prove that it is fully secure, i.e., adaptively secure, and resilient to auxiliary leakage under the decisional linear (DLIN) assumption [BBS04] in the standard model. Secondly, we show that the proof of security of the IBE scheme by Yuen et al. [YCZY12] proposed in Eurocrypt'12 is wrong. Yuen et al. insist that the IBE scheme is correctly proven to be fully secure and resilient to auxiliary leakage under the decisional subgroup (DSG) assumptions [LW10] in the standard model. As far as we know, among the previous works, the IBE scheme has been the only one which is proven to be resilient to auxiliary leakage. Therefore, our IBE construction is the only one proven to be fully secure and resilient to auxiliary leakage under standard assumptions. Below, we explain each result in more detail.

Result 1: Presenting an IBE construction fully secure and resilient to auxiliary leakage under the DLIN assumption. Our security model for IBE scheme is a model adding a leakage oracle leaking information from a secret-key for the target ID to the standard full security model without leakage-resilience such as one in [Wat05]. The leakage oracle is added between the two phases "Phase 1" and "Challenge". The leakage oracle takes as inputs an ID ID^* as the target ID and a function f, then generates a secret-key sk^* for ID^* and returns $f(sk^*)$. The leakage function f is a function included in a function class $\mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda))$ parameterized by variables such as the system public-key pk and master-key mk generated in the security game, the target ID ID^*, and a negligible function $\xi(\lambda)$. And, the definition of $\mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda))$ is as follows. It consists of every function f s.t. for every PPT $\mathcal{B}$, it holds that $\Pr[\mathcal{B}(pk, mk, \mathcal{L}, ID^*, OUT, f, f(sk^*)) \rightarrow sk^* | (sk^*, OUT) \leftarrow \mathsf{KeyGen}'_{IN}(pk, mk, ID^*)] < \xi(\lambda)$, where the algorithm $\mathsf{KeyGen}'_{IN}(pk, mk, ID^*)$ is a key-generation algorithm such that the secret-key sk^* is generated in the same manner as the "normal" key-generation algorithm KeyGen and the variable OUT includes some information about the secret-key sk^*.

Our IBE scheme is a modified variant of one presented by Kurosawa and Phong [KP13] which has been proven to fully secure and resilient to bounded leakage under the DLIN assumption. We introduce an assumption "assumption X parameterized by (an integer) l " which is implied by the DLIN assumption. We prove that our IBE construction is secure in our security model (mentioned earlier) with auxiliary leakage parameter $\xi(\lambda) = 2^{-m^\epsilon}$ under the assumption X parameterized by m, where the integer m is set to $m := (4\lambda)^{1/\epsilon}$ by using a constant $0 < \epsilon < 1$. Actually, m is related to the number of elements included in a secret-key for each ID. Specifically, secret-key space is $\mathbb{Z}_p^{2m}$, where p is a prime. Although both the IBE constructions presented in our work and the work [KP13] are proven to be secure under the DLIN assumption, reduction cost to the assumption achieved by each work is different. Ours is determined by the parameter m only. On the other hand, one in [KP13] is determined by the bit-length of an ID and the total number of oracle queries made by an adversary in security game. The difference can be interesting.

Result 2: Showing that security proof for the IBE construction proposed by Yuen et al. [YCZY12] is wrong. Although Yuen et al. describes a proof sketch in [YCZY12], they have not disclosed the full version of the proof. Based on the proof sketch, we show that their proof is wrong. Specifically, we introduce several concrete examples of polynomial-time adversary as counterexamples, each one of which becomes an evidence that their proof is wrong[1]. The proof by Yuen et al. is done by a hybrid argument using a sequence of

[1]Note that each one of our counterexamples indicates that their current proof is wrong, but not that their scheme cannot be proven to be secure in their security model. Thus, it is possible that their scheme is proven

games. They insist that a lemma guarantees that two successive games are indistinguishable since it can be reduced to an indistinguishability-type assumption, i.e., a DSG assumption [LW10]. However, each one of our counterexamples distinguishes the two games with a non-negligible advantage. Thus, the lemma at least does not hold.

Yuen et al. also presents an IBE scheme resilient to continual auxiliary leakage. We can show that properly modified variants of the counterexamples for the scheme resilient to auxiliary leakage work effectively against it as well[3].

Although the ABE schemes by Zhang et al. [ZWTM13] and Wang et al. [WY15] have been considered to be the only ABE schemes proven to be resilient to auxiliary leakage, we show that their security proofs are defective. For details, see Sect. 3.6.

3.1.4 Organization

This chapter is organized as follows. In Sect. 3.2, definition of indistinguishability in HL model of IBE schemes is given. In Subsect. 3.3.1, concrete construction of our IBE scheme is given. We prove its security and consider its leakage-resilience in Subsect. 3.3.2 and Subsect. 3.3.3, respectively. In Sect. 3.4, the results by Yuen et al. [YCZY12] are described. Especially, definition of indistinguishability in HL model of IBE schemes used in their paper is given in Subsect. 3.4.3, and their concrete construction of IBE scheme is given in Subsect. 3.4.4. In Sect. 3.5, we introduce some counterexamples against the security proof by Yuen et al. In Sect. 3.6, we discuss the deficiency included in security proofs for ABE schemes proposed by Zhang et al. [ZWTM13] and Wang et al. [WY15]. Sect. 3.7 is the conclusion for this chapter.

3.2 Definition of Indistinguishability in HL Model of IBE

We define ciphertext indistinguishability in the auxiliary leakage (AL) model for an IBE scheme $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$. Before its concrete definition, we define a probabilistic polynomial-time algorithm KeyGen'_{IN} as follows.

- $\mathsf{KeyGen}'_{IN}(pk, mk, ID) \to (sk, OUT)$: This algorithm takes the system public key pk, the master key mk, $ID \in \mathcal{I}$, and an input-information $IN \in \{0,1\}^*$ as inputs, and outputs a user's secret key sk for ID, and an output-information $OUT \in \{0,1\}^*$ which actually is an information about the secret-key sk.

The algorithm is used in the definition of the ciphertext indistinguishability of an IBE scheme. The secret-key generated by the algorithm must be generated in the same manner as the secret-key generated by the normal secret-key generation algorithm KeyGen.

Let us define the ciphertext indistinguishability for IBE. Firstly, we define a security game $\xi(\lambda)$-AL-IND-ID-CPA for an IBE scheme $\Pi_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ and its algorithm KeyGen'_{IN}, which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$. Here, AL-IND-ID-CPA indicates ciphertext indistinguishability against adaptively chosen ID and plaintexts attacks in the auxiliary leakage model. The parameter $\xi(\cdot)$ whom we call auxiliary

to be secure if the proof is done in another manner.

leakage parameter, is a negligible function s.t. $\xi(\lambda) > 2^{-k}$, where k denotes the minimum entropy of the user's secret key. Concrete definition of the game is as follows.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$. $\mathcal{CH}$ sends pk to $\mathcal{A}$. $\mathcal{CH}$ initializes the list $\mathcal{L}$ by an empty set $\emptyset$.

Query. $\mathcal{A}$ is allowed to use the key-revelation oracle **Reveal** adaptively.

> **Reveal**(ID)**:** $\mathcal{A}$ issues an ID as a query. $\mathcal{CH}$ runs $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, then returns sk to $\mathcal{A}$. Then, $\mathcal{CH}$ sets $\mathcal{L} := \mathcal{L} \cup \{(ID, sk)\}$.

Leak$(ID^*, f \in \mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda)))$**.** $\mathcal{A}$ sends an ID ID^* which was not queried to **Reveal** in **Query** and a function $f \in \mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda))$. Definition of the function class $\mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda))$ is given below the game. $\mathcal{CH}$ computes a secret user-key for the ID ID^* by running $sk^* \leftarrow \mathsf{KeyGen}(pk, mk, ID^*)$. Then, $\mathcal{CH}$ computes $f(sk^*; r)$, where r is a randomness of the function.

Challenge(M_0, M_1)**.** $\mathcal{A}$ sends two distinct plaintexts $M_0, M_1 \in \mathcal{M}$. $\mathcal{CH}$ sets $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$, runs $C^* \leftarrow \mathsf{Enc}(pk, M_b, ID^*)$, and sends C^* to $\mathcal{A}$.

Query 2. $\mathcal{A}$ uses the oracle **Reveal** adaptively in the same manner as **Query** except that he cannot query ID^* to the oracle.

Guess$(b' \in \{0, 1\})$**.** $\mathcal{A}$ sends a guess $b' \in \{0, 1\}$ for b.

Advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{\xi(\lambda)-AL-IND-ID-CPA}_{\Pi_{IBE},\mathsf{KeyGen}'_{IN},\mathcal{A}}(\lambda) := |\Pr[b = b'] - 1/2|$. The function class $\mathcal{F}_{pk,mk,\mathcal{L},ID^*}(\xi(\lambda))$ consists of every probabilistic or deterministic polynomial-time function $f : \{0, 1\}^{|sk|} \to \{0, 1\}^*$ such that for every PPT $\mathcal{B}$, it holds that $\Pr[\mathcal{B}(pk, mk, \mathcal{L}, ID^*, OUT, f, f(sk^*)) \to sk^* | (sk^*, OUT) \leftarrow \mathsf{KeyGen}'_{IN}(pk, mk, ID^*)] < \xi(\lambda)$.

Definition 13. *A scheme* $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *is* $(\xi(\lambda))$*-AL-IND-ID-CPA w.r.t.* KeyGen'_{IN}*, if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{\xi(\lambda)-AL-IND-ID-CPA}_{\Pi_{IBE},\mathsf{KeyGen}'_{IN},\mathcal{A}}(\lambda)$ *is negligible.*

Differences from the Security Model in [YCZY12]. Our security model is different from one in [YCZY12]. For the definition of the latter model, see [YCZY12] or Subsect. 3.4.3 of this paper.

Actually, our security model is weaker than one in [YCZY12]. For instance, the following three properties make our security model weaker. Firstly, our security model lacks leakage-resilience for the master-key. Secondly, ours does not allow the adversary to use adaptively in the phase 1 an oracle which reveals secret-keys for IDs other than the target ID ID^* and an oracle which leaks some information from secret-key(s) for ID^*. Thirdly, ours does not allow the adversary to use the leakage oracle multiple times.

On the other hand, our security model is superior to one in [YCZY12] in some respects. One of them is related to the definition of the leakage function class.

In [YCZY12], the authors define the leakage function class in a way that any PPT cannot find any secret-key for the target ID ID^* with a probability larger than a concrete negligible function. It is hard for us to present concrete examples of functions which satisfy the definition. It is uncertain whether even a function which outputs only 1 bit of a secret-key for ID^* really satisfies the definition. Actually, in [YCZY12], any specific example which satisfies the definition is not given.

We define the leakage function class in a way that any PPT which is given a leakage-information from a single secret-key sk^* for ID^* cannot identify sk^* with a probability larger than a concrete negligible function. A benefit whom we obtain by using such a definition is that it is possible for us to present some concrete examples of functions which satisfy the definition. Note that in the definition, the inverter $\mathcal{B}$ is given a variable OUT which includes some information about the target secret-key sk^*. It is obvious that if $OUT = sk^*$, no function satisfies the definition. So, we should make the information about sk^* which is revealed by OUT as small as possible. In Subsect. 3.3.3, we consider what kind of leakage functions satisfy the definition. We show that not only a function which reveals 1 bit of sk^*, but also various concrete functions, can satisfy the definition.

3.3 Proposed IBE Scheme

In Subsect. 3.3.1, concrete construction of our IBE scheme Π_{IBE} is given. In Subsect. 3.3.2, its security, i.e., ciphertext indistinguishability, is proven. In Subsect. 3.3.3, we consider the leakage allowed for our IBE scheme.

3.3.1 Concrete Construction

The scheme $\Pi_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ is defined as follows.

$\mathsf{Setup}(1^\lambda, 1^n)$**:** Run $(p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g) \leftarrow \mathcal{G}_{pg}(1^\lambda)$, where $\mathcal{G}_{pg}$ denotes a generator of symmetric bilinear pairing with groups of prime order. Each one of ID space and plaintext space is set as $\mathcal{I} := \{0,1\}^n$ and $\mathcal{M} := \mathbb{G}_T$, respectively. For $\epsilon \in \mathbb{R}$ such that $0 < \epsilon < 1$, set $m := (4\lambda)^{1/\epsilon}$. $\mathbf{A}_0$ and $\mathbf{E}$ are set as $\mathbf{A}_0 \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p^{2\times m}$ and $\mathbf{E} \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p^{m\times 1}$, respectively.

For $i \in [0, n]$, set $r_i \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p$. Set $\mathbf{A}'_0 \in \mathbb{Z}_p^{2\times m}$, $\mathbf{A}_i \in \mathbb{Z}_p^{2\times m}$, where $i \in [1, n]$, and $\mathbf{D} \in \mathbb{Z}_p^{2\times 1}$ as $\mathbf{A}'_0 := r_0\mathbf{A}_0$, $\mathbf{A}_i := r_i\mathbf{A}_0$, and $\mathbf{D} := \mathbf{A}_0\mathbf{E}$, respectively.

`Return` (pk, mk), where $pk := (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^{\mathbf{A}_0}, g^{\mathbf{A}'_0}, g^{\mathbf{A}_1}, \cdots, g^{\mathbf{A}_n}, g^{\mathbf{D}})$ and $mk := (r_0, r_1, \cdots, r_n, \mathbf{E})$.

Hereafter, for an $ID \in \mathcal{I}$, $ID[i] \in \{0,1\}$ denotes the i-th bit of ID. An $ID \in \mathcal{I}$ is associated to a matrix $\mathbf{F}(ID) \in \mathbb{Z}_p^{2\times 2m}$ which is defined as $\mathbf{F}(ID) = \left[\mathbf{A}_0 | \mathbf{A}'_0 + \Sigma_{i=1}^n ID[i]\mathbf{A}_i\right] = \mathbf{A}_0 \left[\mathbf{I}_m | (r_0 + \Sigma_{i=1}^n ID[i] r_i)\mathbf{I}_m\right]$.

$\mathsf{KeyGen}(pk, mk, ID)$**:** For $i \in [m+1, 2m]$, set $v_i \overset{\text{U}}{\longleftarrow} \mathbb{Z}_p$. For $i \in [1, m]$, set $v_i := E_i - (r_0 + \sum_{k=1}^n ID[k] r_k) v_{m+i}$, where E_i is the $(i, 1)$-th element of the vector $\mathbf{E} \in \mathbb{Z}_p^{m\times 1}$. Set $sk := g^{\mathbf{v}}$, where the $(i, 1)$-th element of $\mathbf{v} \in \mathbb{Z}_p^{2m\times 1}$ is set as v_i [2]. `Return` sk.

[2]It obviously holds that $\left[\mathbf{I}_m | (r_0 + \Sigma_{i=1}^n ID[i] r_i)\mathbf{I}_m\right] \mathbf{v} = \mathbf{E}$. Note that by pre-multiplying this equation by $\mathbf{A}_0$, we obtain $\mathbf{F}(ID)\mathbf{v} = \mathbf{D}$.

Enc(pk, M, ID): Set $\mathbf{z} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times 2}$. Calculate $C_1 := g^{\mathbf{zF}(ID)} \in \mathbb{G}^{1\times 2m}$ and $C_2 := \hat{e}(g,g)^{\mathbf{zD}} \cdot M \in \mathbb{G}_T$. Return $C := (C_1, C_2)$.

Dec(pk, C, sk): C is parsed as (C_1, C_2). By using $\mathbf{c} \in \mathbb{Z}_p^{1\times 2m}$, C_1 is written as $g^{\mathbf{c}} \in \mathbb{G}^{1\times 2m}$. By using $\mathbf{v} \in \mathbb{Z}_p^{1\times 2m}$, sk is written as $g^{\mathbf{v}} \in \mathbb{G}^{2m\times 1}$. Calculate $K := \hat{e}(g,g)^{\mathbf{cv}} \in \mathbb{G}_T$. Return C_2/K.

The IBE scheme is correct. We consider a ciphertext for an ID ID and a plaintext M generated properly, i.e., $C = (C_1, C_2) = (g^{\mathbf{zF}(ID)}, \hat{e}(g,g)^{\mathbf{zD}} \cdot M)$, and a secret user-key for the same ID ID generated properly, i.e., $sk = g^{\mathbf{v}}$. When C is decrypted by using sk, the calculated value of K becomes $K = \hat{e}(g,g)^{\mathbf{zF}(ID)\mathbf{v}} = \hat{e}(g,g)^{\mathbf{zD}}$ since $\mathbf{F}(ID)\mathbf{v} = \mathbf{D}$. Hence, M is correctly obtained.

The algorithm KeyGen$'_l$ is defined as follows, where l is a polynomial function.

KeyGen$'_l(pk, mk, ID)$: For $i \in [m+1, 2m]$, set $v_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. For $i \in [1, m]$, set $v_i := E_i - (r_0 + \sum_{k=1}^{n} ID[k]r_k)v_{m+i}$, where E_i is the $(i,1)$-th element of the vector $\mathbf{E} \in \mathbb{Z}_p^{m\times 1}$. For every $i \in [1, m]$, do the following:

- Set $H_i := \{v_i\}$ and $H_{m+i} := \{v_{m+i}\}$. For every $j \in [2, l]$, do the following:
  - Set $v_{m+i,j} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p \setminus H_{m+i}$ and $v_{i,j} := E_i - (r_0 + \sum_{k=1}^{n} ID[k]r_k)v_{m+i,j}$. Set $H_i := H_i \cup \{v_{i,j}\}$ and $H_{m+i} := H_{m+i} \cup \{v_{m+i,j}\}$.

Set $H^* := H_1 \cup \cdots \cup H_{2m}$. Set $sk := g^{\mathbf{v}}$, where the $(i,1)$-th element of $\mathbf{v} \in \mathbb{Z}_p^{2m\times 1}$ is set as v_i. Return (sk, H^*).

Obviously, for any $ID \in \mathcal{I}$, the secret user-key outputted by KeyGen$'_l$ is generated in the same manner as the secret user-key outputted by KeyGen.

3.3.2 Proof of Indistinguishability in HL Model

In this subsection, security, i.e., ciphertext indistinguishability in AL model, of the IBE scheme Π_{IBE} is proven. Before entering the details of the proof, we introduce a new computational assumption. The assumption X parameterized by an integer l is an assumption which insists that for every PPT $\mathcal{A}$ and $\lambda \in \mathbb{N}$,

$$\Bigg|\Pr\Bigg[\mathcal{A}(p, \mathbb{G}, g, g^{\mathbf{A}_b}) \to b \;\Big|\; (p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda), \mathbf{x}, \mathbf{y}, \mathbf{z} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times l}, s, t, \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p,$$
$$\mathbf{A}_0 := \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{z} \end{bmatrix} \in \mathbb{Z}_p^{3\times l}, \mathbf{A}_1 := \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \\ s\mathbf{x} + t\mathbf{y} \end{bmatrix} \in \mathbb{Z}_p^{3\times l}, b \overset{\mathrm{U}}{\leftarrow} \{0,1\}\Bigg] - 1/2\Bigg|$$

is negligible. Validity of the assumption is guaranteed by the following theorem whose proof is given below.

Theorem 3.3.1. *For any $l \in \mathbb{N}$, the assumption X parameterized by l is implied by the DLIN assumption.*

Proof of Theorem 3.3.1. For a PPT $\mathcal{A}$, an integer $\lambda \in \mathbb{N}$, and integers $0 \le i, j \le l$, we define

$$\begin{aligned}\mathtt{Adv}^{i,j}_{\mathcal{A}}(\lambda) := \Big|\Pr\Big[&\mathcal{A}\big(p, \mathbb{G}, g, g^{\mathbf{x}}, g^{\mathbf{y}}, g^{(sx_1+ty_1\ \cdots\ sx_i+ty_i\ z_{i+1}\ \cdots\ z_l)}\big) \to 1 \\ &\Big|(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda), \mathbf{x}, \mathbf{y}, \mathbf{z} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times l}, s, t \overset{\text{U}}{\leftarrow} \mathbb{Z}_p\Big] \\ - \Pr\Big[&\mathcal{A}\big(p, \mathbb{G}, g, g^{\mathbf{x}}, g^{\mathbf{y}}, g^{(sx_1+ty_1\ \cdots\ sx_j+ty_j\ z_{j+1}\ \cdots\ z_l)}\big) \to 1 \\ &\Big|(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda), \mathbf{x}, \mathbf{y}, \mathbf{z} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times l}, s, t \overset{\text{U}}{\leftarrow} \mathbb{Z}_p\Big]\Big|, \end{aligned} \tag{3.1}$$

where for every integer $k \in [1, l]$, the $(1, k)$-th element of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are denoted by x_k, y_k and z_k, respectively. Theorem 3.3.1 is proven by the following two lemmas whose proofs are given below. □

Lemma 3.3.1. *For any integer $4 \le i \le l$ and any PPT $\mathcal{A}$, $\mathtt{Adv}^{i,i-1}_{\mathcal{A}}(\lambda)$ is negligible under the DLIN assumption.*

Lemma 3.3.2. *For any PPT $\mathcal{A}$, $\mathtt{Adv}^{3,0}_{\mathcal{A}}(\lambda)$ is negligible under the DLIN assumption.*

Proof of Lemma 3.3.1. We prove the lemma by contradiction. We prove that if we assume that there is a PPT $\mathcal{A}$ which makes $\mathtt{Adv}^{i,i-1}_{\mathcal{A}}(\lambda)$ non-negligible, we can construct a PPT $\mathcal{S}$ which breaks the DLIN assumption, cf. Subsect. 2.3.2.

$\mathcal{S}$ behaves as follows. $\mathcal{S}$ receives $(p, \mathbb{G}, g, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{s_b})$ as an instance of the DLIN problem. $\mathcal{S}$ sets $\mathbf{x}, \mathbf{y}, \mathbf{z} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times l}$. The vectors $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are parsed as $(x_1\ \cdots\ x_l)$, $(y_1\ \cdots\ y_l)$ and $(z_1\ \cdots\ z_l)$, respectively. $\mathcal{S}$ sets a matrix $g^{\mathbf{A}}$ to

$$\begin{pmatrix} g_1^{x_1} & \cdots & g_1^{x_{i-1}} & g_3 \cdot g_1^{x_i} & g_1^{x_{i+1}} & \cdots & g_1^{x_l} \\ g_2^{y_1} & \cdots & g_2^{y_{i-1}} & g_3 \cdot g_2^{y_i} & g_2^{y_{i+1}} & \cdots & g_2^{y_l} \\ g_1^{r_1 x_1} \cdot g_2^{r_2 y_1} & \cdots & g_1^{r_1 x_{i-1}} \cdot g_2^{r_2 y_{i-1}} & g_3^{s_b} \cdot g_1^{r_1 x_i} \cdot g_2^{r_2 y_i} & g_3^{z_{i+1}} & \cdots & g_3^{z_l} \end{pmatrix}. \tag{3.2}$$

For $i \in \{1, 2, 3\}$, let $g^{\mathbf{a}_i}$ denote the i-th row vector of $g^{\mathbf{A}}$. $\mathcal{S}$ gives $(p, \mathbb{G}, g, g^{\mathbf{a}_1}, g^{\mathbf{a}_2}, g^{\mathbf{a}_3})$ to $\mathcal{A}$, then outputs the output $b' \in \{0, 1\}$ by $\mathcal{A}$.

If $s_b := r_1 + r_2$ (resp. $s_b \overset{\text{U}}{\leftarrow} \mathbb{Z}_p$), $\mathcal{A}$ is given an input which distributes identically to the input whom the former (resp. latter) PPT $\mathcal{A}$ in the $\mathtt{Adv}^{i,i-1}_{\mathcal{A}}$ defined by (3.1) is given. Hence, $\mathtt{Adv}^{DLIN}_{\mathcal{S}}(\lambda) = \mathtt{Adv}^{i,i-1}_{\mathcal{A}}(\lambda)$. □

Proof of Lemma 3.3.2. Consider a PPT $\mathcal{S}$ which attempts to break the DLIN assumption. $\mathcal{S}$ receives $(p, \mathbb{G}, g_1, g_2, g_3, g_1^{r_1}, g_2^{r_2}, g_3^{s_b})$ as an instance of the DLIN problem. $\mathcal{S}$ sets $\mathbf{x}, \mathbf{y} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times l}$. $\mathcal{S}$ sets a matrix $g^{\mathbf{A}}$ to

$$\begin{pmatrix} g_1^{x_1} & g_1^{x_2} & g_3 \cdot g_1^{x_3} \\ g_2^{y_1} & g_2^{y_2} & g_3 \cdot g_2^{y_3} \\ g_1^{r_1 x_1} \cdot g_2^{r_2 y_1} & g_1^{r_1 x_2} \cdot g_2^{r_2 y_2} & g_3^{s_b} \cdot g_1^{r_1 x_3} \cdot g_2^{r_2 y_3} \end{pmatrix}. \tag{3.3}$$

$\mathcal{S}$ gives $(p, \mathbb{G}, g, g^{\mathbf{a}_1}, g^{\mathbf{a}_2}, g^{\mathbf{a}_3})$ to $\mathcal{A}$, then outputs the output $b' \in \{0, 1\}$ by $\mathcal{A}$.

If $s_b := r_1 + r_2$ (resp. $s_b \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$), $\mathcal{A}$ is given an input which distributes identically to the input whom the former (resp. latter) PPT $\mathcal{A}$ in the $\mathtt{Adv}^{3,0}_{\mathcal{A}}$ defined by (3.1) is given. Hence, $\mathtt{Adv}^{DLIN}_{\mathcal{A}}(\lambda) = \mathtt{Adv}^{3,0}_{\mathcal{S}}(\lambda)$. □

The indistinguishability in HL model of the IBE scheme Π_{IBE} is guaranteed by the following theorem.

Theorem 3.3.2. *IBE scheme* Π_{IBE} *is* $(2^{-m^{\epsilon}})$*-AL-IND-ID-CPA-secure with respect to* KeyGen'_l *under the assumption X parameterized by m.*

Proof of Theorem 3.3.2. $\mathtt{Game}_i$, where $i \in \{0, 1, 2\}$, is defined as follows.

$\mathtt{Game}_0$: This is a normal $(2^{-m^{\epsilon}})$-AL-IND-ID-CPA game for the IBE scheme Π_{IBE} and KeyGen'_l which is played between a PPT adversary $\mathcal{A}$ and challenger $\mathcal{CH}$.

$\mathtt{Game}_1$: This game is the same as $\mathtt{Game}_0$ except that the challenge ciphertext $C^* = (C_1^*, C_2^*)$ is generated as follows: $C_1^* := g^{[\mathbf{y}^*|r^*\mathbf{y}^*]} \in \mathbb{G}^{1\times 2m}$ and $C_2^* := \hat{e}(g, g)^{[\mathbf{y}^*|r^*\mathbf{y}^*]\mathbf{v}^*} \cdot M_b \in \mathbb{G}_T$, where $\mathbf{y}^* \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$, $r^* := r_0 + \Sigma_{i=1}^{n} ID^*[i] r_i$, and $\mathbf{v}^* \in \mathbb{Z}_p^{2m\times 1}$ is the exponent of the secret-key $sk^* = g^{\mathbf{v}^*}$ for ID^*.

$\mathtt{Game}_2$: This game is the same as $\mathtt{Game}_1$ except that C_2^* of the challenge ciphertext C^* is set to $C_2^* := \hat{e}(g, g)^u \cdot M_b$, where $u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$.

For $i \in \{0, 1, 2\}$, W_i denotes the event where $\mathcal{A}$ outputs b' such that $b' = b$ in $\mathtt{Game}_i$. The advantage of an adversary $\mathcal{A}$ playing the game of $\left(2^{-m^{\epsilon}}\right)$-AL-IND-ID-CPA for the IBE scheme Π_{IBE} and KeyGen'_l satisfies $\mathtt{Adv}^{(2^{-m^{\epsilon}})-AL-IND-ID-CPA}_{\Pi_{IBE},\mathsf{KeyGen}'_l,\mathcal{A}}(\lambda) = |\Pr[W_0] - 1/2| \leq |\Pr[W_0] - \Pr[W_1]| + |\Pr[W_1] - \Pr[W_2]| + |\Pr[W_2] - 1/2|$.

By the above inequality and the following three lemmas, Theorem 3.3.2 is proven. □

Lemma 3.3.3. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible under the assumption X parameterized by m.*

Lemma 3.3.4. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible.*

Lemma 3.3.5. $|\Pr[W_2] - 1/2|$ *is negligible.*

We prove the first two lemmas. Since Lemma 3.3.5 is true obviously, its proof is not needed.

Proof of Lemma 3.3.3. We prove the lemma by contradiction. Specifically, we prove that if there exists a PPT $\mathcal{A}$ which makes $|\Pr[W_0] - \Pr[W_1]|$ non-negligible, then a PPT simulator $\mathcal{S}$ which breaks the assumption X parameterized by m can be constructed.

The simulator $\mathcal{S}$ is given a matrix $g^{\bar{\mathbf{A}}} \in \mathbb{G}^{3\times m}$ as an instance of the problem X, then simulates $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) against $\mathcal{A}$ if $\bar{\mathbf{A}}$ is the matrix $\mathbf{A}_1$ (resp. $\mathbf{A}_0$) whose third row vector is a linear combination of the first two row vectors (resp. a randomly generated one). The concrete behavior by $\mathcal{S}$ is as follows.

Setup: $(p, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$ is run. g denotes a generator of $\mathbb{G}$. $\mathcal{S}$ is given $(g, g^{\bar{\mathbf{A}}})$ as an instance of the problem in the assumption X, where $\bar{\mathbf{A}} \in \mathbb{Z}_p^{3\times m}$. $\mathcal{S}$ sets the first (resp. second) row vector of $g^{\mathbf{A}_0}$ as the first (resp. second) row vector of $g^{\bar{\mathbf{A}}}$.

$\mathcal{S}$ sets $r_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ where $i \in [1, n]$, and $\mathbf{E} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{m\times 1}$. $\mathcal{S}$ computes $g^{\mathbf{A}'_0}, g^{\mathbf{A}_i}$, where $i \in [1, n]$, and $g^{\mathbf{D}}$ by using $g^{\mathbf{A}_0}, r_i$, where $i \in [1, n]$, and $\mathbf{E}$, where $\mathbf{A}'_0 = r_0\mathbf{A}_0$, $\mathbf{A}_i = r_i\mathbf{A}_0$, and $\mathbf{D} = \mathbf{A}_0\mathbf{E}$.

$\mathbf{F}(ID)$ is defined as $\mathbf{F}(ID) = \left[\mathbf{A}_0|\mathbf{A}'_0 + \Sigma_{i=1}^n ID[i]\mathbf{A}_i\right] = \mathbf{A}_0\left[\mathbf{I}_m|(r_0 + \Sigma_{i=1}^n ID[i]r_i)\mathbf{I}_m\right]$.

$\mathcal{S}$ sets $pk := (q, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^{\mathbf{A}_0}, g^{\mathbf{A}'_0}, g^{\mathbf{A}_1}, \cdots, g^{\mathbf{A}_n}, g^{\mathbf{D}})$ and $mk := (r_0, r_1, \cdots, r_n, \mathbf{E})$.

$\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes the list $\mathcal{L}$ as an empty set $\emptyset$.

Query: When $\mathcal{A}$ issues a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows:

Reveal$(ID \neq ID^*)$**:** $\mathcal{S}$ randomly chooses $\mathbf{v} \in \mathbb{Z}_p^{2m\times 1}$ satisfying $\left[\mathbf{I}_m|(r_0 + \Sigma_{i=1}^n ID[i]r_i)\mathbf{I}_m\right]\mathbf{v} = \mathbf{E}$, and sets $sk := g^{\mathbf{v}}$. $\mathcal{S}$ sets $\mathcal{L} := \mathcal{L} \cup \{(ID, sk)\}$.

Leak(ID^*, f)**:** $\mathcal{S}$ randomly chooses $\mathbf{v}^* \in \mathbb{Z}_p^{2m\times 1}$ satisfying $\left[\mathbf{I}_m|(r_0 + \Sigma_{i=1}^n ID^*[i]r_i)\mathbf{I}_m\right]\mathbf{v}^* = \mathbf{E}$, then sets $sk := g^{\mathbf{v}^*}$. $\mathcal{S}$ computes $f(sk^*)$, then gives it to $\mathcal{A}$.

Challenge(M_0, M_1)**:** $\mathcal{S}$ receives two plaintexts (M_0, M_1). $b \in \{0, 1\}$ is set as $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$. $r^* \in \mathbb{Z}_p$ is set as $r^* := r_0 + \Sigma_{i=1}^n ID^*[i]r_i$. $g^{\mathbf{y}} \in \mathbb{G}^{1\times m}$ is set as the third row vector of $g^{\bar{\mathbf{A}}} \in \mathbb{G}^{3\times m}$. $\mathbf{v}^* \in \mathbb{Z}_p^{2m\times 1}$ is the exponent of $sk^* = g^{\mathbf{v}^*} \in \mathbb{G}^{2m\times 1}$ which was generated in **Leak**. $\mathcal{S}$ sets the challenge ciphertext C^* to $C^* := (C_1^*, C_2^*) := (g^{[\mathbf{y}|r^*\mathbf{y}]}, \hat{e}(g, g)^{[\mathbf{y}|r^*\mathbf{y}]\mathbf{v}^*} \cdot M_b)$, then sends it to $\mathcal{A}$.

Query 2: If $\mathcal{A}$ issues a query to the oracle **Reveal**, $\mathcal{S}$ behaves in the same manner as **Query**.

Guess(b')**:** $\mathcal{S}$ receives $b' \in \{0, 1\}$ sent by $\mathcal{A}$. $\mathcal{S}$ outputs $\beta' := 1$ if $b' = b$. $\mathcal{S}$ outputs $\beta' := 0$ if $b' \neq b$.

We now verify that $\mathcal{S}$ simulates $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) against $\mathcal{A}$ perfectly when the matrix $\bar{\mathbf{A}} \in \mathbb{Z}_p^{3\times m}$ is the matrix $\mathbf{A}_1$ (resp. $\mathbf{A}_0$).

Firstly, we consider the case that $\bar{\mathbf{A}}$ is $\mathbf{A}_1$. In this case, the third row of $\bar{\mathbf{A}}$, i.e., $\mathbf{y} \in \mathbb{Z}_p^{1\times m}$, is a linear combination of the first row and the second row of $\bar{\mathbf{A}}$, so there is $\mathbf{z}^* \in \mathbb{Z}_p^{1\times 2}$ such that $\mathbf{y} = \mathbf{z}^*\mathbf{A}_0$. Hence, we obtain $[\mathbf{y}|r^*\mathbf{y}] = [\mathbf{z}^*\mathbf{A}_0|\mathbf{z}^*(r^*\mathbf{A}_0)] = \mathbf{z}^*\mathbf{F}(ID^*)$, and $[\mathbf{y}|r^*\mathbf{y}]\mathbf{v}^* = \mathbf{z}^*\mathbf{F}(ID^*)\mathbf{v}^* = \mathbf{z}^*\mathbf{D}$. Therefore, the challenge ciphertext is written as $C^* = (C_1^*, C_2^*) = (g^{\mathbf{z}^*\mathbf{F}(ID^*)}, \hat{e}(g, g)^{\mathbf{z}^*\mathbf{D}} \cdot M_b)$. The challenge ciphertext C^* is the proper challenge ciphertext in $\mathtt{Game}_0$. Hence, if $\bar{\mathbf{A}}$ is $\mathbf{A}_1$, then $\mathcal{S}$ simulates $\mathtt{Game}_0$ to $\mathcal{A}$ perfectly.

Next, we consider another case that $\bar{\mathbf{A}}$ is $\mathbf{A}_0$. In this case, the third row of $\bar{\mathbf{A}}$, i.e., $\mathbf{y} \in \mathbb{Z}_p^{1\times m}$, distributes equivalently to a vector chosen uniformly at random, so the challenge ciphertext C^* generated by $\mathcal{S}$ is the proper challenge ciphertext in $\mathtt{Game}_1$ exactly. Hence, $\mathcal{S}$ simulates $\mathtt{Game}_1$ against $\mathcal{A}$ perfectly.

Therefore, we obtain $\mathtt{Adv}_{\mathcal{S}}^{\mathrm{DLIN}} = |\Pr[W_1] - \Pr[W_0]|$. $\square$

Proof of Lemma 3.3.4. At first, we define three events $U^{(pk,mk)}$, $V^{\mathcal{L}}_{(pk,mk)}$ and $V^{ID^*,f}_{(pk,mk),\mathcal{L}}$ for the game $\texttt{Game}_1$ or $\texttt{Game}_2$. Here, (pk, mk) is a pair of a system public-key pk and master-key mk generated in the phase **Setup**, $\mathcal{L}$ is a list generated in the phase **Query** under the queries issued by the adversary $\mathcal{A}$ given (pk, mk) in **Setup**, and (ID^*, f) is the pair of a target ID and a leakage function chosen by $\mathcal{A}$ in the phase **Leak** when (pk, mk) and $\mathcal{L}$ were generated.

The probability for each event $U^{(pk,mk)}$, $V^{\mathcal{L}}_{(pk,mk)}$ and $V^{ID^*,f}_{(pk,mk),\mathcal{L}}$ to happen is denoted by $\Pr[U^{(pk,mk)}]$, $\Pr[V^{\mathcal{L}}_{(pk,mk)}]$ and $\Pr[V^{ID^*,f}_{(pk,mk),\mathcal{L}}]$. We define a set $\mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$ as $\{\mathcal{L}\ s.t.\ \Pr[V^{\mathcal{L}}_{(pk,mk)}] > 0\}$. Likewise, we define a set $\mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}$ as $\{(ID^*, f)\ s.t.\ \Pr[V^{ID^*,f}_{(pk,mk),\mathcal{L}}] > 0\}$.

We define games $\texttt{Game}_1^{(pk,mk),\mathcal{L},(ID^*,f)}$ and $\texttt{Game}_2^{(pk,mk),\mathcal{L},(ID^*,f)}$, and events $W_1^{(pk,mk),\mathcal{L},(ID^*,f)}$ and $W_2^{(pk,mk),\mathcal{L},(ID^*,f)}$. $\texttt{Game}_1^{(pk,mk),\mathcal{L},(ID^*,f)}$ is the following game where a PPT adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$ communicate each other:

Setup: $\mathcal{CH}$ runs $sk^* \leftarrow \mathsf{KeyGen}(pk, mk, ID^*)$, then $\mathcal{CH}$ computes $f(sk^*)$. $\mathcal{CH}$ sends $(pk, \mathcal{L}, ID^*, f, f(sk^*))$ to $\mathcal{A}$.

Challenge(M_0, M_1)**:** $\mathcal{CH}$ receives two distinct plaintexts $M_0, M_1 \in \mathcal{M}$ from $\mathcal{A}$. $\mathcal{CH}$ sets $C_1^* := g^{[\mathbf{y}|r^*\mathbf{y}]}$ and $C_2^* := \hat{e}(g, g)^{[\mathbf{y}|r^*\mathbf{y}]\mathbf{v}^*} \cdot M_b$, where $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$, $\mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$, $r^* := r_0 + \sum_{i=1}^{n} r_i \cdot ID^*[i]$, and $\mathbf{v}^* \in \mathbb{Z}_p^{1\times 2m}$ is the exponent of the secret-key $sk^* := g^{\mathbf{v}^*}$. $\mathcal{CH}$ sends $C^* := (C_1^*, C_2^*)$ to $\mathcal{A}$.

Query 2: $\mathcal{A}$ is allowed to use the oracle **Reveal** adaptively.

Reveal$(ID \neq ID^*)$**:** $\mathcal{CH}$ runs $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, then sends it to $\mathcal{A}$.

Guess$(b' \in \{0, 1\})$**:** $\mathcal{A}$ sends a guess $b' \in \{0, 1\}$ for b.

$\texttt{Game}_2^{(pk,mk),\mathcal{L},(ID^*,f)}$ is the same as $\texttt{Game}_1^{(pk,mk),\mathcal{L},(ID^*,f)}$ except that the second element C_2^* of the challenge ciphertext C^* is set to $C_2^* := \hat{e}(g, g)^u$, where $u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$.

Let $W_{1,(pk,mk),\mathcal{L},(ID^*,f)}$ (resp. $W_{2,(pk,mk),\mathcal{L},(ID^*,f)}$) denote the event where $\mathcal{A}$ outputs b' such that $b' = b$ in $\texttt{Game}_1^{(pk,mk),\mathcal{L},(ID^*,f)}$ (resp. $\texttt{Game}_2^{(pk,mk),\mathcal{L},(ID^*,f)}$).

By the definitions of W_1, $U^{(pk,mk)}$, $V^{\mathcal{L}}_{(pk,mk)}$, $V^{ID^*,f}_{(pk,mk),\mathcal{L}}$ and $W_{2,(pk,mk),\mathcal{L},(ID^*,f)}$, we obtain

$$\Pr[W_1] = \sum_{(pk,mk)\leftarrow \mathsf{Setup}(1^\lambda)} \sum_{\mathcal{L}\in\mathbb{Q}^{\mathcal{L}}_{(pk,mk)}} \sum_{(ID^*,f)\in\mathbb{Q}^{(ID^*,f)}_{(pk,mk),\mathcal{L}}} \Pr\left[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}\right] \cdot \Pr\left[V^{ID^*,f}_{(pk,mk),\mathcal{L}}\right] \cdot \Pr\left[V^{\mathcal{L}}_{(pk,mk)}\right] \cdot \Pr\left[U^{(pk,mk)}\right] \quad (3.4)$$

Likewise, we obtain an equation for $\Pr[W_2]$. Hence, we obtain

$$
\begin{aligned}
&|\Pr[W_1] - \Pr[W_2]| \\
= &\left| \sum_{(pk,mk)\leftarrow \mathsf{Setup}(1^\lambda)} \sum_{\mathcal{L}\in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}} \sum_{(ID^*,f)\in \mathbb{Q}^{(ID^*,f)}_{(pk,mk),\mathcal{L}}} \left(\Pr\left[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}\right] - \Pr\left[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}\right]\right) \right. \\
&\left. \cdot \Pr\left[V^{ID^*,f}_{(pk,mk),\mathcal{L}}\right] \cdot \Pr\left[V^{\mathcal{L}}_{(pk,mk)}\right] \cdot \Pr\left[U^{(pk,mk)}\right] \right| \\
\leq &\sum_{(pk,mk)\leftarrow \mathsf{Setup}(1^\lambda)} \sum_{\mathcal{L}\in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}} \sum_{(ID^*,f)\in \mathbb{Q}^{(ID^*,f)}_{(pk,mk),\mathcal{L}}} \left|\Pr\left[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}\right] - \Pr\left[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}\right]\right| \\
&\cdot \Pr\left[V^{ID^*,f}_{(pk,mk),\mathcal{L}}\right] \cdot \Pr\left[V^{\mathcal{L}}_{(pk,mk)}\right] \cdot \Pr\left[U^{(pk,mk)}\right].
\end{aligned}
$$

By Lemma 3.3.6, there exists a negligible function $negl(\lambda)$ s.t.

$$
\begin{aligned}
&|\Pr[W_1] - \Pr[W_2]| \leq negl(\lambda) \\
&\cdot \left\{ \sum_{(pk,mk)\leftarrow \mathsf{Setup}(1^\lambda)} \sum_{\mathcal{L}\in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}} \sum_{(ID^*,f)\in \mathbb{Q}^{(ID^*,f)}_{(pk,mk),\mathcal{L}}} \Pr\left[V^{ID^*,f}_{(pk,mk),\mathcal{L}}\right] \cdot \Pr\left[V^{\mathcal{L}}_{(pk,mk)}\right] \cdot \Pr\left[U^{(pk,mk)}\right] \right\}.
\end{aligned} \tag{3.5}
$$

We give three facts. It holds that $\sum_{(pk,mk)\leftarrow \mathsf{Setup}(1^\lambda)} \Pr\left[U^{(pk,mk)}\right] = 1$. It also holds that for any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, $\sum_{\mathcal{L}\in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}} \Pr\left[V^{\mathcal{L}}_{(pk,mk)}\right] = 1$. It also holds that for any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$ and any $\mathcal{L} \in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$, $\sum_{(ID^*,f)\in \mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}} \Pr\left[V^{ID^*,f}_{(pk,mk),\mathcal{L}}\right] = 1$.

By the above three facts and (3.5), it is true that for any PPT $\mathcal{A}$, there exists a negligible function $negl(\lambda)$ such that $|\Pr[W_1] - \Pr[W_2]| \leq negl(\lambda)$. $\square$

Lemma 3.3.6. *For any PPT $\mathcal{A}$, any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$ and any $(ID^*, f) \in \mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}$, there exists $negl(\lambda)$ such that $|\Pr[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}] - \Pr[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}]| \leq negl(\lambda)$.*

Proof of Lemma 3.3.6. To prove the lemma, we use Lemma 3.3.7.

We consider a PPT simulator $\mathcal{S}$. $\mathcal{S}$ behaves as the distinguisher $\mathcal{D}$ in Lemma 3.3.7. $\mathcal{S}$ behaves concurrently as the challenger in $\mathtt{Game}_1^{(pk,mk)}$ or $\mathtt{Game}_2^{(pk,mk)}$, so $\mathcal{S}$ has to simulate each game against $\mathcal{A}$. The concrete behavior of $\mathcal{S}$ is as follows:

Setup: $\mathcal{S}$ is the distinguisher $\mathcal{D}$ in Lemma 3.3.7, so $\mathcal{S}$ is given $(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], s)$, where $s \in \mathbb{Z}_p$ is $[\mathbf{y}|r^*\mathbf{y}]\,\mathbf{v}^*$ or u. $\mathcal{S}$ sends $(pk, \mathcal{L}, ID^*, f, f(sk^*))$ to $\mathcal{A}$.

Challenge(M_0, M_1): $\mathcal{S}$ receives two plaintexts $M_0, M_1 \in \mathcal{M}$ sent from $\mathcal{A}$. $\mathcal{S}$ sets $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$, then sets $C_1^* := g^{[\mathbf{y}|r^*\mathbf{y}]} \in \mathbb{G}^{1\times 2m}$ and $C_2^* := \hat{e}(g, g)^s \cdot M_b$. After that, $\mathcal{S}$ sends $C^* := (C_1^*, C_2^*)$ to $\mathcal{A}$.

Query 2: When $\mathcal{A}$ issues a query to **Reveal**, $\mathcal{S}$ acts as follows:

Reveal$(ID \neq ID^*)$**:** $\mathcal{S}$ runs $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, then sends sk to $\mathcal{A}$.

Guess$(b' \in \{0,1\})$**:** $\mathcal{S}$ receives $b' \in \{0,1\}$ from $\mathcal{A}$. If $b' = b$, then $\mathcal{S}$ outputs $\beta' := 1$. Otherwise, then $\mathcal{S}$ outputs $\beta' := 0$.

For any PPT $\mathcal{A}$, any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$ and any $(ID^*, f) \in \mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}$, we obtain $\Pr[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}] = \Pr[\mathcal{S}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], [\mathbf{y}|r^*\mathbf{y}]\mathbf{v}^*) \to 1 | (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}]$ and
$\Pr[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}] = \Pr[\mathcal{S}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], u) \to 1 |$
$(sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p]$.

Hence, we obtain

$$\begin{aligned}
\delta &:= \left|\Pr\left[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}\right] - \Pr\left[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}\right]\right| \\
&= \Big|\Pr\left[\mathcal{S}\left(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], [\mathbf{y}|r^*\mathbf{y}]\,\mathbf{v}^*\right) \to 1 \right. \\
&\quad \left.\Big|(sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}\right] \\
&\quad - \Pr\left[\mathcal{S}\left(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], u\right) \to 1 \right. \\
&\quad \left.\Big|(sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p\right]\Big|.
\end{aligned} \tag{3.6}$$

Lemma 3.3.7 guarantees that there exists a PPT inverter $\mathcal{B}$ such that

$$\begin{aligned}
&\Pr\left[\mathcal{B}\left(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), H^*\right) \to sk^* \right. \\
&\quad \left.\Big|(sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*)\right] \geq \frac{\delta^3}{1024 \cdot m^2 \cdot p^3 \cdot l}.
\end{aligned} \tag{3.7}$$

We assume that there exists a polynomial function $poly(\lambda)$ such that $\delta \geq 1/poly(\lambda)$. Since p is $p < 2^\lambda$, m is $m = (4\lambda)^{1/\epsilon}$, and l can be written as a polynomial function $poly^*(\lambda)$, we obtain

$$\frac{\delta^3}{1024 \cdot m^2 \cdot p^3 \cdot l} > \frac{1}{1024 \cdot (4\lambda)^{2/\epsilon} \cdot 2^{3\lambda} \cdot poly(\lambda)^3 \cdot poly^*(\lambda)} > 2^{-3\lambda} \cdot 2^{-\lambda} = 2^{-4\lambda} = 2^{-m^\epsilon}. \tag{3.8}$$

(3.7) and (3.8) lead us to a contradiction against the fact that $f \in \mathcal{F}_{(pk,mk),\mathcal{L},(ID^*,f)}(2^{-m^\epsilon})$. Hence, there are no polynomial functions $poly(\lambda)$ such that $\delta \geq 1/poly(\lambda)$. Therefore, by (3.6), for any PPT $\mathcal{A}$, any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$ and any $(ID^*, f) \in \mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}$, $\left|\Pr\left[W_{1,(pk,mk),\mathcal{L},(ID^*,f)}\right] - \Pr\left[W_{2,(pk,mk),\mathcal{L},(ID^*,f)}\right]\right|$ is negligible. □

Proof of the following Lemma 3.3.7 is given below.

Lemma 3.3.7. *The notations of pk, mk, $\mathcal{L}$, ID^* and f are the same as the ones in the proof of Lemma 3.3.4. pk and mk are parsed as $pk = (p, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^{\mathbf{A}_0}, g^{\mathbf{A}'_0}, g^{\mathbf{A}_1}, \cdots, g^{\mathbf{A}_n}, g^{\mathbf{D}})$ and $mk = (r_0, r_1, \cdots, r_n, \mathbf{E})$, respectively. r^* is set as $r^* := r_0 + \sum_{i=1}^{n} r_i \cdot ID^*[i]$.*

We assume that there exists a distinguisher $\mathcal{D}$ running in time t such that

$$\begin{aligned}\delta \ :=\ & \Big| \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], [\mathbf{y}|r^*\mathbf{y}]\,\mathbf{v}^*) \to 1 \\ & \quad \Big| (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m} \big] \\ & - \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], u) \to 1 \\ & \quad \Big| (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p\big]\Big|,\end{aligned}$$

where $sk^* = g^{\mathbf{v}^*}$ and $\mathbf{v}^* \in \mathbb{Z}_p^{2m\times 1}$. If such a distinguisher exists, we can construct an inverter $\mathcal{B}$ running in time $t' = t \cdot poly(m, l, \frac{1}{\delta})$ such that

$$\begin{aligned}&\Pr\big[\mathcal{B}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), H^*) \to sk^* \\ &\quad \big| (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*)\big] \geq \frac{\delta^3}{1024 \cdot m^2 \cdot p^3 \cdot l}.\end{aligned}$$

<u>Proof of Lemma 3.3.7.</u> The proof is similar to the proof of Goldreich-Levin Theorem for large fields by Dodis et al. [DGK+10].

Consider a PPT adversary $\mathcal{A}$, a pair of keys $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, a list $\mathcal{L} \in \mathbb{Q}^{\mathcal{L}}_{(pk,mk)}$, and a pair $(ID^*, f) \in \mathbb{Q}^{ID^*,f}_{(pk,mk),\mathcal{L}}$.

We assume that there exists a distinguisher $\mathcal{D}$ running in time t such that

$$\begin{aligned}\delta := & \Big| \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], [\mathbf{y}|r^*\mathbf{y}]\,\mathbf{v}^*) \to 1 \\ & \Big| r' \overset{\mathrm{U}}{\leftarrow} \{0,1\}^*, (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*; r'), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}\big] \\ & - \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], u) \to 1 \\ & \Big| r' \overset{\mathrm{U}}{\leftarrow} \{0,1\}^*, (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*; r'), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p\big]\Big|, \end{aligned} \tag{3.9}$$

where $r' \in \{0,1\}^*$ is the randomness which is used when running KeyGen'_l. As is obvious from the definition of KeyGen' of Π_{IBE}, $|H^*| \leq 2ml$.

For a randomness $r' \in \{0,1\}^*$, $\alpha_{r'}$ and $\beta_{r'}$ denote the following.

$$\begin{aligned}\alpha_{r'} \ :=\ & \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], [\mathbf{y}|r^*\mathbf{y}]\,\mathbf{v}^*) \to 1 \\ & \Big| (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*; r'), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}\big] \end{aligned} \tag{3.10}$$

$$\begin{aligned}\beta_{r'} \ :=\ & \Pr\big[\mathcal{D}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), [\mathbf{y}|r^*\mathbf{y}], u) \to 1 \\ & \Big| (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*; r'), \mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}, u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p\big] \end{aligned} \tag{3.11}$$

By (3.9), we obtain $\mathbb{E}_{r'}[\alpha_{r'} - \beta_{r'}] \geq \delta$. In this proof, we say that $r' \in \{1,0\}^*$ is good when r' satisfies $\alpha_{r'} - \beta_{r'} \geq \delta/2$. By the averaging argument, we obtain

$$\Pr\Big[r' \texttt{ is good.} \Big| r' \overset{\mathrm{U}}{\leftarrow} \{0,1\}^*\Big] \geq \delta/2. \tag{3.12}$$

Let $n' := 128 \cdot (2ml)^2 \cdot m/\delta^2 = 512l^2m^3/\delta^2$. Let $c \in \mathbb{N}$ be the minimum integer greater than or equal to 2 which satisfies $p^c \geq 512l^2m^3/\delta^2$.

An inverter $\mathcal{B}$ is given $(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), H^*)$ as inputs, where (sk^*, H^*) were randomly generated as $(sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*; r')$. The randomness $r' \in \{0,1\}^*$ is supposed to be "good". $\mathcal{B}$ behaves as follows:

- Choose $r^\dagger \overset{\text{U}}{\leftarrow} \mathbb{Z}_p$ as a guess for $r^* \in \mathbb{Z}_p$. Obviously,
$$\Pr\left[r^\dagger = r^*\right] = 1/p. \tag{3.13}$$
- Calculate the probability $\kappa_{r'} \in [0,1]$ such that $\alpha_{r'} - \delta/8 \geq \kappa_{r'} \geq \beta_{r'} + \delta/8$ as follows: Generate $O(n^\dagger/\delta^2)$ pairs of $(\mathbf{y}, u)$ by $\mathbf{y} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$ and $u \overset{\text{U}}{\leftarrow} \mathbb{Z}_p$. For each pair of $(\mathbf{y}, u)$, calculate the value of $\beta_{r'}$. Let e denote the average of the calculated values of $\beta_{r'}$. By the Chernoff's bound, e satisfies $\Pr[|e - \beta_{r'}| > \delta/8] \leq 2^{-n^\dagger}$. We set $\kappa_{r'}$ as $\kappa_{r'} := e + \delta/4$.
- Generate a set $S \subseteq \mathbb{Z}_p^{c\times 1} \setminus 0^{c\times 1}$ of cardinality n' such that every distinct two elements $\mathbf{u}_1, \mathbf{u}_2 \in S$ is linearly independent as follows: Generate a set S such that every element of the set is a vector whose the $(1,1)$-th element is 1.
- Set $\mathbf{z}_1, \cdots, \mathbf{z}_c \overset{\text{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$ and $h_1, \cdots, h_c \overset{\text{U}}{\leftarrow} \mathbb{Z}_p$. For every $\bar{\rho} = (\rho_1, \cdots, \rho_c) \in S$, where $\rho_1, \cdots, \rho_c \in \mathbb{Z}_p$, set $\mathbf{y}_{\bar{\rho}} := \sum_{i=1}^{c} \rho_i \mathbf{z}_i$ and $k_{\bar{\rho}} := \sum_{i=1}^{c} \rho_i h_i$.
- For $i \in [1, m]$, repeat the following procedures until the guess for (v_i^*, v_{m+i}^*) is determined:
  - A guess $a \in H^*$ (resp. $b \in H^*$) for v_i^* (resp. v_{m+i}^*) is chosen randomly.
  - For every $\bar{\rho} \in S$, do:
    * Set $\tau_{\bar{\rho}}^{i,a,b} \overset{\text{U}}{\leftarrow} \mathbb{Z}_p$.
    * Give the following tuple to $\mathcal{D}$ as an input, and then observe the output, where $\mathbf{e}_i$ denotes the unit column vector whose i-th element is 1:
$$\begin{aligned}\Big(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), \left[\mathbf{y}_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b}\mathbf{e}_i \middle| r^\dagger \mathbf{y}_{\bar{\rho}} + r^\dagger \tau_{\bar{\rho}}^{i,a,b}\mathbf{e}_i\right], \\ k_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} a + r^\dagger \tau_{\bar{\rho}}^{i,a,b} b\Big).\end{aligned} \tag{3.14}$$
  - Calculate the average $s^{i,a,b} \in [0,1]$ of the outputs of the $\mathcal{D}$s. Then, if $s^{i,a,b} \geq \kappa_{r'}$, the guess of (a,b) for (v_i^*, v_{m+i}^*) is determined, and set $v_i^\dagger := a$ and $v_{m+i}^\dagger := b$. Else if $s^{i,a,b} < \kappa_{r'}$, choose another guess of $(a,b) \in H^* \times H^*$ for (v_i^*, v_{m+i}^*), and repeat the above procedures until the guess is determined.
- Set $\mathbf{v}^\dagger \in \mathbb{Z}_p^{2m\times 1}$ as a column vector whose the i-th element is $v_i^\dagger$ where $i \in [1, 2m]$. Output $sk^\dagger := g^{\mathbf{v}^\dagger} \in \mathbb{G}^{2m\times 1}$ as a (determined) guess for $sk^* = g^{\mathbf{v}^*}$.

We consider the running time of $\mathcal{B}$. Since $\kappa_{r'}$ can be calculated efficiently [DGK+10], we can ignore that. Hence, the running time of $\mathcal{B}$ is approximately equivalent to the time which $\mathcal{B}$ needs to run $\mathcal{D}$. $\mathcal{B}$ runs the distinguisher $m \cdot |H^*|^2 \cdot n' \leq 2048l^4m^6/\delta^2$ times in the worst case. The running time of $\mathcal{D}$ is assumed to be t, so the order of the running time of $\mathcal{B}$ is $t \cdot poly(m, l, 1/\delta)$.

We two events. E_1 is defined as the event where the guess $r^\dagger$ for r^* by $\mathcal{B}$ is correct. Obviously, $\Pr[E_1] = \Pr\left[r^\dagger = r^*\right] = 1/p$. E_2 is defined as the event where for every $\bar{\rho} \in S$, $\mathbf{y}_{\bar{\rho}}$ and $k_{\bar{\rho}}$ satisfy $k_{\bar{\rho}} = \left[\mathbf{y}_{\bar{\rho}} \middle| r^*\mathbf{y}_{\bar{\rho}}\right]\mathbf{v}^*$. Obviously, $\Pr[E_2] = \Pr\left[\forall\bar{\rho} \in S, k_{\bar{\rho}} = \left[\mathbf{y}_{\bar{\rho}} \middle| r^*\mathbf{y}_{\bar{\rho}}\right]\mathbf{v}^*\right] \geq \Pr\left[\forall j \in [1, c], h_j = \left[\mathbf{z}_j \middle| r^*\mathbf{z}_j\right]\mathbf{v}^*\right] = 1/p^c$. E_1 and E_2 are independent, so we obtain

$$\Pr[E_1 \wedge E_2] = \Pr[E_2]\Pr[E_1] \geq 1/p^{c+1}. \tag{3.15}$$

Below, we explain that $\mathcal{B}$ can correctly guess sk^* at least with probability $1/2$ in the case when $E_1 \wedge E_2$ occurs.

Firstly, we consider the case that both of the guess of $a \in H^*$ for v_i^* by $\mathcal{B}$ and the guess of $b \in H^*$ for v_{m+i}^* are correct. $\mathcal{B}$ gives the input (3.14) to $\mathcal{D}$, and observes the output. By Lemma 3.3.8, in the case that $a = v_i^* \wedge b = v_{m+i}^*$, the input given to $\mathcal{D}$ by $\mathcal{B}$ and the input given to $\mathcal{D}$ in (3.10) distribute identically. Hence, the probability that the distinguisher $\mathcal{D}$ whom $\mathcal{B}$ runs outputs 1 is exactly $\alpha_{r'}$. Therefore, the probability that $\mathcal{B}$ lets through the correct guess of (a, b) is, by Chebyshev's inequality, $\Pr\left[s^{i,a,b} \leq \kappa_{r'}\right] \leq \frac{1}{n' \cdot (\delta/8)^2} = \frac{1}{8l^2m^3}$.

Secondly, we consider the case that either the guess of $a \in H^*$ for v_i^* by $\mathcal{B}$ or the guess of $b \in H^*$ for v_{m+i}^* is wrong. By Lemma 3.3.8, in the case that $a \neq v_i^* \vee b \neq v_{m+i}^*$, the input given to $\mathcal{D}$ by $\mathcal{B}$ and the input given to $\mathcal{D}$ in (3.11) distribute identically. Hence, the probability that the distinguisher $\mathcal{D}$ whom $\mathcal{B}$ runs outputs 1 is exactly $\beta_{r'}$. Therefore, the probability that $\mathcal{B}$ determines the wrong guess of (a, b) is, by Chebyshev's inequality, $\Pr\left[s^{i,a,b} \geq \kappa_{r'}\right] \leq \frac{1}{n' \cdot (\delta/8)^2} = \frac{1}{8l^2m^3}$.

Hence, the probability that $\mathcal{B}$ fails to guess $sk^* = g^{\mathbf{v}^*}$ correctly is at most $\sum_{j=1}^{|H^*|^2 \cdot m} 1/8l^2m^3 = |H^*|^2 \cdot m/8l^2m^3 \leq 4l^2m^3/8l^2m^3 = 1/2$. We can also say that the probability that $\mathcal{B}$ guesses $sk^* = g^{\mathbf{v}^*}$ correctly is at least $1/2$. By this fact, (3.12) and (3.15), we obtain

$$\begin{aligned}
&\Pr\Big[\mathcal{B}\Big(pk, mk, ID^*, H^*, \mathcal{L}^\dagger_{q_\mathcal{A}}, \{f_i, \mathcal{L}_{ID,i}, f_i(pk, mk, \mathcal{L}_i)\}_{i\in[1,q_\mathcal{A}]}\Big) \to sk^* \\
&\Big|(sk^*, H^*, \mathcal{L}^\dagger_{q_\mathcal{A}}, \{f_i(pk, mk, \mathcal{L}_i)\}_{i\in[1,q_\mathcal{A}]}) \leftarrow \textsf{PrmRfr}(pk, mk, R_\mathcal{A})\Big] \\
&\geq \frac{\delta}{2} \cdot \frac{1}{p^{c+1}} \cdot \frac{1}{2} = \frac{\delta}{4p^{c+1}}.
\end{aligned} \tag{3.16}$$

If $p \geq 512l^2m^3/\delta^2$, then by the definition of $c \in \mathbb{N}$, c is $c = 2$. It obviously holds that $p^3 < p^3 \cdot 256 \cdot l \cdot m^2/\delta^2$. Hence, we obtain $\delta/4p^{c+1} = \delta/4p^3 > \delta^3/1024 \cdot l \cdot m^2 \cdot p^3$.

Else if $p < 512l^2m^3/\delta^2$, then by the definition of $c \in \mathbb{N}$, we obtain the following two inequalities, $p^c \geq 512 \cdot l^2 \cdot m^3/\delta^2$ and $p^c \leq p \cdot 512 \cdot l^2 \cdot m^3/\delta^2$. By the latter inequality, it obviously holds that $p^{c+1} \leq p^2 \cdot 512 \cdot l^2 \cdot m^3/\delta^2$. Hence, by $p > 2lm$, we obtain $\delta/4^{c+1} \geq \delta^3/2048 \cdot l^2 \cdot m^3 \cdot p^2 > \delta^3/1024 \cdot l \cdot m^2 \cdot p^3$.

□

Lemma 3.3.8. *We assume that the event $E_1 \wedge E_2$ occurs. For any $i \in [1, m]$, if $a = v_i^* \wedge b = v_{m+i}^*$ (resp. $a \neq v_i^* \vee b \neq v_{m+i}^*$), the input which is given to $\mathcal{D}$ by $\mathcal{B}$ and the input which is given to $\mathcal{D}$ in (3.10) (resp. (3.11)) distribute identically.*

Proof of Lemma 3.3.8. Firstly, we consider the case that $a = v_i^* \wedge b = v_{m+i}^*$. Since we assume that the event $E_1 \wedge E_2$ occurs, the last element of the input which is given to $\mathcal{D}$ by $\mathcal{B}$ is transformed as follows: $k_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} a + r^\dagger \tau_{\bar{\rho}}^{i,a,b} b = \left[\mathbf{y}_{\bar{\rho}} \middle| r^* \mathbf{y}_{\bar{\rho}}\right] \mathbf{v}^* + \tau_{\bar{\rho}}^{i,a,b} a + r^* \tau_{\bar{\rho}}^{i,a,b} b = \left[\mathbf{y}_{\bar{\rho}} \middle| r^* \mathbf{y}_{\bar{\rho}}\right] \mathbf{v}^* + \tau_{\bar{\rho}}^{i,a,b} v_i^* + r^* \tau_{\bar{\rho}}^{i,a,b} v_{m+i}^* = \left[\mathbf{y}_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i \middle| r^* \mathbf{y}_{\bar{\rho}} + r^* \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i\right] \mathbf{v}^*$. Since any two elements in S are linearly independent and $\tau_{\bar{\rho}}^{i,a,b}$ is uniformly random in $\mathbb{Z}_p$, $\mathbf{y}_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i \in \mathbb{Z}_p^{1\times m}$ which is generated by $\mathcal{B}$ distributes identically to $\mathbf{y}$ which is generated by $\mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$. Hence, second to the last element in the input given to $\mathcal{D}$ by $\mathcal{B}$ distributes identically to the input given to $\mathcal{D}$ in (3.10). Therefore, if $a = v_i^* \wedge b = v_{m+i}^*$, the input given to $\mathcal{D}$ by $\mathcal{B}$ distributes identically to the input given to $\mathcal{D}$ in (3.10).

Secondly, we prove the lemma in the case that $a \neq v_i^* \vee b \neq v_{m+i}^*$. We prove the lemma only in the case that $a \neq v_i^* \wedge b = v_{m+i}^*$. Neither proof of the lemma for the case that $a = v_i^* \wedge b \neq v_{m+i}^*$ nor the case that $a \neq v_i^* \wedge b \neq v_{m+i}^*$ is omitted, since the proof for each case is almost the same as the case that $a \neq v_i^* \wedge b = v_{m+i}^*$.

In this case, there exists $u_i^* \in \mathbb{Z}_p$ such that $a - v_i^* = u_i^* \neq 0$. Then, the last element in the input given to $\mathcal{D}$ by $\mathcal{B}$ is transformed as follows: $k_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} a + r^\dagger \tau_{\bar{\rho}}^{i,a,b} b = \left[\mathbf{y}_{\bar{\rho}} \middle| r^* \mathbf{y}_{\bar{\rho}}\right] \mathbf{v}^* + \tau_{\bar{\rho}}^{i,a,b} (v_i^* + u_i^*) + r^* \tau_{\bar{\rho}}^{i,a,b} v_{m+i}^* = \left[\mathbf{y}_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i \middle| r^* \mathbf{y}_{\bar{\rho}} + r^* \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i\right] \mathbf{v}^* + \tau_{\bar{\rho}}^{i,a,b} u_i^*$. Since any two elements in S are linearly independent and $\tau_{\bar{\rho}}^{i,a,b}$ is uniformly random in $\mathbb{Z}_p$, $\mathbf{y}_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} \mathbf{e}_i \in \mathbb{Z}_p^{1\times m}$ which is generated by $\mathcal{B}$ distributes identically to $\mathbf{y}$ which is generated by $\mathbf{y} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^{1\times m}$. Moreover, since $\tau_{\bar{\rho}}^{i,a,b}$ is uniformly random in $\mathbb{Z}_p$, $k_{\bar{\rho}} + \tau_{\bar{\rho}}^{i,a,b} a + r^\dagger \tau_{\bar{\rho}}^{i,a,b} b$ generated by $\mathcal{B}$ distributes identically to u which is generated by $u \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. Therefore, if $a \neq v_i^* \wedge b = v_{m+i}^*$, the input given to $\mathcal{D}$ by $\mathcal{B}$ distributes identically to the input given to $\mathcal{D}$ in (3.11). $\square$

3.3.3 Specific Examples of Allowed Leakage-Functions

In this subsection, we consider the leakage allowed for our IBE scheme Π_{IBE}. We write the secret user-key sk^* as $sk^* = g^{\mathbf{v}^*} \in \mathbb{G}^{2m\times 1}$, where $\mathbf{v}^* \in \mathbb{Z}_p^{2m\times 1}$. For $i \in [1, 2m]$, the $(i, 1)$-th element of $\mathbf{v}^*$ is denoted by v_i^*.

At first, we consider the case that the leakage function leaks no information about sk^*. As is obvious from the concrete construction of our IBE scheme, for any $i \in [1, m]$, v_i^* and v_{m+i}^* are dependent each other. Hence, any PPT $\mathcal{B}$ in the definition of leakage-function class which is given some information about v_i^* can get some information about v_{m+i}^*, and vice versa. On the contrary, if neither information about v_i^* nor v_{m+i}^* is leaked, any PPT $\mathcal{B}$ cannot identify either v_i^* or v_{m+i}^* with probability greater than $1/l$, information-theoretically. Therefore, the following statement is true, information-theoretically: for any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}_{(pk,mk)}^{\mathcal{L}}$, any ID^* which is not in the list $\mathcal{L}$, any function f which leaks no information about sk^*, and any PPT $\mathcal{B}$, it holds that $\Pr[\mathcal{B}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), H^*) \to sk^* | (sk^*, H^*) \leftarrow \mathsf{KeyGen}'_l(pk, mk, ID^*)] \leq 1/l^m$. Since $m > 4\lambda$, if we set l such that $l > 2$, then $l^{-m} < 2^{-4\lambda} = 2^{-m^\epsilon}$. Hence, for any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}_{(pk,mk)}^{\mathcal{L}}$ and any ID^* which is not in the list $\mathcal{L}$, any leakage function f which leaks no information about sk^* is allowed to query.

Next, we consider the case that the leakage function leaks some information about sk^*. Obviously, the leakage function which fully leaks sk^* is not an allowed one, because any

PPT $\mathcal{B}$ given the leakage information can identify sk^* with probability 1. Hereafter, we let c be a constant positive real number, and we consider a leakage function which satisfies the following condition: among m "pairs" of $(v_1^*, v_{m+1}^*), \cdots, (v_{m-1}^*, v_{2m}^*)$, there are at least $4\lambda/c$ pairs of (v_i^*, v_{m+i}^*), where $i \in [1, m]$, such that the leakage function does not leak any information about either v_i^* or v_{m+i}^*.

The following statement is true, information-theoretically: for any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}_{(pk,mk)}^{\mathcal{L}}$, any ID^* which is not in the list $\mathcal{L}$, any leakage function f which satisfies the above condition, and any PPT $\mathcal{B}$, it holds that $\Pr[\mathcal{B}(pk, mk, \mathcal{L}, ID^*, f, f(sk^*), H^*) \rightarrow sk^* | (sk^*, H^*) \rightarrow \mathsf{KeyGen}'_l(pk, mk, ID^*)] \leq 1/l^{4\lambda/c}$. If we set l such that $l > 2^c$, then $l^{-4\lambda/c} < 2^{-4\lambda} = 2^{-m^\epsilon}$. Hence, For any $\lambda \in \mathbb{N}$, any $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$, any $\mathcal{L} \in \mathbb{Q}_{(pk,mk)}^{\mathcal{L}}$, any ID^* which is not in the list $\mathcal{L}$, any leakage function f which satisfies the above condition is allowed to query.

How Large is the Amount of Leakage Allowed for Our IBE Scheme? In the above, we considered a leakage function such that there are at least $4\lambda/c$ pairs of (v_j^*, v_{m+j}^*), where $j \in [1, m]$, any one of which does not leak any information about either v_j^* or v_{m+j}^*. We did not impose any restriction on the leakage function regarding the remaining $m - 4\lambda/c$ pairs. We can allow the function to fully leak every one of the $m - 4\lambda/c$ pairs. The bigger the parameter $m = (4\lambda)^{1/\epsilon}$ whose ϵ is $0 < \epsilon < 1$ is, the bigger the number $m - 4\lambda/c$ of pairs which can be fully leaked is. Therefore, the amount of allowed leakage linearly increases with the size of the secret user-key.

3.4 The Results in [YCZY12]

In this section, the results by Yuen et al. [YCZY12] is described. In Subsect. 3.4.1, bilinear groups of composite order is explained. In Sebsect. 3.4.2, definitions of decisional subgroup (DSG) assumptions introduced in [YCZY12] are given. In Subsect. 3.4.3 and Subsect. 3.4.4, their definition of the ciphertext indistinguishability for IBE schemes and their concrete construction of IBE scheme are described, respectively. They adopts the dual system encryption methodology, so defining the semi-functional secret-key and ciphertext is needed. The definitions are given in Subsect. 3.4.5. The proof sketch for the security of their IBE construction is given in Subsect. 3.4.6. Our counterexamples against their security proof will be described in the next section.

3.4.1 Bilinear Groups of Composite Order

$\mathcal{G}_{comp}$ denotes a generator of bilinear groups of composite order. $\mathcal{G}_{comp}$ takes 1^λ as input, and outputs $(N = p_1 p_2 p_3, \mathbb{G}, \mathbb{G}_T, \hat{e})$, where p_1, p_2 and p_3 are distinct λ-bit primes, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order N, and $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a map computable in polynomial time which satisfies the following conditions. Firstly, for every $g, h \in \mathbb{G}$ and $a, b \in \mathbb{Z}_N$, it holds that $\hat{e}(g^a, h^b) = \hat{e}(g, h)^{ab}$. Secondly, if g is a generator of $\mathbb{G}$, $\hat{e}(g, g)$ becomes a generator of $\mathbb{G}_T$.

$\mathbb{G}_{p_i}$, where $i \in \{1, 2, 3\}$, denotes a subgroup of order p_i in $\mathbb{G}$. For every $i, j \in \{1, 2, 3\}$ such that $i \neq j$, for every $h_i \in \mathbb{G}_{p_i}$ and $h_j \in \mathbb{G}_{p_j}$, it holds that $\hat{e}(h_i, h_j) = 1$. $\mathbb{G}_{p_i p_j}$, where

$i, j \in \{1,2,3\}$ such that $i \neq j$, denotes a subgroup of order $p_i p_j$ in $\mathbb{G}$. Any element $T \in \mathbb{G}_{p_i p_j}$ can be uniquely written as the product of an element in $\mathbb{G}_{p_i}$ and an element in $\mathbb{G}_{p_j}$.

3.4.2 Decisional Subgroup (DSG) Assumptions

DSG Assumption 1 [YCZY12]. We consider a problem whose advantage of a PPT adversary $\mathcal{A}$ is defined as:

$$\begin{aligned}&\mathtt{Adv}^{(1)}_{\mathcal{A},DSG}(\lambda)\\ &:= \Pr\Big[b \leftarrow \mathcal{A}(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1, X_2, X_3, T_b)\Big|(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{comp}(1^\lambda),\\ &g, X_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_1}, X_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_2}, X_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_3}, T_0 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_1p_2}, T_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_1}, b \overset{\mathrm{U}}{\leftarrow} \{0,1\}\Big].\end{aligned}$$

DSG Assumption 2 [YCZY12]. We consider a problem whose advantage of a PPT adversary $\mathcal{A}$ is defined as:

$$\begin{aligned}&\mathtt{Adv}^{(2)}_{\mathcal{A},DSG}(\lambda)\\ &:= \Pr\Big[b \leftarrow \mathcal{A}(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1X_2, X_3, Y_2Y_3, T_b)\Big|(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{comp}(1^\lambda),\\ &g, X_1, Z_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_1}, X_i, Y_i, Z_i \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_i}(i \in \{2,3\}), T_0 := Z_1Z_3, T_1 := Z_1Z_2Z_3, b \overset{\mathrm{U}}{\leftarrow} \{0,1\}\Big].\end{aligned}$$

DSG Assumption 3 [YCZY12]. We consider a problem whose advantage of a PPT adversary $\mathcal{A}$ is defined as:

$$\begin{aligned}&\mathtt{Adv}^{(3)}_{\mathcal{A},DSG}(\lambda)\\ &:= \Pr\Big[b \leftarrow \mathcal{A}(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, g^\alpha X_2, g^s Y_2, Z_2, X_3, T_b)\Big|(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{comp}(1^\lambda),\\ &g \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_1}, X_2, Y_2, Z_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_2}, X_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_3}, \alpha, s \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N, T_0 := \hat{e}(g,g)^{\alpha s}, T_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_T, b \overset{\mathrm{U}}{\leftarrow} \{0,1\}\Big].\end{aligned}$$

Definition 14. *We say that DSG assumption $i \in \{1,2,3\}$ holds, if for every PPT $\mathcal{A}$, his advantage $\mathtt{Adv}^{(i)}_{\mathcal{A},DSG}(\lambda)$ is negligible.*

3.4.3 Definition of Indistinguishability in HL Model of IBE

Yuen et al. use the following IND-ID-CPA game for an IBE scheme $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$. $\mathcal{F}(g^u(k_u))$ denotes a function class which consists of functions computable in polynomial time, where k_u denotes the minimum entropy of the secret user-key, and $g^u(k_u)$ denotes a negligible function such that $g^u(k_u) > 2^{-k_u}$.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$. $\mathcal{CH}$ sends pk to $\mathcal{A}$.

Query. $\mathcal{A}$ is allowed to use extraction oracle $\mathcal{KEO}$, leakage oracle $\mathcal{LO}$, and update oracle $\mathcal{UO}$ adaptively.

$\mathcal{KEO}(ID, i)$**:** i should be $i \in \mathbb{N}^+$, and $ID \in \mathcal{I}$ should be such that list $\mathcal{L}_{ID}$ for the ID has been generated. $\mathcal{CH}$ finds out the tuple in the form of (sk_{ID}, ID, j) in $\mathcal{L}_{ID}$ whose third element is the biggest among all tuples in the list. Let $j^\dagger \in \mathbb{N}^+$ denote the biggest integer. If $i \le j^\dagger$, then $\mathcal{CH}$ retrieves the tuple $(sk'_{ID}, ID, i) \in \mathcal{L}_{ID}$, and sends sk'_{ID} to $\mathcal{A}$.

$\mathcal{LO}(f_i, ID)$**:** f_i should be $f_i \in \mathcal{F}_i(g^u(k_u))$, and $ID \in \mathcal{I}$ should be such that list $\mathcal{L}_{ID}$ for the ID has been generated. The index i of f_i indicates that this query is the i-th query to the leakage oracle. The definition of the function class $\mathcal{F}_i(g^u(k_u))$ is described after the definition of the game. $\mathcal{CH}$ computes $f_i(pk, mk, \mathcal{L}_{ID}, ID)$, then sends it to $\mathcal{A}$.

$\mathcal{UO}(ID)$**:** ID should be $ID \in \mathcal{I}$. If the list $\mathcal{L}_{ID}$ for ID has not been generated, $\mathcal{CH}$ generates the list $\mathcal{L}_{ID}$ initialized by $\emptyset$, and sets $j^\dagger := 0$. Otherwise, $\mathcal{CH}$ finds out the tuple in the form of (sk_{ID}, ID, j) in $\mathcal{L}_{ID}$ whose third element is the biggest integer among all tuples in the list, and sets $j^\dagger$ to the biggest integer.

$\mathcal{CH}$ runs $sk'_{ID} \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, and sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{(sk'_{ID}, ID, j^\dagger + 1)\}$.

Challenge(M_0, M_1, ID^*)**.** $\mathcal{A}$ sends two plaintexts M_0, M_1 and an ID ID^*. $\mathcal{CH}$ sets $b \overset{U}{\leftarrow} \{0, 1\}$ and calculates $C^* \leftarrow \mathsf{Enc}(pk, M_b, ID^*)$. $\mathcal{CH}$ sends C^* to $\mathcal{A}$.

Query 2. $\mathcal{A}$ is allowed to use the oracle $\mathcal{KEO}$ in the same manner as **Query**.

Guess(b')**.** $\mathcal{A}$ sends a bit $b' \in \{0, 1\}$ to $\mathcal{CH}$ as a guess for b.

Every ID queried to $\mathcal{KEO}$ in **Query** or **Query 2** should not be ID^*. Advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{(g^u(k_u))\text{-}AL\text{-}IND\text{-}ID\text{-}CPA}_{\Pi_{IBE},\mathcal{A}}(\lambda) := |\Pr[b' = b] - 1/2|$.

Definition 15. *We say that an IBE scheme* $\Sigma_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *is* $(g^u(k_u))$*-AL-IND-ID-CPA secure, if for every PPT adversary* $\mathcal{A}$ *in the game of* $(g^u(k_u))$*-AL-IND-ID-CPA for* Π_{IBE}*,* $\mathcal{A}$*'s advantage* $\mathsf{Adv}^{(g^u(k_u))\text{-}AL\text{-}IND\text{-}ID\text{-}CPA}_{\Pi,\mathcal{A}}(\lambda)$ *is negligibly small.*

The definition of the function class $\mathcal{F}_i(g^u(k_u))$, where $i \in [1, q_l]$, is as follows. $\mathcal{F}_i(g^u(k_u))$ is a set consists of every function f_i which is computable in polynomial time and satisfies the following condition: for every PPT $\mathcal{B}$, it holds that

$$\Pr\Big[\mathcal{B}\Big(pk, ID^*, S, \{f_j(pk, mk, \mathcal{L}_{ID_j,j}, ID_j)\}_{j\in[1,i]}\Big) \to sk_{ID^*} \text{ such that } sk_{ID^*} \in S^*\Big] \le g^u(k_u).$$

Here, S is the set of all secret keys which has been extracted until the i-th leakage oracle query (f_i, ID_i) is issued, S^* is the set of all secret keys for ID^*, and f_j and ID_j, where $j \in [1, i-1]$, are the function and ID in the j-th leakage oracle query (f_j, ID_j) respectively. Every element or function of pk, mk, S, S^* and $\{f_j(pk, mk, \mathcal{L}_{ID_j,j}, ID_j)\}_{j\in[1,i]})$ is randomly generated or computed.

3.4.4 Concrete IBE Construction Secure in HL Model

Each algorithm of the IBE construction $\Pi^{YCZY12}_{IBE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}\}$ proposed by Yuen et al. is described below.

Setup(1^λ): Run $(N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}) \leftarrow \mathcal{G}_{comp}(1^\lambda)$, where $\mathcal{G}_{comp}$ is a generator of bilinear groups of composite order. Set $g_1, u, h \overset{U}{\leftarrow} \mathbb{G}_{p_1}$, $X_3 \overset{U}{\leftarrow} \mathbb{G}_{p_3}$, and $m := (3\lambda)^{1/\epsilon}$, where $0 < \epsilon < 1$. For every $i \in [1, m]$, set $\alpha_i, t_i \overset{U}{\leftarrow} \mathbb{Z}_N$, $v_i \overset{U}{\leftarrow} \mathbb{G}_{p_1}$, $T_{1,i}, T_{2,i}, T_{3,i} \overset{U}{\leftarrow} \mathbb{G}_{p_3}$, $K_{1,i} := g_1^{\alpha_i} \cdot h^{t_i} \cdot T_{1,i}$, $K_{2,i} := u^{t_i} \cdot T_{2,i}$, and $K_{3,i} := v_i^{t_i} \cdot T_{3,i}$. `Return` (pk, mk), where $pk := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i := \hat{e}(g_1, v_i)^{\alpha_i}\}_{i\in[1,m]})$ and $mk := (\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in[1,m]})$. ID space (resp. plaintext space) is $\mathcal{I} = \mathbb{Z}_N$ (resp. $\mathcal{M} = \mathbb{G}_T$).

KeyGen(pk, mk, ID): For every $i \in [1, m]$, set $r_i \overset{U}{\leftarrow} \mathbb{Z}_N$ and $R_{1,i}, R_{2,i} \overset{U}{\leftarrow} \mathbb{G}_{p_3}$. `Return` $sk_{ID} := (\{D_i, E_i\}_{i\in[1,m]})$, where $D_i := K_{1,i} \cdot K_{2,i}^{ID} \cdot (u^{ID} \cdot h)^{r_i} \cdot R_{1,i}$, and $E_i := K_{3,i} \cdot v_i^{r_i} \cdot R_{2,i}$.

Enc(pk, M, ID): For every $i \in [1, m]$, set $s_i \overset{U}{\leftarrow} \mathbb{Z}_N$. `Return` $C := (A, \{B_i, C_i\}_{i\in[1,m]})$, where $A := M \cdot \prod_{i\in[1,m]} y_i^{s_i}$, $B := v_i^{s_i}$, and $C_i := (u^{ID} \cdot h)^{s_i}$.

Dec(pk, C, ID): `Return` $A \cdot \prod_{i\in[1,m]} \hat{e}(C_i, E_i) / \prod_{i\in[1,m]} \hat{e}(B_i, D_i)$.

3.4.5 Semi-functional Secret-Keys/Ciphertexts

Under the dual system encryption framework, Yuen et al. define master-keys, secret user-keys and ciphertexts in semi-functional (SF) form, each one of which is perturbed by an element in $\mathbb{G}_{p_2}$. Below, each one of $\bar{g}_2$ and $\hat{g}_2$ denotes a randomly generated generator of $\mathbb{G}_{p_2}$.

Given a normal master-key $mk = (\{K_{1,i}, K_{2,i}, K_{3,i}\}_{i\in[1,m]})$, an SF master-key mk' is computed as follows: $mk' := (\{K'_{1,i}, K'_{2,i}, K'_{3,i}\}_{i\in[1,m]}) := (\{K_{1,i} \cdot \bar{g}_2^{\theta_i}, K_{2,i} \cdot \bar{g}_2^{\tau\theta_i}, K_{3,i} \cdot \bar{g}_2^{w_i}\}_{i\in[1,m]})$, where $\theta_1, \cdots, \theta_m \overset{U}{\leftarrow} [1, \lambda]$, and $w_1, \cdots, w_m, \tau \overset{U}{\leftarrow} \mathbb{Z}_N$.

Given a normal secret user-key $sk = (\{D_i, E_i\}_{i\in[1,m]})$, an SF secret user-key sk' is computed as follows: $sk' := (\{C'_i, E'_i\}_{i\in[1,m]}) := (\{C_i \cdot \bar{g}_2^{\kappa_i}, E_i \cdot \bar{g}_2^{z_i}\})$, where $\kappa_1, z_1, \cdots, \kappa_m, z_m \overset{U}{\leftarrow} \mathbb{Z}_N$.

Given a normal ciphertext $C = (A, \{B_i, C_i\}_{i\in[1,m]})$, an SF ciphertext C' is computed as follows: $C' := (A', \{B'_i, C'_i\}) := (A, \{B_i \cdot \hat{g}_2^{\delta_i}, C_i \cdot \hat{g}_2^{x_i}\})$, where $\delta_1, x_i, \cdots, \delta_m, x_m \overset{U}{\leftarrow} \mathbb{Z}_N$.

Decrypting a normal ciphertext for an ID by using an SF secret user-key for the same ID always succeeds. Likewise, decrypting an SF ciphertext for an ID by using a normal secret user-key for the same ID always succeeds. On the other hand, decrypting an SF ciphertext for ID by using an SF secret user-key for the same ID almost always fails, because the decryption results in a blinded message by a factor $\hat{e}(\bar{g}_2, \hat{g}_2)^{\Sigma_{i=1}^m\{z_ix_i - \kappa_i\delta_i\}}$. We say that an SF secret user-key $sk' = (\{D'_i, E'_i\}_{i\in[1,m]})$ for ID is nominally semi-functional (NSF) for an SF ciphertext $C' = (A', \{B'_i, C'_i\})$ for the same ID, or that an SF ciphertext C' for ID is NSF for an SF secret user-key sk' for the same ID, if it holds that $\Sigma_{i=1}^m\{z_ix_i - \kappa_i\delta_i\} \equiv 0 \pmod{p_2}$. If an SF secret user-key (resp. ciphertext) is not NSF, we say that the key (resp. ciphertext) is truly semi-functional (TSF).

3.4.6 Sketch of Proof of Indistinguishability in HL Model

Yuen et al. claim that security of their IBE scheme is guaranteed by the following theorem.

Theorem 3.4.1. *([YCZY12]) IBE scheme Π_{IBE}^{YCZY12} is $(2^{-m^{\epsilon}})$-AL-ID-CPA secure under the DSG assumptions 1, 2, 3 given in Subsect. 3.4.2.*

Proof-Sketch of Theorem 3.4.1. The definition of the games whom Yuen et al. use to prove Theorem 3.4.1 is as follows:

- $\mathtt{Game}_{real}$ is the normal $(2^{-m^{\epsilon}})$-AL-IND-ID-CPA game for Π_{IBE}^{YCZY12} in which an adversary $\mathcal{A}$ plays with a challenger $\mathcal{CH}$. Let q_e, q_l and q_u denote the number of times which $\mathcal{A}$ uses $\mathcal{KEO}$, $\mathcal{LO}$ and $\mathcal{UO}$, respectively. q denotes $q = q_e + q_l + q_u$.
- $\mathtt{Game}_{restricted}$ is the same as $\mathtt{Game}_{real}$ except that every ID queried to the extraction oracle satisfies $ID \neq ID^* \mod p_2$.
- $\mathtt{Game}_0$ is the same as $\mathtt{Game}_{restricted}$ except for the following respects. Firstly, a generator $\hat{g}_2$ of $\mathbb{G}_{p_2}$ is randomly generated in **Setup**. Secondly, the challenge ciphertext is generated in semi-functional form.
- $\mathtt{Game}_i$, where $i \in [1, q]$, is the same as $\mathtt{Game}_0$ except for the following respects. Firstly, a generator $\bar{g}_2$ of $\mathbb{G}_{p_2}$ is randomly generated in **Setup**. Secondly, for every $j \in [1, i]$, the reply against the j-th oracle query is computed by using secret-keys in semi-functional form.
- $\mathtt{Game}_{final}$ is the same as $\mathtt{Game}_q$ except that the challenge ciphertext is generated as a (semi-functional) ciphertext of a uniformly chosen plaintext $M' \overset{\mathrm{U}}{\leftarrow} \mathcal{M}$.

$\mathtt{Adv}_X$, where $X \in \{real, restricted, 0, 1, \cdots, q, final\}$, denotes the advantage of $\mathcal{A}$ in $\mathtt{Game}_X$. For the advantage of $\mathcal{A}$ in the game of $(2^{-m^{\epsilon}})$-AL-IND-ID-CPA for the IBE scheme Π_{IBE}^{YCZY12}, it holds that

$$\mathtt{Adv}^{(2^{-m^{\epsilon}})\text{-}AL\text{-}IND\text{-}ID\text{-}CPA}_{\Pi_{IBE}^{YCZY12},\mathcal{A}}(\lambda) = \mathtt{Adv}_{real} \leq |\mathtt{Adv}_{real} - \mathtt{Adv}_{restricted}|$$

$$+ |\mathtt{Adv}_{restricted} - \mathtt{Adv}_0| + \sum_{i=1}^{q} |\mathtt{Adv}_{i-1} - \mathtt{Adv}_i| + \left|\mathtt{Adv}_q - \mathtt{Adv}_{final}\right| + \mathtt{Adv}_{final}.$$

Theorem 3.4.1 is proven by all of the following lemmas whose proofs are omitted in [YCZY12].

Lemma 3.4.1. *If the DSG assumption 2 holds, then $|\mathtt{Adv}_{real} - \mathtt{Adv}_{restricted}|$ is negligible.*

Lemma 3.4.2. *If the DSG assumption 1 holds, then $|\mathtt{Adv}_{restricted} - \mathtt{Adv}_0|$ is negligible.*

Lemma 3.4.3. *If the DSG assumption 2 holds, then for any $i \in [1, q]$, $|\mathtt{Adv}_{i-1} - \mathtt{Adv}_i|$ is negligible.*

Lemma 3.4.4. *If the DSG assumption 3 holds, then $\left|\mathtt{Adv}_q - \mathtt{Adv}_{final}\right|$ is negligible.*

$f_1(pk, mk, \mathcal{L}_{ID^*}, ID^*)$: $\mathcal{L}_{ID^*}$ is parsed as $\{(sk_{ID^*}, ID^*, 1)\}$, sk_{ID^*} is parsed as $(\{D_i, E_i\}_{i\in[1,m]})$, $\mathtt{Return}\ (D_1, E_1)$
$f_2(pk, mk, \mathcal{L}_{ID^*}, ID^*)$: pk is parsed as $pk := (N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i\}_{i\in[1,m]})$, $\mathcal{L}_{ID^*}$ is parsed as $\{(sk_{ID^*}, ID^*, 1)\}$ sk_{ID^*} is parsed as $(\{D_i, E_i\}_{i\in[1,m]})$, $r'_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N$, $D'_1 := D_1 \cdot (u^{ID} \cdot h)^{r'_1}$, $E'_1 := E_1 \cdot v^{r'_1}$ $\mathtt{Return}\ (D'_1, E'_1)$
$f_3(pk, mk, \mathcal{L}_{ID^*}, ID^*)$: $\mathcal{L}_{ID^*}$ is parsed as $\{(sk_{ID^*}, ID^*, 1), (sk'_{ID^*}, ID^*, 2)\}$, sk'_{ID^*} is parsed as $(\{D'_i, E'_i\}_{i\in[1,m]})$ $\mathtt{Return}\ (D'_1, E'_1)$
$f_4(pk, mk, \mathcal{L}_{ID^*}, ID^*)$: $sk'_{ID^*} \leftarrow \mathsf{KeyGen}(pk, mk, ID^*)$, sk'_{ID^*} is parsed as $(\{D'_i, E'_i\}_{i\in[1,m]})$, $\mathtt{Return}\ (D'_1, E'_1)$

Figure 3.1: Functions f_1, f_2, f_3 and f_4.

3.5 Counterexamples against the Security Proof in [YCZY12]

We have not found any cryptanalysis of their scheme. However, we have found some counterexamples of adversaries, each one of which indicates that their proof of Theorem 3.4.1 is defective. More precisely, they indicate that Lemma 3.4.3, which originally is Lemma 9 in [YCZY12], is false.

A Counterexample. In this paragraph, we give the details about one of the counterexamples. In the next paragraph, we describe the other ones of them.

One of the counterexamples, denoted by $\mathcal{A}_1$, behaves in $\mathtt{Game}_{real}$ as follows.

1. In $\mathtt{Setup}$, $\mathcal{A}_1$ receives pk which is parsed as $(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i\}_{i\in[1,m]})$.

2. In $\mathtt{Query}$, $\mathcal{A}_1$ queries ID^* to $\mathcal{UO}$, where $ID^* \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N$. After that, $\mathcal{A}_1$ queries ID^* and a function f_1 in Fig. 3.1 to $\mathcal{LO}$, then receives $(D_1, E_1) \in \mathbb{G}^2$. After that, $\mathcal{A}_1$ queries ID^* and a function f_2 in Fig.3.1 to $\mathcal{LO}$, then receives $(D'_1, E'_1) \in \mathbb{G}^2$.

3. In $\mathtt{Challenge}$, $\mathcal{A}_1$ queries two plaintexts $M_0, M_1 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_T$ and ID^*, and receives a challenge ciphertext $C^* = (A^*, \{B^*_i, C^*_i\}_{i\in[1,m]})$.

4. In $\mathtt{Guess}$, $\mathcal{A}_1$ calculates $K_1 := \hat{e}(C^*_1, E_1)$, $K_2 := \hat{e}(B^*_1, D_1)$, $K'_1 := \hat{e}(C^*_1, E'_1)$, and $K'_2 := \hat{e}(B^*_1, D'_1)$. $\mathcal{A}_1$ outputs 1 if $K_2/K_1 = K'_2/K'_1$, and outputs 0 otherwise.

As is obvious from $\mathcal{A}_1$'s behavior, $\mathcal{A}_1$ does not try to guess the challenge bit b correctly. However, such a behavior is allowed and there can be a PPT adversary which behaves like $\mathcal{A}_1$. In [YCZY12], the authors insist that every one of the 4 lemmas holds for any PPT adversary which can be $\mathcal{A}_1$. For the case of $\mathcal{A}_1$, we need to consider 6 concrete games $\mathtt{Game}_{real}$, $\mathtt{Game}_{restricted}$, $\mathtt{Game}_1$, $\mathtt{Game}_2$, $\mathtt{Game}_3$ and $\mathtt{Game}_{final}$. The existence of $\mathcal{A}_1$ indicates that Lemma 3.4.3 is false since it states that $|\mathtt{Adv}_1 - \mathtt{Adv}_2|$ is negligible because of the DSG assumption 2. The details are written below.

Although the real proof of Lemma 3.4.3 is not written in [YCZY12] and has not been disclosed, it must proceed as follows. A PPT simulator $\mathcal{S}$ which tries to break the DSG assumption 2 receives an instance which is denoted by $(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g, X_1X_2, X_3, Y_2Y_3, T_b)$. $\mathcal{S}$ perfectly simulates one of $\mathtt{Game}_1$ and $\mathtt{Game}_2$ when $b = 0$, and the other one when $b = 1$, to a PPT adversary $\mathcal{A}$ which can be $\mathcal{A}_1$. As a result, we get a fact that distinguishing $\mathtt{Game}_1$ and $\mathtt{Game}_2$ is at least as hard as distinguishing T_0 and T_1 in the DSG assumption 2.

If the proof is done as above, it implies that any PPT adversary can not distinguish $\mathtt{Game}_1$ from $\mathtt{Game}_2$ with a non-negligible advantage, because if such an adversary exists, the simulator can distinguish T_0 from T_1 with a non-negligible advantage by using the adversary. Let us explain below that $\mathcal{A}_1$ can distinguish $\mathtt{Game}_1$ from $\mathtt{Game}_2$ with a non-negligible advantage. Precisely, $\mathcal{A}_1$ which tries to distinguish $\mathtt{Game}_1$ from $\mathtt{Game}_2$, communicates with $\mathcal{CH}$ as follows.

Setup: $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda)$. pk and mk are parsed as $pk = (N = p_1p_2p_3, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i := \hat{e}(g_1, v_i)^{\alpha_i}\}_{i\in[1,m]})$ and $mk = (\{K_{1,i}, K_{2,i}, K_{3,i}\})$, where $K_{1,i} := g_1^{\alpha_i} \cdot h^{t_i} \cdot T_{1,i}$, $K_{2,i} := u^{t_i} \cdot T_{2,i}$, and $K_{3,i} := v_i^{t_i} \cdot T_{3,i}$, respectively. $\mathcal{CH}$ sends pk to $\mathcal{A}_1$. Generators of $\mathbb{G}_{p_2}$-part of semi-functional secret key and challenge ciphertext are randomly chosen, and they are denoted by $\bar{g}_2$ and $\hat{g}_2$ respectively.

Query: If $\mathcal{A}_1$ issues the following three queries sequentially, $\mathcal{CH}$ acts as follows.

$\mathcal{UO}(ID^*)$**:** For every $i \in [1, m]$, after $\mathcal{CH}$ sets $r_i, \kappa_i, z_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N$ and $R_{1,i}, R_{2,i} \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_{p_3}$, $\mathcal{CH}$ sets D_i and E_i as follows, respectively:

$$\begin{aligned} D_i &:= K_{1,i} \cdot K_{2,i}^{ID^*} \cdot (u^{ID^*} \cdot h)^{r_i} \cdot R_{1,i} \cdot \bar{g}_2^{\kappa_i} = g_1^{\alpha_i} \cdot (u^{ID^*} \cdot h)^{r_i+t_i} \cdot T_{1,i} \cdot T_{2,i} \cdot R_{1,i} \cdot \bar{g}_2^{\kappa_i} \\ E_i &:= K_{3,i} \cdot v_i^{r_i} \cdot R_{2,i} = v_i^{r_i+t_i} \cdot T_{3,i} \cdot R_{2,i} \cdot \bar{g}_2^{z_i}. \end{aligned}$$

After that if the game is $\mathtt{Game}_2$, $\mathcal{CH}$ sets $sk_{ID^*} := (\{D_i, E_i\}_{i\in[1,m]})$, and if the game is $\mathtt{Game}_1$, $\mathcal{CH}$ sets $sk_{ID^*} := (\{D_i/\bar{g}_2^{\kappa_i}, E_i/\bar{g}_2^{z_i}\}_{i\in[1,m]})$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{ID^*} := \{(sk_{ID^*}, ID^*, 1)\}$.

$\mathcal{LO}(f_1, ID^*)$**:** $\mathcal{CH}$ calculates $f_1(pk, mk, \mathcal{L}_{ID^*}, ID^*)$, and returns it to $\mathcal{A}_1$. After that, in $\mathtt{Game}_2$, the tuple $(sk_{ID^*}, ID^*, 1) \in \mathcal{L}_{ID^*}$ is retrieved, and the secret-key $sk_{ID^*} = (\{D_i, E_i\}_{i\in[1,m]})$ is changed to $sk'_{ID^*} := (\{D_i/\bar{g}_2^{\kappa_i}, E_i/\bar{g}_2^{z_i}\}_{i\in[1,m]})$.

$\mathcal{LO}(f_2, ID^*)$**:** $\mathcal{CH}$ calculates $f_2(pk, mk, \mathcal{L}_{ID^*}, ID^*)$, and returns it to $\mathcal{A}_1$.

Challenge(M_0, M_1, ID^*)**:** $\mathcal{CH}$ sets $b \overset{\mathrm{U}}{\leftarrow} \{0, 1\}$. For every $i \in [1, m]$, after setting $s_i, \delta_i, x_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N$, $\mathcal{CH}$ sets A^*, B_i^* and C_i^* as follows:

$$A^* := M_b \cdot \prod_{i\in[1,m]} y_i^{s_i}, \quad B_i^* := v_i^{s_i} \cdot \hat{g}_2^{\delta_i}, \quad C_i^* := (u^{ID^*} \cdot h)^{s_i} \cdot \hat{g}_2^{x_i}.$$

After that, $\mathcal{CH}$ sends $C^* = (A^*, \{B_i^*, C_i^*\}_{i\in[1,m]})$ to $\mathcal{A}_1$.

Guess(b')**:** $\mathcal{CH}$ receives b' from $\mathcal{A}_1$, where b' is set as 1 iff $K_2/K_1 = K_2'/K_1'$.

In $\mathtt{Game}_1$, K_1, K_2, K_1' and K_2' are written as follows.

$$\begin{aligned}
K_1 &= \hat{e}(C_1^*, E_1) = \hat{e}((u^{ID^*} \cdot h)^{s_1} \cdot \hat{g}_2^{x_1}, v_1^{r_1+t_1} \cdot T_{3,1} \cdot R_{2,1}) = \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+t_1)} \\
K_2 &= \hat{e}(B_1^*, D_1) = \hat{e}(v_1^{s_1} \cdot \hat{g}_2^{\delta_1}, g_1^{\alpha_1} \cdot (u^{ID^*} \cdot h)^{r_1+t_1} \cdot T_{1,1} \cdot T_{2,1} \cdot R_{1,1}) \\
&= \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+t_1)} \cdot \hat{e}(v_1, g_1)^{s_1\alpha_1} \\
K_1' &= \hat{e}(C_1^*, E_1') = \hat{e}((u^{ID^*} \cdot h)^{s_1} \cdot \hat{g}_2^{x_1}, v_1^{r_1+r_1'+t_1} \cdot T_{3,1} \cdot R_{2,1} \cdot \bar{g}_2^{z_1}) = \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+r_1'+t_1)} \\
K_2' &= \hat{e}(B_1^*, D_1') = \hat{e}(v_1^{s_1} \cdot \hat{g}_2^{\delta_1}, g_1^{\alpha_1} \cdot (u^{ID^*} \cdot h)^{r_1+r_1'+t_1} \cdot T_{1,1} \cdot T_{2,1} \cdot R_{1,1} \cdot \bar{g}_2^{\kappa_1}) \\
&= \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+r_1'+t_1)} \cdot \hat{e}(v_1, g_1)^{s_1\alpha_1}
\end{aligned}$$

Because $K_2/K_1 = K_2'/K_1' = \hat{e}(v_1, g_1)^{s_1\alpha_1}$, $\mathcal{A}_1$ outputs $b' = 1$ with probability 1.

In $\mathtt{Game}_2$, K_1' and K_2' are written in the same form as $\mathtt{Game}_1$. K_1 and K_2 are written as follows.

$$\begin{aligned}
K_1 &= \hat{e}(C_1^*, E_1) = \hat{e}((u^{ID^*} \cdot h)^{s_1} \cdot \hat{g}_2^{x_1}, v_1^{r_1+t_1} \cdot T_{3,1} \cdot R_{2,1} \cdot \bar{g}_2^{z_1}) = \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+t_1)} \cdot \hat{e}(\bar{g}_2, \hat{g}_2)^{x_1z_1} \\
K_2 &= \hat{e}(B_1^*, D_1) = \hat{e}(v_1^{s_1} \cdot \hat{g}_2^{\delta_1}, g_1^{\alpha_1} \cdot (u^{ID^*} \cdot h)^{r_1+t_1} \cdot T_{1,1} \cdot T_{2,1} \cdot R_{1,1} \cdot \bar{g}_2^{\kappa_1}) \\
&= \hat{e}(u^{ID^*} \cdot h, v_1)^{s_1(r_1+t_1)} \cdot \hat{e}(v_1, g_1)^{s_1\alpha_1} \cdot \hat{e}(\bar{g}_2, \hat{g}_2)^{\delta_1\kappa_1}
\end{aligned}$$

Hence, K_2/K_1 is as follows.

$$\begin{aligned}
K_2/K_1 &= \hat{e}(v_1, g_1)^{s_1\alpha_1} \cdot \hat{e}(\bar{g}_2, \hat{g}_2)^{\delta_1\kappa_1 - x_1z_1} \\
&= \begin{cases} \hat{e}(v_1, g_1)^{s_1\alpha_1} = K_2'/K_1' & (\delta_1\kappa_1 \equiv x_1z_1 \pmod{p_2}) \\ \hat{e}(v_1, g_1)^{s_1\alpha_1} \cdot \hat{e}(\bar{g}_2, \hat{g}_2)^{\delta_1\kappa_1 - x_1z_1} \neq K_2'/K_1' & (otherwise) \end{cases}
\end{aligned}$$

Each one of κ_1, z_1, δ_1 and x_1 is chosen uniformly at random from $\mathbb{Z}_N$. So, the probability that they satisfy the relation $\delta_1\kappa_1 \equiv x_1z_1 \pmod{p_2}$ is $1/p_2$. Hence, $\mathcal{A}_1$ outputs $b' = 1$ with probability $1/p_2$, and outputs $b' = 0$ with probability $1 - 1/p_2$.

Therefore, advantage of $\mathcal{A}_1$ distinguishing $\mathtt{Game}_1$ from $\mathtt{Game}_2$ becomes the following non-negligible value.

$$\left|\Pr[b' = 1|\mathtt{Game}_1] - \Pr[b' = 1|\mathtt{Game}_2]\right| = |1 - 1/p_2| = 1 - 1/p_2.$$

In the above, we explained how one of our counterexamples, i.e., $\mathcal{A}_1$, effectively works. We have found some other counterexamples. The details of them are given in the next paragraph.

The Other Counterexamples. In the last paragraph, a counterexample $\mathcal{A}_1$ was given. In this paragraph, the other two counterexamples are given.

The second PPT adversary is denoted by $\mathcal{A}_2$ and behaves in $\mathtt{Game}_{real}$ as follows.

1. In **Setup**, $\mathcal{A}_2$ receives pk which is parsed as $(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i\}_{i\in[1,m]})$.

2. In **Query**, $\mathcal{A}_2$ queries ID^* to $\mathcal{UO}$, where $ID^* \overset{\text{U}}{\leftarrow} \mathbb{Z}_N$. After that, $\mathcal{A}_2$ queries ID^* and a function f_1 in Fig. 3.1 to $\mathcal{LO}$, then receives $(D_1, E_1) \in \mathbb{G}^2$. After that, $\mathcal{A}_2$ queries ID^* to $\mathcal{UO}$ again. After that, $\mathcal{A}_2$ queries ID^* and a function f_3 in Fig. 3.1 to $\mathcal{LO}$, then receives $(D_1', E_1') \in \mathbb{G}^2$.

3. In **Challenge**, $\mathcal{A}_2$ queries M_0, M_1 and ID^*, where $M_1, M_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_T$. $\mathcal{A}_2$ receives a challenge ciphertext C^*, where C^* is parsed as $(A^*, \{B_i^*, C_i^*\}_{i\in[1,m]})$.

4. In **Guess**, $\mathcal{A}_2$ calculates $K_1 := \hat{e}(C_1^*, E_1)$, $K_2 := \hat{e}(B_1^*, D_1)$, $K_1' := \hat{e}(C_1^*, E_1')$, and $K_2' := \hat{e}(B_1^*, D_1')$. $\mathcal{A}_2$ outputs 1 if $K_2/K_1 = K_2'/K_1'$, and outputs 0 otherwise.

The third PPT adversary is denoted by $\mathcal{A}_3$ and behaves in $\mathtt{Game}_{real}$ as follows.

1. In **Setup**, $\mathcal{A}_3$ receives pk which is parsed as $(N, \mathbb{G}, \mathbb{G}_T, \hat{e}, g_1, u, h, X_3, \{v_i, y_i\}_{i\in[1,m]})$.

2. In **Query**, $\mathcal{A}_3$ queries ID^* to $\mathcal{UO}$, where $ID^* \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_N$. After that, $\mathcal{A}_3$ queries ID^* and the function f_1 in Fig. 3.1 to $\mathcal{LO}$, then receives $D_1 \in \mathbb{G}, E_1 \in \mathbb{G}$. After that, $\mathcal{A}_3$ queries ID^* and a function f_4 in Fig. 3.1, then receives $D_1' \in \mathbb{G}, E_1' \in \mathbb{G}$.

3. In **Challenge**, $\mathcal{A}_3$ queries M_0, M_1 and ID^*, where $M_1, M_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}_T$. $\mathcal{A}_2$ receives a challenge ciphertext C^*, where C^* is parsed as $(A, \{B_i, C_i\}_{i\in[1,m]})$.

4. In **Guess**, $\mathcal{A}_3$ calculates $K_1 := \hat{e}(C_1, E_1)$, $K_2 := \hat{e}(B_1, D_1)$, $K_1' := \hat{e}(C_1, E_1')$, and $K_2' := \hat{e}(B_1, D_1')$. $\mathcal{A}_2$ outputs 1 if $K_2/K_1 = K_2'/K_1'$, and outputs 0 otherwise.

As $\mathcal{A}_1$, the new PPT adversaries $\mathcal{A}_2$ and $\mathcal{A}_3$ can also distinguish $\mathtt{Game}_1$ from $\mathtt{Game}_2$ with a non-negligible advantage.

3.6 Deficiency of Security Proofs for Some ABE Schemes

We have found some defective parts in the full security proof of the ABE schemes by Zhang et al. [ZWTM13] and Wang et al. [WY15].

Zhang et al. do not use the GL theorem properly in the proof of their Lemma 5.4, so the proof of the lemma is not correct. And, we have found some concrete PPT adversaries as counterexamples which indicate that the proof of their Lemma 5.3 is wrong. The counterexamples behave similarly to ones against the proof of Yuen et al. [YCZY12] introduced in Subsect. 3.5.

Since Wang et al. [WY15] do not use the GL theorem anywhere in their security proof, their scheme lacks the guarantee for its leakage-resilience.

3.7 Conclusion for Chapter 3

In the work introduced in this chapter, we achieved the following contributions.

Firstly, we showed that the IBE scheme [YCZY12] which has been believed to be the only known one whose IND-CPA security in HL model (or AL model) was correctly proven is defective in its security proof. Specifically, we presented some concrete PPT adversaries as counterexamples which indicate the deficiency of the security proof. Note that the counterexamples effectively work to imply the deficiency of security proof for their another IBE scheme which has been believed to be the only known one whose security in *continual auxiliary leakage (CAL) model*[3] was correctly proven. We also showed that only known ABE schemes with HL resilience proposed in [ZWTM13, WY15] includes deficiency in their security proofs.

Secondly, we gave another definition of IND-CPA security considering HL resilience distinct from the one in [YCZY12]. We suitably modified the IBE scheme proposed by Kurosawa and Phong [KP13] which has been proven to be secure in BL model. Then, we proved that our IBE scheme is secure in our HL model under the DLIN assumption in the standard model. As a result, considering the first contribution given above, we can say that our IBE scheme is the first one whose HL resilience was correctly proven.

Related to this work, we can present various open problems.

For instance, one of them is presenting an IBE scheme secure in an *improved* security model. As we mentioned in Sect. 3.2, adversaries considered in our security model are weak in the following respects. Firstly, we do not allow them to use the key-revelation oracle and the leakage oracle simultaneously and adaptively. Secondly, we do not allow *multiple* leakage (from a single key for the target ID ID^*). Also, we do not allow (single) leakage from multiple keys for ID^*. Improving our model in such respects and presenting an IBE scheme secure in the model can be an open problem.

The other open problems include presenting the first ABE or more functional encryption scheme secure in HL model, and presenting an IBE scheme secure in HL model under assumptions other than the DLIN assumption.

[3]Continual auxiliary leakage model [YCZY12] is a combination model of CL model and AL model (or HL model).

Chapter 4

Digital Signatures with Hard-to-Invert Leakage-Resilience

4.1 Introduction for Chapter 4

4.1.1 Background

We say that a signature scheme is weakly (existentially) unforgeable if it is hard to forge a signature on a message not signed before. We say that a signature scheme is strongly (existentially) unforgeable if it is hard to forge a signature on any message which can be a message signed before. Formal definitions of each security notion in non-leakage setting was given in Sect. 2.9. Since most of the signature schemes generate a signature randomly, there is a theoretical/practical gap between the weak unforgeability and strong unforgeability. Moreover, in some applications, strongly unforgeable signature scheme is required, e.g., Canetti-Halevi-Katz transformation [CHK04].

4.1.2 Related Work

Katz and Vaikuntanathan [KV09] defined that fully leakage-resilient (FLR) signature is a signature resilient to not only the direct leakage from the secret-key, but also the leakage from the randomnesses used to generate the secret-key and signatures generated by the signing oracle. FLR or non-FLR signature schemes secure in the bounded-leakage model have been proposed in [ADW09, KV09, MTVY11, BSW11] and the others.

The concept of the hard-to-invert leakage-resilience was presented by Dodis et al. [DKL09]. They proposed symmetric-key encryption schemes which is IND-CPA or IND-CCA secure and resilient to exponentially hard-to-invert leakage. In a subsequent work, Dodis et al. [DGK+10] started a research on public-key encryption with hard-to-invert leakage-resilience. They defined two leakage-function classes. The class $\mathcal{H}_{\mathsf{ow}}(\xi(\lambda))$ (resp. $\mathcal{H}_{\mathsf{pkow}}(\xi(\lambda))$) consists of every polynomial-time computable function $f : \{0,1\}^{|pk|+|sk|} \to \{0,1\}^*$ such that any PPT algorithm $\mathcal{A}$ which is given $f(pk, sk)$ (resp. $(pk, f(pk, sk))$) as input is able to guess sk correctly only with a probability smaller than $\xi(\lambda)$, where $\xi(\lambda) > 2^{-k}$ is a negligible function and (pk, sk) is a randomly generated key-pair. They proved that the BHHO encryption scheme [BHHO08] and a slightly modified version of the GPV encryption scheme [GPV08] are IND-CPA secure in HL model w.r.t. the function class $\mathcal{H}_{\mathsf{ow}}(1/\mu_1(\lambda))$, where $\mu_1(\lambda)$ is a sub-exponential function. They also mentioned in Subsect. 1.2 of [DGK$^+$10] that a PKE scheme which is IND-CPA secure in HL model w.r.t. $\mathcal{H}_{\mathsf{pkow}}(1/\mu_2(\lambda))$, where $\mu_2(\lambda)$ is a polynomial function, is given in its full paper.

Faust et al. presented the first research result on digital signature with hard-to-invert leakage-resilience [FHN$^+$12]. To construct a signature scheme secure in HL model, there is an obstacle whom we have to overcome. It is how to prevent from an adversary choosing an algorithm generating a valid signature on a message as a leakage-function, then output the pair of signature and message. Faust et al. proposed a signature scheme which is wEUF-CMA (weakly existentially unforgeability under adaptively chosen messages attack) secure in HL model w.r.t. the function class $\mathcal{H}_{\mathsf{pkow}}(1/\mu_3(\lambda))$, where $\mu_3(\lambda)$ is an exponential function. Their solution to overcome the obstacle explained earlier is to include a ciphertext of the secret-key sk in a signature. Specifically, their signature scheme adopts the labeled public-key encryption (LPKE) as a building block, and includes a ciphertext of the secret-key in a signature. Moreover, for their signature scheme, the hardness parameter $1/\mu_3(\lambda)$ in the leakage function class is set as $1/\mu_3(\lambda) << 2^{-l_{dk}}$, where $l_{dk} \in \mathbb{N}$ denotes the bit-length of the decryption-key dk of the LPKE scheme. This solution effectively works. The reason is as follows. Since any PPT algorithm is able to guess the decryption-key dk correctly with probability $2^{-l_{dk}}$, any PPT inverter in the definition of the function class $\mathcal{H}_{\mathsf{pkow}}(1/\mu_3(\lambda))$ which is given a signature including a ciphertext C of the secret-key sk is able to guess sk correctly with probability $2^{-l_{dk}} >> 1/\mu_3(\lambda)$ by guessing the decryption-key dk, then decrypting the ciphertext C with the guessed dk. Hence, the signing algorithm is excluded from the class $\mathcal{H}_{\mathsf{pkow}}(1/\mu_3(\lambda))$. By the way, they showed that their signature scheme can be instantiated under standard assumptions such as the DLIN assumption [BBS04].

Independently of Faust et al. [FHN$^+$12], Yuen et al. [YYH12] also presented a research result on signature secure in HL model. To overcome the obstacle to construct a signature with hard-to-invert leakage resilience, Yuen et al. proposed an original security model, which is named *selective auxiliary input model*. In the security model, the adversary is allowed to choose as the leakage-functions only functions which are independent of the public-key. They proposed a signature scheme secure in the security model. Their signature scheme is FLR and resilient to polynomially hard-to-invert leakage. Here, their definition of leakage function f being resilient to polynomially hard-to-invert leakage is as follows: any PPT algorithm which is given $(pk, S, f(state))$ is able to guess sk correctly only with a negligible probability, where (pk, sk) is a randomly generated key-pair, S is a set of randomly generated signatures on the messages queried to the signing oracle, and $state$ is a set of randomnesses used to generate sk and the signatures S. Their definition of leakage-function is undesirable since it depends on the messages whose signatures are generated on the signing oracle.

Subsequently, Wang et al. [WMHT16] proposed a signature scheme secure in the selective auxiliary input model. Their signature scheme is FLR and resilient to polynomially hard-to-invert leakage. Their definition for a function to be resilient to polynomially hard-to-invert leakage is not the same as the one by Yuen et al. [YYH12]. It is improved as follows: any PPT algorithm which is given $f(sk)$ is able to identify sk only with a negligible probability. However, their scheme needs differing input obfuscator (diO), indistinguishable obfuscator (iO), and point-function obfuscator with auxiliary input (AIPO), each one of which has been constructed only under strong assumptions.

Note that each one of the signature schemes with auxiliary leakage resilience by Faust et al., Yuen et al., and Wang et al., is not strongly existentially unforgeable, but weakly existentially unforgeable.

Boneh et al. [BSW06] proposed a method to transform a weakly unforgeable signature scheme into a strongly unforgeable one. However, their transformation can be applied to *partitioned* signatures only. In a subsequent work, Steinfeld et al. [SPW07] proposed a method to transform *any* weakly unforgeable signature into a strongly unforgeable one. Note that each transformation by Boneh et al. and Steinfeld et al. has a common property such that each one of the public-key, secret-key and signature of the strongly unforgeable signature scheme becomes each one of the public-key, secret-key and signature of the weakly unforgeable signature whom some new elements are added to. Huang et al. [HWZ07] proposed another transformation where no new elements are added to the public-key, secret-key and signature.

Wang et al. [WT14] modified the transformation by Steinfeld et al. [SPW07] to get a transformation from a signature weakly existentially unforgeable and FLR in the bounded leakage model to a strongly unforgeable one. The transformation by Steinfeld et al. utilizes two chameleon hash functions (with no leakage-resilience). In the transformation by Wang et al., one of the chameleon hash functions is assumed to satisfy a property such that any PPT algorithm cannot find a strong collision even if the algorithm is given a length-bounded information about the secret-key.

The transformation by Wang et al. needs to add some new elements to both the key-pair and signature. Huang et al. [HHP16] modifies the transformation by Huang et al. [HWZ07] to get a method to transform a signature weakly existentially unforgeable and FLR in BL model into a strongly unforgeable one where no elements are added to the signature[1].

4.1.3 Our Result

We propose a generic construction of signature scheme strongly which is unforgeable and resilient to polynomially hard-to-invert leakage and can be instantiated under standard assumptions. Specifically, we give an example of its instantiation under the decisional linear (DLIN) assumption [BBS04]. Our security model is not the selective auxiliary leakage model [YYH12], so the leakage-function can be dependent on the public-key.

Our result is a desirable one because of the following two independent points. Firstly, our signature instantiation is the first one which is resilient to polynomially hard-to-invert leakage under standard assumptions. Secondly, our signature instantiation is the first one which is strongly unforgeable and has hard-to-invert leakage-resilience.

[1]By the transformation in [HHP16], some new elements are added to the public-key and secret-key.

4.1.4 Our Approach

Our result is obtained by modifying the one by Faust et al. [FHN+12]. Before explaining how the modification is done, we explain the result by Faust et al. in detail.

Faust et al. proposed a generic construction of a signature scheme secure in the wEUF-CMA security model w.r.t. the function class $\mathcal{H}_{\mathtt{pkow}}(\xi(\lambda))$. It consists of three building blocks. They are second pre-image resistant hash function (SPRHF), labeled PKE (LPKE) whose decryption-key dk has bit-size $l_{dk} \in \mathbb{N}$, and non-interactive zero-knowledge proof (NIZK) whose trapdoor td has bit-size $l_{td} \in \mathbb{N}$. The hardness parameter of the leakage function class is set as $\xi(\lambda) = 2^{-(\lambda+l_{dk}+l_{td})}$. A signature σ on a message m consists of an LPKE ciphertext c and an NIZK proof π. Concretely, the ciphertext c is an LPKE ciphertext encrypting the secret-key sk under the label m, and the proof π is an NIZK proof which proves that there exists a secret-key sk' such that the ciphertext c is a ciphertext of sk' on the label m and the hashed value of sk' is equivalent to the hashed value of the real secret-key sk which is included in the public-key pk.

Intuitively speaking, the security proof for the signature by Faust et al. is done as follows. By modifying the initial security game several times, we get the final game $\mathtt{Game}_{final}$. In $\mathtt{Game}_{final}$, for a signature $\sigma = (c, \pi)$ on the signing oracle, the ciphertext c is generated by encrypting $0^{|sk|}$ instead of sk, and the proof π is generated by using the trapdoor td instead of sk. In addition, the adversary is considered to win the game, if he successfully outputs a signature $\sigma^* = (c^*, \pi^*)$ and a message m^* such that c^* is a valid ciphertext of sk^* on label m^*, and π^* is a valid proof. We prove that every PPT $\mathcal{A}$ wins the game only with a negligible probability by a reduction to the hard-to-invert property of the leakage-function $f \in \mathcal{H}_{\mathtt{pkow}}(2^{-(\lambda+l_{dk}+l_{td})})$. In the reduction, a simulator $\mathcal{S}$ needs both td and dk to simulate $\mathtt{Game}_{final}$ and decrypt the ciphertext c^*. However, by the definition of the leakage function class $\mathcal{H}_{\mathtt{pkow}}(\cdot)$, $\mathcal{S}$ is given neither td nor dk, so $\mathcal{S}$ has to guess them, and the guess succeeds with probability $2^{-(l_{dk}+l_{td})}$. By the above reason, the hardness parameter for Faust et al.'s signature scheme becomes $2^{-(\lambda+l_{dk}+l_{td})}$.

The above is the result by Faust et al. We modify the result with three steps.

In the first step, we generalize the second pre-image resistance (SPR) property of the SPRHF, which is one of the building blocks. Intuitively, the SPR property is a property such that no PPTA given a key-pair (pk, sk) is able to find a secret-key sk^* which is not sk, but has a hashed value equivalent to the hashed value of sk with a non-negligible probability. We generalize it to a property such that no PPTA given (pk, sk) is able to find a secret-key sk^* such that a relation holds between sk^* and sk and another relation also holds between sk^* and pk with a non-negligible probability.

The second step is to modify the definition of the leakage-function class $\mathcal{H}_{\mathtt{pkow}}(\cdot)$. In the modified definition of the function class, the PPT algorithm (or inverter) $\mathcal{A}$ is given not only the public-key of the key-pair (pk, sk), but also some variables which are generated during generation of the key-pair and are not directly included in either pk or sk. Specifically, for our signature scheme, the variables are the decryption-key dk and the trapdoor td. If the definition of the leakage-function class is modified to such one, the simulator in the proof for $\mathtt{Game}_{final}$ is not forced to guess dk and td with probability $2^{-(l_{dk}+l_{td})}$, so the polynomially hard-to-invert leakage-resilience security is achieved. Instantiating the generic construction of the signature scheme, we can concretely generate the first signature scheme (weakly unforgeable and) resilient to polynomially hard-to-invert leakage under standard assumptions such as the DLIN assumption.

In the third step, we apply a methodology which is invented by modifying the one by Wang et al. [WT14] to the weakly unforgeable signature scheme in the second step, then get a strongly unforgeable one. Note that unlike Wang et al., we do not propose a generic transformation from a weakly unforgeable and resilient to hard-to-invert leakage to a strongly unforgeable one. In the transformation by Wang et al., a chameleon hash function with strong collision-resistance in the bounded leakage model (BLR-CHF) was used. We use a CHF with strong collision-resistance in HL model (HLR-CHF). Moreover, the secret-key of the strongly unforgeable signature scheme obtained by the transformation by Wang et al. includes not only the *original* secret-key, i.e., the secret-key of the weakly unforgeable signature, but also the secret-key of the BLR-CHF. However, the secret-key of our strongly unforgeable signature includes only the secret-key of the HLR-CHF. By instantiating the signature scheme, we obtain the first concrete construction of digital signature strongly unforgeable and resilient to polynomially hard-to-invert leakage under standard assumptions such as the DLIN assumption.

4.1.5 Organization

This chapter is organized as follows. In Sect. 4.2, we give the syntax and the security notions of chameleon hash function. In the section, we also give the definition of strong unforgeability in HL model of digital signature. In Sect. 4.3, our generic construction of signature and its security proof are given. In Sect. 4.4, we show that the generic construction of signature given in the previous section can be instantiated under the DLIN assumption.

4.2 Preliminaries for Chapter 4

4.2.1 Chameleon Hash Function (CHF)

A chameleon hash function scheme consists of the polynomial time algorithms $\{\mathsf{Gen}, \mathsf{Eval}, \mathsf{TC}, \mathsf{SKVer}, \mathsf{SKVer2}\}$, where SKVer and $\mathsf{SKVer2}$ are algorithms originally introduced by us. Gen and TC are probabilistic, and Eval, SKVer and $\mathsf{SKVer2}$ are deterministic.

$\mathsf{Gen}(1^\lambda) \rightarrow (pk, sk)$**.** The key-generation algorithm takes a security parameter 1^λ, where $\lambda \in \mathbb{N}$, as an input, and outputs a public-key pk and a secret-key (or trapdoor) sk. The message space $\mathcal{M}$, randomness space $\mathcal{R}$ and hashed value space $\mathcal{H}$ are uniquely determined by pk.

$\mathsf{Eval}(pk, m; r) \rightarrow h$**.** The evaluation algorithm takes the public-key pk and a message $m \in \mathcal{M}$ as inputs, and outputs the hashed value $h \in \mathcal{H}$ which was calculated under a randomness $r \in \mathcal{R}$.

$\mathsf{TC}(pk, sk, (m_1, r_1), m_2) \rightarrow r_2$**.** The trapdoor collision finder algorithm takes the public-key pk, the secret-key sk, a pair of a message and randomness $(m_1, r_1) \in \mathcal{M} \times \mathcal{R}$, and a message $m_2 \in \mathcal{M}$ as inputs, and outputs a randomness $r_2 \in \mathcal{R}$.

$\mathsf{SKVer}(pk, sk') \rightarrow 1\ /\ 0$. The first secret-key-verification algorithm takes the public-key pk and a secret-key sk' as inputs, and outputs 1 or 0.

$\mathsf{SKVer2}(pk, sk', sk^{\dagger}) \rightarrow 1\ /\ 0$. The second secret-key-verification algorithm takes the public-key pk, a secret-key sk', and a secret-key $sk^{\dagger}$ as inputs, and outputs 1 or 0. Even if the two secret-keys are inputted in the reversed order, the output is required to be equivalent. Thus, for any $\lambda \in \mathbb{N}$, any $(pk, sk) \leftarrow \mathsf{Gen}(1^{\lambda})$, and any two valid secret-keys sk' and $sk^{\dagger}$, it holds that $\mathsf{SKVer2}(pk, sk', sk^{\dagger}) = \mathsf{SKVer2}(pk, sk^{\dagger}, sk')$.

Every CHF scheme must be correct. A CHF scheme $\Sigma_{\mathrm{CHF}} = \{\mathsf{Gen}, \mathsf{Eval}, \mathsf{TC}, \mathsf{SKVer}, \mathsf{SKVer2}\}$ is correct, if for every $\lambda \in \mathbb{N}$, every $(pk, sk) \leftarrow \mathsf{Gen}(1^{\lambda})$, every $m \in \mathcal{M}$, every $m' \in \mathcal{M}$, every $r \in \mathcal{R}$, and every $r' := \mathsf{TC}(pk, sk, (m, r), m')$, it holds that $[\mathsf{Eval}(pk, m; r) = \mathsf{Eval}(pk, m'; r')] \wedge [1 \leftarrow \mathsf{SKVer}(pk, sk)] \wedge [1 \leftarrow \mathsf{SKVer2}(pk, sk, sk)]$.

We give the definitions of two standard properties for the CHF scheme. They are strong collision-resistance and random trapdoor collision.

Definition 16. *A CHF scheme $\Sigma_{\mathrm{CHF}} = \{\mathsf{Gen}, \mathsf{Eval}, \mathsf{TC}, \mathsf{SKVer}, \mathsf{SKVer2}\}$ is strongly collision-resistant, if for every $\lambda \in \mathbb{N}$ and every PPT $\mathcal{A}$, it holds that*

$$\begin{aligned}\Pr[\mathcal{A}(pk) \rightarrow ((m_1, r_1), (m_2, r_2))\ s.t.\ &[(m_1, r_1) \neq (m_2, r_2)] \\ &\wedge [\mathsf{Eval}(pk, m_1; r_1) = \mathsf{Eval}(pk, m_2; r_2)]]\end{aligned}$$

is negligible, where $(pk, sk) \xleftarrow{\mathrm{R}} \mathsf{Gen}(1^{\lambda})$.

Definition 17. *A CHF scheme $\Sigma_{\mathrm{CHF}} = \{\mathsf{Gen}, \mathsf{Eval}, \mathsf{TC}, \mathsf{SKVer}, \mathsf{SKVer2}\}$ is said to have the property of random trapdoor collision, if for any $\lambda \in \mathbb{N}$, any $(pk, sk) \leftarrow \mathsf{Gen}(1^{\lambda})$ and any two messages $m_1, m_2 \in \mathcal{M}$, a randomness r_1 chosen uniformly at random from $\mathcal{R}$ distributes identically with $r_2 := \mathsf{TC}(pk, (m_1, r_1), m_2)$.*

We give the definition of an original property for a CHF scheme. The property is hard-to-compute-secret-key (HtC-SK).

Definition 18. $\Sigma_{\mathrm{CHF}} = \{\mathsf{Gen}, \mathsf{Eval}, \mathsf{TC}, \mathsf{SKVer}, \mathsf{SKVer2}\}$ *is said to have the property of HtC-SK, if for every $\lambda \in \mathbb{N}$, and every PPT $\mathcal{A}$, it holds that*

$$\Pr\left[\mathcal{A}(pk, sk) \rightarrow sk^*\ s.t.\ [1 \leftarrow \mathsf{SKVer}(pk, sk^*)] \wedge [0 \leftarrow \mathsf{SKVer2}(pk, sk^*, sk)]\right]$$

is negligible, where $(pk, sk) \xleftarrow{\mathrm{R}} \mathsf{Gen}(1^{\lambda})$.

Remark. The property is related to the second pre-image resistance (SPR) [DHLAW10b, FHN+12]. The algorithm SKVer given pk and sk as inputs is an algorithm which verifies whether or not a relation holds between pk and sk. The algorithm SKVer2 given two secret-keys sk and $sk^{\dagger}$ can be defined as the algorithm outputting 1 iff the two keys are equivalent. Thus, the HtC-SK property can be the SPR property. We can say that the HtC-SK property is a generalization of the SPR property.

4.2.2 Definition of Strong Existential Unforgeability in HL Model of Signature

We consider strong existential unfrogeability in HL model for signature schemes. Specifically, we define strong existential unforgeability against adaptively chosen messages attack in HL model (HL-sEUF-CMA) for signature schemes.

At first, we define a game w.r.t. a signature scheme $\Sigma_{\mathrm{SIG}} = \{\mathsf{Gen}, \mathsf{Sig}, \mathsf{Ver}\}$, which is played between an adversary $\mathcal{A}$ and challenger $\mathcal{CH}$, as follows. Note that a leakage function $f : \{0,1\}^{|pk|+|sk|} \to \{0,1\}^*$ whose randomness space is denoted by $\mathcal{R}$ is included in a class $\mathcal{F}_{\Sigma_{\mathrm{SIG}}}(\lambda)$, i.e., $f \in \mathcal{F}_{\Sigma_{\mathrm{SIG}}}(\lambda)$ [2].

Key-Generation. $\mathcal{CH}$ runs $(pk, sk) \leftarrow \mathrm{SIG}.\mathsf{Gen}(1^\lambda)$. $\mathcal{CH}$ chooses $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then computes $f(pk, sk; r)$. $\mathcal{CH}$ sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{CH}$ initializes the list $\mathcal{L}_S$ as an empty set $\emptyset$.

Query. $\mathcal{A}$ is allowed to use the signing oracle **Sign**, adaptively.

> **Sign**$(m \in \mathcal{M})$**:** $\mathcal{CH}$ generates $\sigma \leftarrow \mathrm{SIG}.\mathsf{Sig}(pk, m, sk)$, then sends σ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \sigma)\}$.

Forgery(m^*, σ^*)**.** $\mathcal{CH}$ receives (m^*, σ^*) sent by $\mathcal{A}$.

In the above game, $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathrm{SIG}.\mathsf{Ver}(pk, m^*, \sigma^*)] \wedge [(m^*, \sigma^*) \notin \mathcal{L}_S]$. Advantage $\mathsf{Adv}^{\mathcal{F}(\lambda)-HL-sEUF-CMA}_{\Sigma_{\mathrm{SIG}},\mathcal{A}}(\lambda)$ is defined as the probability $\Pr[\mathcal{A}\ wins.]$.

Definition 19. Σ_{SIG} *is HL-sEUF-CMA-secure with respect to the leakage-function class* $\mathcal{F}_{\Sigma_{\mathrm{SIG}}}(\lambda)$*, if for every PPT* $\mathcal{A}$ *and every function* $f \in \mathcal{F}_{\Sigma_{\mathrm{SIG}}}(\lambda)$*,* $\mathsf{Adv}^{\mathcal{F}(\lambda)-HL-sEUF-CMA}_{\Sigma_{\mathrm{SIG}},\mathcal{A}}(\lambda)$ *is negligible.*

Remark. Its weaker version, i.e., weak existential unforgeability against chosen messages attack (wEUF-CMA), is defined in the same manner as sEUF-CMA except for the winning condition of the adversary $\mathcal{A}$ in the game. The adversary is said to win the game if the signature σ^* is a valid signature on the message m^*, i.e., $[1 \leftarrow \mathrm{SIG}.\mathsf{Ver}(pk, m^*, \sigma^*)]$, and the message m^* has not been queried to the signing oracle **Sign**.

4.3 Proposed Signature Scheme

In Subsect. 4.3.1, the generic construction of our signature scheme is given. In Subsect. 4.3.2, the signature scheme is proven to be strongly existentially unforgeable and resilient to polynomially hard-to-invert leakage. In the next section, i.e., Section 4.4, we show that the signature scheme can be instantiated under the DLIN assumption.

[2]In this paper, the function class $\mathcal{F}_{\Sigma_{\mathrm{SIG}}}(\lambda)$ can be simply written as $\mathcal{F}(\lambda)$, if it is obvious that the function class is for the signature scheme Σ_{SIG}.

4.3.1 Generic Construction

Our generic construction of signature scheme $\Sigma_{\text{SIG}} = \{\mathsf{SIG.Gen}, \mathsf{SIG.Sig}, \mathsf{SIG.Ver}\}$ has the following 4 building blocks: An LPKE scheme $\Sigma_{\text{LPKE}} = \{\mathsf{LPKE.Gen}, \mathsf{LPKE.Enc}, \mathsf{LPKE.Dec}\}$, an NIZK scheme $\Sigma_{\text{NIZK}} = \{\mathsf{NIZK.Gen}, \mathsf{NIZK.Pro}, \mathsf{NIZK.Ver}\}$, a CHF scheme $\Sigma_{\text{CHF}} = \{\mathsf{CHF.Gen}, \mathsf{CHF.Eval}, \mathsf{CHF.TC}, \mathsf{CHF.SKVer}, \mathsf{CHF.SKVer2}\}$ and a CHF scheme $\Sigma_{\text{CHF2}} = \{\mathsf{CHF2.Gen}, \mathsf{CHF2.Eval}, \mathsf{CHF2.TC}\}$.

The signature scheme Σ_{SIG} is generically constructed as follows.

$\mathsf{SIG.Gen}(1^\lambda)$: Run $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, $(pk_1, sk_1) \leftarrow \mathsf{CHF.Gen}(1^\lambda)$ and $(pk_2, sk_2) \leftarrow \mathsf{CHF2.Gen}(1^\lambda)$. Run $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$, where $\mathcal{S}_1$ is the first simulator in the definition of zero-knowledge for Σ_{NIZK}.

$\mathcal{R}_E$ and $\mathcal{R}_{E2}$ denote the randomness space of $\mathsf{CHF.Eval}$ and $\mathsf{CHF2.Eval}$, respectively. $\tilde{\mathcal{M}}$, $\bar{\mathcal{M}}$, C, $\mathcal{P}$ and $\mathcal{K}_1$ denote the message space of Σ_{CHF}, the label space of Σ_{LPKE} (or the hashed value space of Σ_{CHF2}), the ciphertext space of Σ_{LPKE}, the proof space of Σ_{NIZK}, and the secret-key space of Σ_{CHF}, respectively. $\mathcal{M}$ is a space satisfying $\tilde{\mathcal{M}} = \mathcal{M} \| C \| \mathcal{P}$.

Verification-key and signing-key are set as $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. `Return` (pk, sk). We define language L as

$$L := \Big\{(c, \bar{m}) \in C \times \bar{\mathcal{M}} \mid {}^{\exists} sk_1 \in \mathcal{K}_1 \ s.t.\ [c \leftarrow \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1)]\Big\}. \quad (4.1)$$

$\mathsf{SIG.Sig}(pk, m \in \mathcal{M}, sk)$: pk is parsed as (pk_1, pk_2, ek, crs). sk is written as sk_1. Do as follows in order.

- $r'_E \overset{\text{U}}{\leftarrow} \mathcal{R}_E$, $r_{E2} \overset{\text{U}}{\leftarrow} \mathcal{R}_{E2}$, $m' \overset{\text{U}}{\leftarrow} \mathcal{M}$, $c' \overset{\text{U}}{\leftarrow} C$, $\pi' \overset{\text{U}}{\leftarrow} \mathcal{P}$, $\sigma' := (c', \pi')$.
- $h := \mathsf{CHF.Eval}(pk_1, m' \| \sigma'; r'_E)$, $\bar{m} := \mathsf{CHF2.Eval}(pk_2, h; r_{E2})$.
- $c := \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})$, $x := (c, \bar{m})$, $w := sk_1$, $\pi := \mathsf{NIZK.Pro}(crs, x, w)$.
- $\sigma := (c, \pi)$, $r_E := \mathsf{CHF.TC}(pk_1, sk_1, (m' \| \sigma', r'_E), m \| \sigma)$.

`Return` $\sigma^\dagger := (\sigma, r_E, r_{E2}) = (c, \pi, r_E, r_{E2})$.

$\mathsf{SIG.Ver}(pk, m \in \mathcal{M}, \sigma^\dagger)$: pk is parsed as (pk_1, pk_2, ek, crs). $\sigma^\dagger$ is parsed as (c, π, r_E, r_{E2}). $h := \mathsf{CHF.Eval}(pk_1, m \| \sigma; r_E)$. $\bar{m} := \mathsf{CHF2.Eval}(pk_2, h; r_{E2})$. $x := (c, \bar{m})$. `Return` $\mathsf{NIZK.Ver}(crs, x, \pi)$.

4.3.2 Proof of Strong Unforgeability in HL Model

Before giving the theorem for the strong unforgeability in the hard-to-invert leakage model of the signature scheme Σ_{SIG}, we give the definitions of the leakage-function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ for the signature scheme and the strong collision-resistance in HL model w.r.t. the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ for the chameleon hash function Σ_{CHF}.

Definition 20. *Function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ consists of every polynomial-time computable probabilistic (or deterministic) function $f : \{0,1\}^{|pk_1|+|pk_2|+|ek|+|crs|+|sk_1|} \to \{0,1\}^*$ which has a randomness space $\mathcal{R}$ and satisfies the following condition: for every PPT $\mathcal{B}$,*

$$\begin{aligned} &\Pr\left[\mathcal{B}\left(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r)\right) \to sk_1^*\right. \\ &\left.\text{s.t. } [1 \leftarrow \mathsf{CHF.SKVer}\left(pk_1, sk_1^*\right)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}\left(pk_1, sk_1^*, sk_1\right)]\right] \end{aligned} \quad (4.2)$$

is negligible, where $(pk_1, sk_1) \overset{R}{\leftarrow} \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{R}{\leftarrow} \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{R}{\leftarrow} \mathsf{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{R}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{R}{\leftarrow} \mathcal{R}$.

Remark. If the chameleon hash function Σ_{CHF} is a CHF with the second pre-image resistance [DHLAW10b, FHN⁺12], the algorithm $\mathsf{CHF.SKVer2}$ is defined as the equality-checking algorithm, and the secret-key sk_1^* which satisfies $[1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]$ is the original secret-key sk_1 only. So, the probability (4.2) is simply written as $\Pr[\mathcal{B}(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r)) \to sk_1]$.

Definition 21. *CHF scheme $\Sigma_{\text{CHF}} = \{\mathsf{CHF.Gen}, \mathsf{CHF.Eval}, \mathsf{CHF.TC}, \mathsf{CHF.SKVer}, \mathsf{CHF.SKVer2}\}$ is said to be strongly collision-resistant in HL model with respect to the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$, if for every PPT $\mathcal{A}$ and every function $f : \{0,1\}^{|pk_1|+|pk_2|+|ek|+|crs|+|sk_1|} \to \{0,1\}^*$ which is included in the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ and has a randomness space $\mathcal{R}$,*

$$\begin{aligned} &\Pr\left[\mathcal{A}\left(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r)\right) \to \left((m_1, r_1), (m_2, r_2)\right)\right. \\ &\left.\text{s.t. } [(m_1, r_1) \neq (m_2, r_2)] \wedge [\mathsf{CHF.Eval}(pk_1, m_1; r_1) = \mathsf{CHF.Eval}(pk_1, m_2; r_2)]\right] \end{aligned}$$

is negligible, where $(pk_1, sk_1) \overset{R}{\leftarrow} \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{R}{\leftarrow} \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{R}{\leftarrow} \mathsf{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{R}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{R}{\leftarrow} \mathcal{R}$.

The strong unforgeability in HL model of the signature scheme Σ_{SIG} is guaranteed by the following theorem.

Theorem 4.3.1. Σ_{SIG} *is HL-sEUF-CMA w.r.t. the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$, if*

- Σ_{LPKE} *is IND-wLCCA,*
- Σ_{NIZK} *is sound and zero-knowledge,*
- Σ_{CHF} *is strongly collision-resistant in HL model w.r.t. the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$, random trapdoor collision, and HtC-SK, and*
- Σ_{CHF2} *is strongly collision-resistant and random trapdoor collision.*

Proof of Theorem 4.3.1. Hereafter, $q_s \in \mathbb{N}$ denotes the number of times that PPT adversary $\mathcal{A}$ uses the signing oracle $\mathtt{Sign}$. To prove Theorem 4.3.1, we use multiple games $\mathtt{Game}_i$, where $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 7|1, \cdots, 7|q_s\}$.

The first game $\mathtt{Game}_0$ is the normal HL-sEUF-CMA game w.r.t. the signature scheme Σ_{SIG} and the function class $\mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$. Specifically, $\mathtt{Game}_0$ is the following game.

Key-Generation. $\mathcal{CH}$ runs $(pk_1, sk_1) \leftarrow \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \leftarrow \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk and sk are set as $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. For a function $f \in \mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$, $\mathcal{CH}$ chooses $r \xleftarrow{\mathrm{R}} \mathcal{R}$, then computes $f(pk, sk; r)$. $\mathcal{CH}$ sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m \in \mathcal{M}$ as a query to the signing oracle $\mathtt{Sign}$, $\mathcal{CH}$ generates a signature (c, π, r_E, r_{E2}) on the message as follows.

- $r'_E \xleftarrow{\mathrm{U}} \mathcal{R}_E$, $r_{E2} \xleftarrow{\mathrm{U}} \mathcal{R}_{E2}$, $m' \xleftarrow{\mathrm{U}} \mathcal{M}$, $c' \xleftarrow{\mathrm{U}} \mathcal{C}$, $\pi' \xleftarrow{\mathrm{U}} \mathcal{P}$, $\sigma' := (c', \pi')$.
- $h := \mathsf{CHF.Eval}(pk_1, m' \| \sigma'; r'_E)$, $\bar{m} := \mathsf{CHF2.Eval}(pk_2, h; r_{E2})$.
- $c := \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})$, $x := (c, \bar{m})$, $w := sk_1$, $\pi := \mathsf{NIZK.Pro}(crs, x, w)$.
- $\sigma := (c, \pi)$, $r_E := \mathsf{CHF.TC}(pk_1, sk_1, (m' \| \sigma', r'_E), m \| \sigma)$.

$\mathcal{CH}$ returns a signature (c, π, r_E, r_{E2}) to $\mathcal{A}$. $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, c, \pi, r_E, r_{E2})\}$.

Forgery$(m^*, (c^*, \pi^*, r^*_E, r^*_{E2}))$**.** $\mathcal{CH}$ is given a message $m^* \in \mathcal{M}$ and a signature $(c^*, \pi^*, r^*_E, r^*_{E2})$.

$\mathcal{A}$ wins the game, if $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S]$, where $h^* := \mathsf{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r^*_E)$, $\bar{m}^* := \mathsf{CHF2.Eval}(pk_2, h^*; r^*_{E2})$ and $x^* := (c^*, \bar{m}^*)$.

We define the games $\mathtt{Game}_i$, where $i \in \{1, 2, 3, 4, 5, 6, 7, 7|1, \cdots, 7|q_s\}$, as follows.

$\mathtt{Game}_1$. $\mathtt{Game}_1$ is the same as $\mathtt{Game}_0$ except that $\mathcal{CH}$ generates a common reference string crs by running $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$ in **Key-Generation**.

$\mathtt{Game}_2$. $\mathtt{Game}_2$ is the same as $\mathtt{Game}_1$ except that $\mathcal{A}$'s winning condition is changed to the following one, where $sk^*_1 := \mathsf{LPKE.Dec}(dk, c^*, \bar{m}^*)$: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk^*_1)]$.

$\mathtt{Game}_3$. $\mathtt{Game}_3$ is the same as $\mathtt{Game}_2$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk^*_1, sk_1)]$.

$\mathtt{Game}_4$: $\mathtt{Game}_4$ is the same as $\mathtt{Game}_3$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk^*_1, sk_1)] \wedge [[\bar{m}^* \notin \{\bar{m}_1, \cdots, \bar{m}_{q_s}\}] \vee [{}^{\exists}i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) = (h_i, r_{E2,i})] \wedge [(m^*, c^*, \pi^*, r^*_E) \neq (m_i, c_i, \pi_i, r_{E,i})]]]]$, where, for $i \in [1, q_s]$, each one of $\bar{m}_i, h_i, r_{E2,i}, c_i, \pi_i$ and $r_{E,i}$ is the element which was generated when computing the reply to the i-th signing oracle query.

Game$_5$ Game$_5$ is the same as Game$_4$ except that when $\mathcal{A}$ issues a message $m \in \mathcal{M}$ as a query to the signing oracle **Sign**, $\mathcal{CH}$ generates a signature (c, π, r_E, r_{E2}) on the message as follows.

- $r_E, r'_E \overset{U}{\leftarrow} \mathcal{R}_E$, $r'_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}$, $m' \overset{U}{\leftarrow} \mathcal{M}$, $c' \overset{U}{\leftarrow} C$, $\pi' \overset{U}{\leftarrow} \mathcal{P}$, $\sigma' := (c', \pi')$.
- $h' :=$ CHF.Eval$(pk_1, m'\|\sigma'; r'_E)$, $\bar{m} :=$ CHF2.Eval$(pk_2, h'; r'_{E2})$.
- $c :=$ LPKE.Enc$(ek, sk_1, \bar{m})$, $x := (c, \bar{m})$, $w := sk_1$, $\pi :=$ NIZK.Pro(crs, x, w).
- $\sigma := (c, \pi)$, $h :=$ CHF.Eval$(pk_1, m\|\sigma; r_E)$.
- $r_{E2} :=$ CHF2.TC$(pk_2, sk_2, (h', r'_{E2}), h)$.

Game$_6$. Game$_6$ is the same as Game$_5$ except that the following two points. Firstly, $\mathcal{CH}$ generates a common reference string crs by running $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ in **Key-Generation**. Secondly, when replying to a query to **Sign** in **Query**, $\mathcal{CH}$ generates a proof π by using $\mathcal{S}_2$, instead of NIZK.Pro, where $\mathcal{S}_2$ denotes the second simulator in the definition of zero-knowledge for Σ_{NIZK}.

Game$_7$(= **Game$_{7|0}$**). Game$_7$ is the same as Game$_6$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \text{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \wedge [\bar{m}^* \notin \{\bar{m}_1, \cdots, \bar{m}_{q_s}\}]$.

Game$_{7|1}$ $, \cdots,$ **Game$_{7|q_s}$**. Game$_{7|i}$, where $i \in [1, q_s]$, is the same as Game$_{7|0}$ except that when replying to the j-th signing oracle query, where $j \leq i$, $\mathcal{CH}$ generates the ciphertext c_j by running $c_j \leftarrow$ LPKE.Enc$(ek, 0^{|sk_1|}, \bar{m}_j)$, where $0^{|sk_1|}$ denotes the bit-string of $|sk_1|$ number of 0.

Hereafter, W_i, where $i \in \{0, 1, 2, 3, 4, 5, 6, 7, 7|1, \cdots, 7|q_s\}$, denotes the event that $\mathcal{A}$ wins the game Game$_i$. It holds obviously that

$$\begin{aligned}
\mathsf{Adv}_{\Sigma_{\text{SIG}},\mathcal{A}}^{\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)\text{-}HL\text{-}sEUF\text{-}CMA}(\lambda) &= \Pr[W_0] \\
&\leq \sum_{i=1}^{7} |\Pr[W_{i-1}] - \Pr[W_i]| + \sum_{i=1}^{q_s} \left|\Pr[W_{7|i-1}] - \Pr[W_{7|i}]\right| + \Pr\left[W_{7|q_s}\right].
\end{aligned}$$

Theorem 4.3.1 is proven by the above inequality and the following all lemmas. □

Lemma 4.3.1. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 4.3.2. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible if* Σ_{NIZK} *is sound.*

Lemma 4.3.3. $|\Pr[W_2] - \Pr[W_3]|$ *is negligible if* Σ_{CHF} *is HtC-SK.*

Lemma 4.3.4. $|\Pr[W_3] - \Pr[W_4]|$ *is negligible if* Σ_{CHF2} *is strongly collision-resistant.*

Lemma 4.3.5. $|\Pr[W_4] - \Pr[W_5]|$ *is negligible if each one of* Σ_{CHF} *and* Σ_{CHF2} *is random trapdoor collision.*

Lemma 4.3.6. $|\Pr[W_5] - \Pr[W_6]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 4.3.7. $|\Pr[W_6] - \Pr[W_{7|0}]|$ *is negligible if* Σ_{CHF} *is strongly collision-resistant in HL model w.r.t. the function class* $\mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$.

Lemma 4.3.8. *For any* $i \in [1, q_s]$, $|\Pr[W_{7|i-1}]-\Pr[W_{7|i}]|$ *is negligible if* Σ_{LPKE} *is IND-wLCCA.*

Lemma 4.3.9. $\Pr[W_{7|q_s}]$ *is negligible.*

Proof of each lemma is given below.

Proof of Lemma 4.3.1. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_0]-\Pr[W_1]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$. On one hand, the simulator $\mathcal{S}$ is a PPT algorithm attempting to break the zero-knowledge for Σ_{NIZK}. On the other hand, $\mathcal{S}$ is the challenger in $\mathtt{Game}_0$ or $\mathtt{Game}_1$. $\mathcal{S}$ is given a common reference string crs. If crs was generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow$ NIZK.Gen(1^λ)), then $\mathcal{S}$ simulates $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) against the PPT adversary $\mathcal{A}$ properly. The concrete behavior by $\mathcal{S}$ is the following.

Key-Generation. $\mathcal{S}$ runs $(pk_1, sk_1) \leftarrow$ CHF.Gen(1^λ), $(pk_2, sk_2) \leftarrow$ CHF2.Gen(1^λ) and $(ek, dk) \leftarrow$ LPKE.Gen(1^λ). $\mathcal{S}$ is given a common reference string crs of Σ_{NIZK}. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$ where $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$. $\mathcal{S}$ sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m \in \mathcal{M}$ as a query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature (c, π, r_E, r_{E2}) on the message in the normal manner. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\mathcal{L}_S \cup \{(m, c, \pi, r_E, r_{E2})\}$.

Forgery$(m^*, c^*, \pi^*, r^*_E, r^*_{E2})$**.** $\mathcal{S}$ outputs 1, if $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S]$, where $h^* := \text{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r^*_E)$, $\bar{m}^* := \text{CHF2.Eval}(pk_2, h^*; r^*_{E2})$ and $x^* := (c^*, \bar{m}^*)$. $\mathcal{S}$ outputs 0, otherwise.

It is obvious that if the common reference string is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow$ NIZK.Gen(1^λ)), then $\mathcal{S}$ simulates the game $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) against $\mathcal{A}$ perfectly, and if and only if the event W_0 (resp. W_1) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain $\Pr[W_0] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]$ and $\Pr[W_1] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)]$. Hence, $|\Pr[W_0] - \Pr[W_1]| = |\Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]|$. □

Proof of Lemma 4.3.2. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_1]-\Pr[W_2]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the soundness property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the soundness of the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Key-Generation. $\mathcal{S}$ runs $(pk_1, sk_1) \leftarrow \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \leftarrow \mathsf{CHF2.Gen}(1^\lambda)$ and $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$. $\mathcal{S}$ is given a common reference string crs of Σ_{NIZK}. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$ where $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m \in \mathcal{M}$ as a query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature (c, π, r_E, r_{E2}) on the message in the normal manner. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\mathcal{L}_S \cup \{(m, c, \pi, r_E, r_{E2})\}$.

Forgery$(m^*, c^*, \pi^*, r_E^*, r_{E2}^*)$. $\mathcal{S}$ checks whether the following condition is satisfied or not: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \wedge [0 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)]$, where $h^* := \mathsf{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r_E^*)$, $\bar{m}^* := \mathsf{CHF2.Eval}(pk_2, h^*; r_{E2}^*)$, $x^* := (c^*, \bar{m}^*)$ and $sk_1^* := \mathsf{LPKE.Dec}(dk, c^*, \bar{m}^*)$.

If the condition is satisfied, then $\mathcal{S}$ outputs $(x^*, \pi^*) = (c^*, \bar{m}^*, \pi^*)$.

It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_1$ or $\mathtt{Game}_2$, perfectly.

By the way, the definitions of W_1 and W_2 gives us the following equations.

$$\begin{aligned} \Pr[W_1] &= \Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S]] && (4.3) \\ \Pr[W_2] &= \Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\ &\quad \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)]] && (4.4) \end{aligned}$$

Hence, we obtain

$$\begin{aligned} &|\Pr[W_1] - \Pr[W_2]| \\ = \ &\Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\ &\wedge [0 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)]] \\ = \ &\Pr[\mathcal{S}(crs) \rightarrow (x^*, \pi^*) \ s.t. \ [1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \\ &\wedge [0 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)]]. && (4.5) \end{aligned}$$

By the definition of the language L given in (4.1), the following statement is true: for any $(c, \bar{m}) \in L$, there exists sk_1 such that $[c \leftarrow \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1)]$.

By the above statement, the correctness of Π_{LPKE}, and the correctness of Π_{CHF}, the following statement is true: for any $(c, \bar{m}) \in L$, it holds that $[1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1)]$, where $sk_1 := \mathsf{LPKE.Dec}(dk, c, \bar{m})$.

The contraposition of the above statement is the following statement: for any $c \in C$ and any $\bar{m} \in \bar{\mathcal{M}}$, if $[0 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1)]$, where $sk_1 := \mathsf{LPKE.Dec}(dk, c, \bar{m})$, then $(c, \bar{m}) \notin L$.

By the above statement and the equation (5.4), we obtain

$$|\Pr[W_1] - \Pr[W_2]| = \Pr[\mathcal{S}(crs) \rightarrow (x^*, \pi^*) \ s.t. \ [1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [x^* \notin L]].$$

□

Proof of Lemma 4.3.3. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_2]-\Pr[W_3]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the HtC-SK property for Σ_{CHF}.

We consider a PPT simulator $\mathcal{S}$ who behaves as a PPT adversary trying to break the property of HtC-SK for Σ_{CHF}. The concrete behavior by $\mathcal{S}$ is the following.

Key-Generation. $\mathcal{S}$ is given (pk_1, sk_1) of the keys of Σ_{CHF}. $\mathcal{S}$ runs $(pk_2, sk_2) \leftarrow \mathrm{CHF2.Gen}(1^\lambda)$, $(ek, dk) \leftarrow \mathrm{LPKE.Gen}(1^\lambda)$ and $crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)$. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$, where $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m \in \mathcal{M}$ as a query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature (c, π, r_E, r_{E2}) on the message in the normal manner. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\mathcal{L}_S \cup \{(m, c, \pi, r_E, r_{E2})\}$.

Forgery$(m^*, c^*, \pi^*, r_E^*, r_{E2}^*)$**.** $\mathcal{S}$ checks whether the following condition is satisfied or not: $[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathrm{CHF.SKVer}(pk_1, sk_1^*)] \wedge [0 \leftarrow \mathrm{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]$, where $h^* := \mathrm{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r_E^*)$, $\bar{m}^* := \mathrm{CHF2.Eval}(pk_2, h^*; r_{E2}^*)$, $x^* := (c^*, \bar{m}^*)$ and $sk_1^* := \mathrm{LPKE.Dec}(dk, c^*, \bar{m}^*)$.

If the condition is satisfied, $\mathcal{S}$ outputs sk_1^*.

It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_2$ or $\mathtt{Game}_3$ against $\mathcal{A}$ perfectly.

By the definitions of W_2 and W_3, we obtain

$$\begin{aligned}
\Pr[W_2] &= \Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \mathrm{CHF.SKVer}(pk_1, sk_1^*)]] \\
\Pr[W_3] &= \Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \mathrm{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathrm{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]]
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
&|\Pr[W_2] - \Pr[W_3]| = \Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\
&\wedge [1 \leftarrow \mathrm{CHF.SKVer}(pk_1, sk_1^*)] \wedge [0 \leftarrow \mathrm{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]]\,. \qquad (4.6)
\end{aligned}$$

The above probability is equal to the probability with whom $\mathcal{S}$ wins the HtC-SK property game for Π_{CHF}. □

Proof of Lemma 4.3.4. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_3]-\Pr[W_4]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the strong collision-resistance property for Σ_{CHF2}.

We consider a PPT simulator $\mathcal{S}$ who behaves as a PPT adversary trying to break the property of strong collision-resistance for Σ_{CHF2}. The concrete behavior by $\mathcal{S}$ is the following.

Key-Generation. $\mathcal{S}$ is given a public key pk_2 of Σ_{CHF2}. $\mathcal{S}$ runs $(pk_1, sk_1) \leftarrow \mathsf{CHF.Gen}(1^\lambda)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$ and $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$, where $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ and $\mathcal{L}_{\bar{m}}$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m_i \in \mathcal{M}$ as the i-th query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ on the message as follows.

- $r'_{E,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_E$, $r_{E2,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_{E2}$, $m'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{M}$, $c'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{C}$, $\pi'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{P}$, $\sigma'_i := (c'_i, \pi'_i)$.
- $h_i := \mathsf{CHF.Eval}(pk_1, m'_i \| \sigma'_i; r'_{E,i})$, $\bar{m}_i := \mathsf{CHF2.Eval}(pk_2, h_i; r_{E2,i})$.
- $c_i := \mathsf{LPKE.Enc}(ek, sk_1, \bar{m}_i)$, $x_i := (c_i, \bar{m}_i)$, $w := sk_1$, $\pi_i := \mathsf{NIZK.Pro}(crs, x_i, w)$.
- $\sigma_i := (c_i, \pi_i)$, $r_{E,i} := \mathsf{CHF.TC}(pk_1, sk_1, (m'_i \| \sigma'_i, r'_{E,i}), m_i \| \sigma_i)$.

$\mathcal{S}$ returns $(c_i, \pi_i, r_{E2,i}, r_{E,i})$ to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\mathcal{L}_S \cup \{(m_i, c_i, \pi_i, r_{E2,i}, r_{E,i})\}$. $\mathcal{L}_{\bar{m}}$ is set to $\mathcal{L}_{\bar{m}} \cup \{\bar{m}_i\}$.

Forgery$(m^*, c^*, \pi^*, r^*_E, r^*_{E2})$**.** $\mathcal{S}$ sets $h^* := \mathsf{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r^*_E)$, $\bar{m}^* := \mathsf{CHF2.Eval}(pk_2, h^*; r^*_{E2})$, $x^* := (c^*, \bar{m}^*)$ and $sk_1^* := \mathsf{LPKE.Dec}(dk, c^*, \bar{m}^*)$. If $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \wedge [{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) \neq (h_i, r_{E2,i})]]$, then $\mathcal{S}$ outputs $((h^*, r^*_{E2}), (h_i, r_{E2,i}))$.

It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_3$ or $\mathtt{Game}_4$ against $\mathcal{A}$ perfectly. By the definitions of W_3 and W_4, we obtain

$$\begin{aligned}
&\Pr[W_3] = \Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \\
&\wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]] \qquad (4.7)\\
&\Pr[W_4] = \Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \\
&\wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \\
&\wedge \Big[[\bar{m}^* \notin \mathcal{L}_{\bar{m}}] \vee \Big[{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) = (h_i, r_{E2,i})] \\
&\wedge [(m^*, c^*, \pi^*, r^*_E) \neq (m_i, c_i, \pi_i, r_{E,i})]\Big]\Big]] \qquad (4.8)
\end{aligned}$$

By (4.7) and (4.8), we obtain

$$\begin{aligned}
&|\Pr[W_3] - \Pr[W_4]| \\
= \ &\Pr[[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \\
&\wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \\
&\wedge \Big[{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) \neq (h_i, r_{E2,i})]\Big]\Big]
\end{aligned}$$

The above probability is equal to the probability with whom $\mathcal{S}$ wins the strong collision-resistance game for Π_{CHF2}. □

Proof of Lemma 4.3.5. For a message $\hat{m} \in \mathcal{M}$ queried to the signing oracle in $\mathtt{Game}_4$ (resp. $\mathtt{Game}_5$), $P_4(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2})$ (resp. $P_5(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2})$) denotes the probability that the signature $(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}) \in \mathcal{C} \times \mathcal{P} \times \mathcal{R}_E \times \mathcal{R}_{E2}$ is generated.

For the probability $P_4(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2})$, we obtain

$$\begin{aligned}
&P_4\left(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}\right) = \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \wedge \left[\hat{r}_E = \mathsf{CHF.TC}(pk_1, sk_1, (m' \| (c', \pi'), r'_E), \hat{m} \| (\hat{c}, \hat{\pi}))\right] \wedge \left[\hat{r}_{E2} = r_{E2}\right] \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \qquad (4.9) \\
=\;& \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \\
&\quad \cdot \Pr\left[\hat{r}_E = \mathsf{CHF.TC}(pk_1, sk_1, (m' \| (c', \pi'), r'_E), \hat{m} \| (\hat{c}, \hat{\pi}))\right. \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \cdot \Pr\left[\hat{r}_{E2} = r_{E2} \mid r_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}\right] \qquad (4.10) \\
=\;& \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \cdot \frac{1}{|\mathcal{R}_E|} \cdot \frac{1}{|\mathcal{R}_{E2}|}, \qquad (4.11)
\end{aligned}$$

where $h' := \mathsf{CHF.Eval}(pk_1, m' \| (c', \pi'); r'_E)$, $\bar{m} := \mathsf{CHF2.Eval}(pk_2, h'; r_{E2})$ and $x := (\hat{c}, \bar{m})$ (4.9) is the definition. The transformation from (4.9) to (4.10) is correct since each event is independent. The transformation from (4.10) to (4.11) is correct since the CHF scheme Π_{CHF} is random trapdoor collision.

On the other hand, for the probability $P_5\left(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}\right)$, we obtain

$$\begin{aligned}
&P_5\left(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}\right) = \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \wedge \left[\hat{r}_E = r_E\right] \wedge \left[\hat{r}_{E2} = \mathsf{CHF2.TC}(pk_2, sk_2, (h', r'_{E2}), \hat{h})\right] \\
&\quad \left. \mid r_E, r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r'_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \qquad (4.12) \\
=\;& \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r'_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \\
&\quad \cdot \Pr\left[\hat{r}_E = r_E \mid r_E \overset{U}{\leftarrow} \mathcal{R}_E\right] \cdot \Pr\left[\hat{r}_{E2} = \mathsf{CHF2.TC}(pk_2, sk_2, (h', r'_{E2}), \hat{h})\right. \\
&\quad \left. \mid r_E, r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r'_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \qquad (4.13) \\
=\;& \Pr\left[\left[\hat{c} = \mathsf{LPKE.Enc}(ek, sk_1, \bar{m})\right] \wedge \left[\hat{\pi} = \mathsf{NIZK.Pro}(crs, x, sk_1)\right]\right. \\
&\quad \left. \mid r'_E \overset{U}{\leftarrow} \mathcal{R}_E, r'_{E2} \overset{U}{\leftarrow} \mathcal{R}_{E2}, m' \overset{U}{\leftarrow} \mathcal{M}, c' \overset{U}{\leftarrow} \mathcal{C}, \pi' \overset{U}{\leftarrow} \mathcal{P}\right] \cdot \frac{1}{|\mathcal{R}_E|} \cdot \frac{1}{|\mathcal{R}_{E2}|}, \qquad (4.14)
\end{aligned}$$

where $h' := \mathsf{CHF.Eval}(pk_1, m' \| (c', \pi'); r'_E)$, $\bar{m} := \mathsf{CHF2.Eval}(pk_2, h'; r'_{E2})$, $x := (\hat{c}, \bar{m})$, and $\hat{h} := \mathsf{CHF.Eval}(pk_1, \hat{m} \| (\hat{c}, \hat{\pi}); r_E)$. (4.12) is the definition. The transformation from (4.12) to (4.13) is correct since each event is independent. The transformation from (4.13) to (4.14) is correct since the CHF scheme Π_{CHF2} is random trapdoor collision.

By (4.11) and (4.14), $P_4(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}) = P_5(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2})$. Thus, for any $\hat{m} \in \mathcal{M}$ and any signature $(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2}) \in \mathcal{C} \times \mathcal{P} \times \mathcal{R}_E \times \mathcal{R}_{E2}$, the probability in $\mathtt{Game}_4$ that the signature $(\hat{c}, \hat{\pi}, \hat{r}_E, \hat{r}_{E2})$ on the message $\hat{m}$ is generated is equal to the one in $\mathtt{Game}_5$. □

<u>Proof of Lemma 4.3.6.</u> We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_5]-\Pr[W_6]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the zero-knowledge property for the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Key-Generation. $\mathcal{S}$ is given a common reference string crs of Σ_{NIZK}. $\mathcal{S}$ runs $(pk_1, sk_1) \leftarrow \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \leftarrow \mathsf{CHF2.Gen}(1^\lambda)$ and $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$, where $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets each list of $\mathcal{L}_S$ and $\mathcal{L}_{\bar{m}}$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m_i \in \mathcal{M}$ as the i-th query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ on the message as follows.

- $r_{E,i}, r'_{E,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_E$, $r'_{E2,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_{E2}$, $m'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{M}$, $c'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{C}$, $\pi'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{P}$, $\sigma'_i := (c'_i, \pi'_i)$.
- $h'_i := \mathsf{CHF.Eval}(pk_1, m'_i \| \sigma'_i; r'_{E,i})$, $\bar{m}_i := \mathsf{CHF2.Eval}(pk_2, h'_i; r'_{E2,i})$.
- $c_i := \mathsf{LPKE.Enc}(ek, sk_1, \bar{m}_i)$, $x_i := (c_i, \bar{m}_i)$, $w := sk_1$.
- Issues (x_i, w) as an query to O_{zk}, then receives a proof π_i.
- $\sigma_i := (c_i, \pi_i)$, $h_i := \mathsf{CHF.Eval}(pk_1, m_i \| \sigma_i; r_{E,i})$.
- $r_{E2,i} := \mathsf{CHF2.TC}(pk_2, sk_2, (h'_i, r'_{E2,i}), h_i)$.

$\mathcal{S}$ returns $(c_i, \pi_i, r_{E2,i}, r_{E,i})$ to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\mathcal{L}_S \cup \{(m_i, c_i, \pi_i, r_{E2,i}, r_{E,i})\}$. $\mathcal{L}_{\bar{m}}$ is set to $\mathcal{L}_{\bar{m}} \cup \{\bar{m}_i\}$.

Forgery$(m^*, c^*, \pi^*, r^*_E, r^*_{E2})$**.** $\mathcal{S}$ checks whether the following condition is satisfied or not: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk^*_1, sk_1)] \wedge [[\bar{m}^* \notin \{\bar{m}_1, \cdots, \bar{m}_{q_s}\}] \vee [{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) = (h_i, r_{E2,i})] \wedge [(m^*, c^*, \pi^*, r^*_E) \neq (m_i, c_i, \pi_i, r_{E,i})]]]$, where $h^* := \mathsf{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r^*_E)$, $\bar{m}^* := \mathsf{CHF2.Eval}(pk_2, h^*; r^*_{E2})$, $x^* := (c^*, \bar{m}^*)$ and $sk^*_1 := \mathsf{LPKE.Dec}(dk, c^*, \bar{m}^*)$.

If the condition is satisfied, $\mathcal{S}$ outputs 1. Else, then $\mathcal{S}$ outputs 0.

It is obvious that if the common reference string crs is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$) and the proof-generation oracle O_{zk} is $O_1^{crs,td}$ (resp. O_0^{crs}), then $\mathcal{S}$ simulates $\mathtt{Game}_6$ (resp. $\mathtt{Game}_5$) against $\mathcal{A}$ perfectly, and if and only if W_6 (resp. W_5) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain

$$\begin{aligned}|\Pr[W_5] - \Pr[W_6]| = \Big|&\Pr\Big[1 \leftarrow \mathcal{S}^{O_0^{crs}(x,w)}(crs) \mid crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)\Big] \\ &- \Pr\Big[1 \leftarrow \mathcal{S}^{O_1^{crs,td}(x,w)}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)\Big]\Big|.\end{aligned}$$

□

Proof of Lemma 4.3.7. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_6]-\Pr[W_7]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the property of strong collision- resistance in HL model w.r.t. $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ for Σ_{CHF}.

We consider a PPT simulator $\mathcal{S}$ who behaves as a PPT adversary trying to break the property of strong collision resistance in HL model w.r.t. $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ for Σ_{CHF}. The concrete behavior by $\mathcal{S}$ is the following.

Key-Generation. $\mathcal{S}$ is given $(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r))$, where $(pk_1, sk_1) \overset{\text{R}}{\leftarrow} \text{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{\text{R}}{\leftarrow} \text{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{\text{R}}{\leftarrow} \text{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{\text{R}}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{\text{R}}{\leftarrow} \mathcal{R}$. $\mathcal{S}$ sets pk to (pk_1, pk_2, ek, crs). $\mathcal{S}$ sends the public key pk and the leakage $f(pk_1, pk_2, ek, crs, sk_1; r)$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ and $\mathcal{L}_{\bar{m}}$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m_i \in \mathcal{M}$ as the i-th query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ on the message as follows.

- $r_{E,i}, r'_{E,i} \overset{\text{U}}{\leftarrow} \mathcal{R}_E$, $r'_{E2,i} \overset{\text{U}}{\leftarrow} \mathcal{R}_{E2}$, $m'_i \overset{\text{U}}{\leftarrow} \mathcal{M}$, $c'_i \overset{\text{U}}{\leftarrow} C$, $\pi'_i \overset{\text{U}}{\leftarrow} \mathcal{P}$, $\sigma'_i := (c'_i, \pi'_i)$.
- $h'_i := \text{CHF.Eval}(pk_1, m'_i \| \sigma'_i; r'_{E,i})$, $\bar{m}_i := \text{CHF2.Eval}(pk_2, h'_i; r'_{E2,i})$.
- $c_i := \text{LPKE.Enc}(ek, sk_1, \bar{m}_i)$, $x_i := (c_i, \bar{m}_i)$, $\pi_i := \mathcal{S}_2(crs, x_i, td)$.
- $\sigma_i := (c_i, \pi_i)$, $h_i := \text{CHF.Eval}(pk_1, m_i \| \sigma_i; r_{E,i})$.
- $r_{E2,i} := \text{CHF2.TC}(pk_2, sk_2, (h'_i, r'_{E2,i}), h_i)$.

$\mathcal{S}$ returns $(c_i, \pi_i, r_{E2,i}, r_{E,i})$ to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\mathcal{L}_S \cup \{(m_i, c_i, \pi_i, r_{E2,i}, r_{E,i})\}$. $\mathcal{L}_{\bar{m}}$ is set to $\mathcal{L}_{\bar{m}} \cup \{\bar{m}_i\}$.

Forgery$(m^*, c^*, \pi^*, r^*_E, r^*_{E2})$. $\mathcal{S}$ checks whether the following condition is satisfied or not: $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \text{CHF.SKVer2}(pk_1, sk^*_1, sk_1)] \wedge [{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) = (h_i, r_{E2,i})] \wedge [(m^*, c^*, \pi^*, r^*_E) \neq (m_i, c_i, \pi_i, r_{E,i})]]$, where $h^* := \text{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r^*_E)$, $\bar{m}^* := \text{CHF2.Eval}(pk_2, h^*; r^*_{E2})$, $x^* := (c^*, \bar{m}^*)$ and $sk^*_1 := \text{LPKE.Dec}(dk, c^*, \bar{m}^*)$.

If the condition is satisfied, $\mathcal{S}$ outputs $((m^* \| (c^*, \pi^*), r^*_E), (m_i \| (c_i, \pi_i), r_{E,i}))$.

It is obvious that $\mathcal{S}$ simulates $\texttt{Game}_6$ or $\texttt{Game}_7$, perfectly. The definitions of W_6 and W_7 gives us the following equations.

$$\begin{aligned}
\Pr[W_6] &= \Pr\big[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \text{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \text{CHF.SKVer2}(pk_1, sk^*_1, sk_1)] \\
&\quad \wedge \Big[[\bar{m}^* \notin \mathcal{L}_{\bar{m}}] \vee \Big[{}^\exists i \in [1, q_s]\ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r^*_{E2}) = (h_i, r_{E2,i})] \\
&\quad \wedge [(m^*, c^*, \pi^*, r^*_E) \neq (m_i, c_i, \pi_i, r_{E,i})]\Big]\Big]\big] \qquad (4.15) \\
\Pr[W_7] &= \Pr\big[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r^*_E, r^*_{E2}) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \text{CHF.SKVer}(pk_1, sk^*_1)] \wedge [1 \leftarrow \text{CHF.SKVer2}(pk_1, sk^*_1, sk_1)] \\
&\quad \wedge [\bar{m}^* \notin \mathcal{L}_{\bar{m}}]\big] \qquad (4.16)
\end{aligned}$$

By (4.15) and (4.16), we obtain

$$\begin{aligned}
&|\Pr[W_6] - \Pr[W_7]| \\
\leq\ & \Pr\big[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \\
&\wedge [1 \leftarrow \text{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \text{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \\
&\wedge \Big[{}^{\exists}i \in [1, q_s] \ s.t.\ [\bar{m}^* = \bar{m}_i] \wedge [(h^*, r_{E2}^*) = (h_i, r_{E2,i})] \\
&\wedge [(m^*, c^*, \pi^*, r_E^*) \neq (m_i, c_i, \pi_i, r_{E,i})]\Big]\big]
\end{aligned}$$

The last probability is equal to the probability with whom $\mathcal{S}$ wins the game for the strong collision-resistance in HL model w.r.t. the function class $\mathcal{F}^{HtI}_{\Sigma_{\text{SIG}}}(\lambda)$ for Π_{CHF}. □

<u>**Proof of Lemma 4.3.8.**</u> We prove that for any $k \in [1, q_s]$ if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_{7|k-1}] - \Pr[W_{7|k}]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the IND-wLCCA security for Σ_{LPKE}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the IND-wLCCA security for the LPKE scheme Σ_{LPKE}. $\mathcal{CH}$ denotes the challenger in the IND-wLCCA security game. Specifically, $\mathcal{S}$ behaves as follows.

Key-Generation. $\mathcal{S}$ is given an encryption-key ek of Σ_{LPKE}. $\mathcal{S}$ runs $(pk_1, sk_1) \leftarrow \text{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \leftarrow \text{CHF2.Gen}(1^\lambda)$ and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. $\mathcal{S}$ sets pk and sk to $pk := (pk_1, pk_2, ek, crs)$ and $sk := sk_1$, respectively. $\mathcal{S}$ computes $f(pk, sk; r)$, where $r \overset{R}{\leftarrow} \mathcal{R}$, then sends $(pk, f(pk, sk; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets each list of $\mathcal{L}_S$ and $\mathcal{L}_{\bar{m}}$ to $\emptyset$.

Query. We consider the case when $\mathcal{A}$ issues a message $m_i \in \mathcal{M}$ as the i-th query to **Sign**. If $i \geq k + 1$ (resp. $i \leq k - 1$), then $\mathcal{S}$ generates a signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ on m_i as follows.

- $r_{E,i}, r'_{E,i} \overset{U}{\leftarrow} \mathcal{R}_E$, $r'_{E2,i} \overset{U}{\leftarrow} \mathcal{R}_{E2}$, $m'_i \overset{U}{\leftarrow} \mathcal{M}$, $c'_i \overset{U}{\leftarrow} C$, $\pi'_i \overset{U}{\leftarrow} \mathcal{P}$, $\sigma'_i := (c'_i, \pi'_i)$.
- $h'_i := \text{CHF.Eval}(pk_1, m'_i \| \sigma'_i; r'_{E,i})$, $\bar{m}_i := \text{CHF2.Eval}(pk_2, h'_i; r'_{E2,i})$.
- $c_i := \text{LPKE.Enc}(ek, sk_1, \bar{m}_i)$ (resp. $c_i := \text{LPKE.Enc}(ek, 0^{|sk_1|}, \bar{m}_i)$).
- $x_i := (c_i, \bar{m}_i)$, $\pi_i := \mathcal{S}_2(crs, x_i, td)$.
- $\sigma_i := (c_i, \pi_i)$, $h_i := \text{CHF.Eval}(pk_1, m_i \| \sigma_i; r_{E,i})$.
- $r_{E2,i} := \text{CHF2.TC}(pk_2, sk_2, (h'_i, r'_{E2,i}), h_i)$.

Else if $i = k$, then $\mathcal{S}$ generates a signature $(c_k, \pi_k, r_{E,k}, r_{E2,k})$ on m_k in the same manner as the case of $i \geq k + 1$ or $i \leq k - 1$ except that how the ciphertext $c_i = c_k$ is generated. In the case of $i = k$, $\mathcal{S}$ sends $(sk_1, 0^{|sk_1|}, \bar{m}_k)$ to $\mathcal{CH}$ in **Challenge** in IND-wLCCA game for Σ_{LPKE}, then gets a ciphertext c_k.

$\mathcal{S}$ returns the generated signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ to $\mathcal{A}$. $\mathcal{S}$ sets $\mathcal{L}_S$ to $\mathcal{L}_S \cup \{(m_i, c_i, \pi_i, r_{E,i}, r_{E2,i})\}$. $\mathcal{S}$ sets $\mathcal{L}_{\bar{m}}$ to $\mathcal{L}_{\bar{m}} \cup \{\bar{m}_i\}$.

Forgery$(m^*, c^*, \pi^*, r_E^*, r_{E2}^*)$. $\mathcal{S}$ computes $h^* := \text{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r_E^*)$ and $\bar{m}^* := \text{CHF2.Eval}(pk_2, h^*; r_{E2}^*)$. $\mathcal{S}$ issues $(c^*, \bar{m}^*)$ as a query to **Decrypt** in **Query 2** in IND-wLCCA game, then receives sk_1^*. $\mathcal{S}$ outputs $\beta' := 1$ in **Guess** in IND-wLCCA game, when the following condition is satisfied: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, c^*, \pi^*, r_E^*, r_{E2}^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)] \wedge [\bar{m}^* \notin \mathcal{L}_{\bar{m}}]$, where $x^* := (c^*, \bar{m}^*)$.

Let $\beta \in \{0, 1\}$ be the challenge-bit in the IND-wLCCA security game for Π_{LPKE}. It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_{7|k-1}$ (resp. $\mathtt{Game}_{7|k}$) when $\beta = 0$ (resp. $\beta = 1$), and if and only if $W_{7|k-1}$ (resp. $W_{7|k}$) happens, $\mathcal{S}$ outputs $\beta' = 1$. It is also obvious that when $W_{7|k-1}$ or $W_{7|k}$ occurs, the label $\bar{m}^*$ in the query $(c^*, \bar{m}^*)$ to the oracle **Decrypt** in **Query 2** issued by $\mathcal{S}$ satisfies $\bar{m}^* \neq \bar{m}_k$, so the query $(c^*, \bar{m}^*)$ is not a forbidden query. Hence, we obtain $\Pr[W_{7|k-1}] = \Pr[\beta' = 1 \mid \beta = 0]$ and $\Pr[W_{7|k}] = \Pr[\beta' = 1 | \beta = 1]$.

It is obvious that $\Pr[\beta' = \beta] = \Pr[\beta' = 0 \wedge \beta = 0] + \Pr[\beta' = 1 \wedge \beta = 1] = \frac{1}{2}(\Pr[\beta' = 0|\beta = 0] + \Pr[\beta' = 1|\beta = 1]) = \frac{1}{2}(\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0] + 1)$.

Hence, we obtain $\mathsf{Adv}^{IND\text{-}wLCCA}_{\Sigma_{\mathrm{LPKE}},\mathcal{S}} = |2 \cdot \Pr[\beta' = \beta] - 1| = |\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0]| = |\Pr[W_{7|k-1}] - \Pr[W_{7|k}]|$. □

Proof of Lemma 4.3.9. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $\Pr[W_{7|q_s}]$ non-negligible, then we are able to construct a PPT algorithm which breaks the property of hardness of inversion for the leakage-function $f \in \mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$.

We consider a PPT algorithm $\mathcal{S}$. $\mathcal{S}$ behaves as a PPT algorithm in the definition of the leakage-function class $\mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$. $\mathcal{S}$ is given $(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r))$ as inputs, and simulates $\mathtt{Game}_{7|q_s}$ against $\mathcal{A}$. The concrete behavior by $\mathcal{S}$ is the following.

Key-Generation. $\mathcal{S}$ is given $(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r))$, where $(pk_1, sk_1) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{\mathrm{R}}{\leftarrow} \mathsf{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{\mathrm{R}}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$. $\mathcal{S}$ sets pk to (pk_1, pk_2, ek, crs). $\mathcal{S}$ sends $(pk, f(pk_1, pk_2, ek, crs, sk_1; r))$ to $\mathcal{A}$. $\mathcal{S}$ sets each list of $\mathcal{L}_S$ and $\mathcal{L}_{\bar{m}}$ to $\emptyset$.

Query. When $\mathcal{A}$ issues a message $m_i \in \mathcal{M}$ as the i-th query to the signing oracle **Sign**, $\mathcal{S}$ generates a signature $(c_i, \pi_i, r_{E,i}, r_{E2,i})$ on the message as follows.

- $r_{E,i}, r'_{E,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_E$, $r'_{E2,i} \overset{\mathrm{U}}{\leftarrow} \mathcal{R}_{E2}$, $m'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{M}$, $c'_i \overset{\mathrm{U}}{\leftarrow} C$, $\pi'_i \overset{\mathrm{U}}{\leftarrow} \mathcal{P}$, $\sigma'_i := (c'_i, \pi'_i)$.
- $h'_i := \mathsf{CHF.Eval}(pk_1, m'_i \| \sigma'_i; r'_{E,i})$, $\bar{m}_i := \mathsf{CHF2.Eval}(pk_2, h'_i; r'_{E2,i})$.
- $c_i := \mathsf{LPKE.Enc}(ek, sk_1, \bar{m}_i)$, $x_i := (c_i, \bar{m}_i)$, $\pi_i := \mathcal{S}_2(crs, x_i, td)$.
- $\sigma_i := (c_i, \pi_i)$, $h_i := \mathsf{CHF.Eval}(pk_1, m_i \| \sigma_i; r_{E,i})$.
- $r_{E2,i} := \mathsf{CHF2.TC}(pk_2, sk_2, (h'_i, r'_{E2,i}), h_i)$.

$\mathcal{S}$ returns $(c_i, \pi_i, r_{E2,i}, r_{E,i})$ to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\mathcal{L}_S \cup \{(m_i, c_i, \pi_i, r_{E2,i}, r_{E,i})\}$. $\mathcal{L}_{\bar{m}}$ is set to $\mathcal{L}_{\bar{m}} \cup \{\bar{m}_i\}$.

Forgery$(m^*, c^*, \pi^*, r_E^*, r_{E2}^*)$**.** $\mathcal{S}$ sets $h^* := \mathsf{CHF.Eval}(pk_1, m^* \| (c^*, \pi^*); r_E^*)$, $\bar{m}^* := \mathsf{CHF2.Eval}(pk_2, h^*; r_{E2}^*)$ and $sk_1^* := \mathsf{LPKE.Dec}(dk, c^*, \bar{m}^*)$. $\mathcal{S}$ outputs sk_1^*.

It is obvious that $\mathcal{S}$ simulates $\texttt{Game}_{7|q_s}$ against $\mathcal{A}$ perfectly. If $\mathcal{A}$ wins the game $\texttt{Game}_{7|q_s}$, then $\mathcal{S}$ is able to acquire a secret-key sk_1^* such that $[1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)]$. Hence, we obtain

$$\begin{aligned}&\Pr\left[\mathcal{S}\left(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r)\right) \rightarrow sk_1^*\right.\\ &\quad \left.\text{s.t. } [1 \leftarrow \mathsf{CHF.SKVer}\left(pk_1, sk_1^*\right)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}\left(pk_1, sk_1^*, sk_1\right)]\right] = \Pr\left[W_{7|q_s}\right],\end{aligned}$$

where $(pk_1, sk_1) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{\mathrm{R}}{\leftarrow} \mathsf{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{\mathrm{R}}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$.

If we assume that there exists a polynomial function $poly(\lambda)$ such that $\Pr\left[W_{7|q_s}\right] \geq 1/poly(\lambda)$, then we obtain

$$\begin{aligned}&\Pr\left[\mathcal{S}\left(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r)\right) \rightarrow sk_1^*\right.\\ &\quad \left.\text{s.t. } [1 \leftarrow \mathsf{CHF.SKVer}\left(pk_1, sk_1^*\right)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}\left(pk_1, sk_1^*, sk_1\right)]\right] \geq 1/poly(\lambda),\end{aligned}$$

where $(pk_1, sk_1) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF.Gen}(1^\lambda)$, $(pk_2, sk_2) \overset{\mathrm{R}}{\leftarrow} \mathsf{CHF2.Gen}(1^\lambda)$, $(ek, dk) \overset{\mathrm{R}}{\leftarrow} \mathsf{LPKE.Gen}(1^\lambda)$, $(crs, td) \overset{\mathrm{R}}{\leftarrow} \mathcal{S}_1(1^\lambda)$ and $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$.

This contradicts to the hardness of inversion for the leakage-function $f \in \mathcal{F}^{HtI}_{\Sigma_{\mathrm{SIG}}}(\lambda)$. □

4.4 An Instantiation Secure under the DLIN Assumption

As a concrete construction for the chameleon hash function Σ_{CHF}, we adopt the chameleon hash function $\Pi_{\mathrm{CHF},n}$ given in Fig. 4.1. For $\Pi_{\mathrm{CHF},n}$, we obtain Theorem 4.4.1, Theorem 4.4.2, and Theorem 4.4.3, whose proofs are given below.

Theorem 4.4.1. *For any $n \in \mathbb{N}$, $\Pi_{\mathrm{CHF},n}$ is HtC-SK under the DL assumption.*

Theorem 4.4.2. *For any $n \in \mathbb{N}$, $\Pi_{\mathrm{CHF},n}$ is random trapdoor collision.*

Theorem 4.4.3. *For any chameleon hash function Π_{CHF2}, any LPKE scheme Π_{LPKE}, any NIZK scheme Π_{NIZK}, and any integer $n \in \mathbb{N}$, $\Pi_{\mathrm{CHF},n}$ is strongly collision-resistant in HL model w.r.t. the function class $\mathcal{F}^{HtI}_{\Pi_{\mathrm{SIG}}}(\lambda)$ under the collision-resistance of the hash function $J : \{0,1\}^* \rightarrow \mathbb{Z}_p \setminus \{0\}$ and the DL assumption, where Π_{SIG} denotes the instantiation of the signature scheme Σ_{SIG} by Π_{CHF}, Π_{CHF2}, Π_{LPKE} and Π_{NIZK}.*

As a concrete construction for the chameleon hash function Σ_{CHF2}, we adopt the chameleon hash function $\Pi_{\mathrm{CHF},1}$ which is $\Pi_{\mathrm{CHF},n}$ in Fig. 4.1 with $n = 1$. The following corollary is obtained by Theorem 4.4.3, obviously.

Corollary 4.4.1. *For any $n \in \mathbb{N}$, $\Pi_{\mathrm{CHF},n}$ is strongly collision-resistant under the collision-resistance of the hash function $J : \{0,1\}^* \rightarrow \mathbb{Z}_p \setminus \{0\}$ and the DL assumption.*

Thus, the random trapdoor collision and strong collision-resistance of the CHF scheme $\Pi_{\mathrm{CHF},1}$ are guaranteed by Theorem 4.4.2 and Corollary 4.4.1, respectively.

As a concrete construction for the LPKE scheme Σ_{LPKE}, we adopt $\Pi_{\mathrm{LPKE},l}$ given in Fig. 5.10. The LPKE scheme is a modification of the LPKE scheme by Camenisch et

```
CHF.Gen$(1^\lambda, 1^n)$:
    $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. $x_1, \cdots, x_n, a_1, \cdots, a_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $g_1 := g^{a_1}, \cdots, g_n := g^{a_n}.y := \prod_{i=1}^n g_i^{x_i}$.
    Return $pk := (p, \mathbb{G}, g_1, \cdots, g_n, y)$ and $sk := (x_1, \cdots, x_n)$.
CHF.Eval$(pk, m; \mathbf{r})$:
    $\mathbf{r} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p^n$, where $\mathbf{r}$ is parsed as $(r_1, \cdots, r_n)$. Return $\left(y \cdot \Pi_{i=1}^n g_i^{r_i}\right)^{J(m)}$.
CHF.TC$(pk, sk, (m, \mathbf{r}), m')$:
    $sk \in \mathbb{Z}_p^n$ is parsed as $(x_1, \cdots, x_n)$. $\mathbf{r} \in \mathbb{Z}_p^n$ is parsed as $(r_1, \cdots, r_n)$.
    For $i \in [1, n]$, $r'_i := J(m)(x_i - r_i)/J(m') - x_i$. Return $\mathbf{r}' := (r'_1, \cdots, r'_n)$.
CHF.SKVer$(pk, sk^*)$:
    $sk^* \in \mathbb{Z}_p^n$ is parsed as $(x_1^*, \cdots, x_n^*)$. Return 1, if $\left[y = \prod_{i=1}^n g_i^{x_i^*}\right]$. Return 0, otherwise;
CHF.SKVer2$(pk, sk^*, sk')$:
    $sk^* \in \mathbb{Z}_p^n$ is parsed as $(x_1^*, \cdots, x_n^*)$. $sk' \in \mathbb{Z}_p^n$ is parsed as $(x'_1, \cdots, x'_n)$.
    Return 1, if $\left[\bigwedge_{i=1}^n \left[x_i^* = x'_i\right]\right]$. Return 0, otherwise;
```

Figure 4.1: Construction of CHF Scheme $\Pi_{\mathrm{CHF},n}$, where $J : \{0,1\}^* \to \mathbb{Z}_p \setminus \{0\}$ is a collision-resistant hash function.

al. [CCS09] which is IND-LCCA secure[3] under the DLIN assumption and the collision-resistance of hash function. Faust et al. [FHN+12] modifies the scheme by Camenisch et al. to get the LPKE scheme $\Pi_{\mathrm{LPKE},l}$ which achieves a weaker security, i.e., IND-wLCCA, but encrypts a plaintext of arbitrary length. Thus,

Theorem 4.4.4. *For any $l \in \mathbb{N}$, $\Pi_{\mathrm{LPKE},l}$ is IND-wLCCA under the collision-resistance of the hash function $H_{CL} : \{0,1\}^* \to \mathbb{Z}_p$ and the DLIN assumption.*

As a concrete construction for the non-interactive zero-knowledge proof Σ_{NIZK}, we adopt the Groth-Sahai proof Π_{NIZK} in [GS08] whose soundness and zero-knowledge are guaranteed under the DLIN assumption.

By the schemes $\Pi_{\mathrm{CHF},n}, \Pi_{\mathrm{CHF2}}, \Pi_{\mathrm{NIZK}}$ and $\Pi_{\mathrm{LPKE},n\lambda}$, where λ denotes the integer in the security parameter 1^λ of $\Pi_{\mathrm{CHF},n}$, our concrete signature scheme Π_{SIG} is constructed. Hereafter, for $i \in [1, n]$ and $j \in [1, \lambda]$, the j-th bit of $x_i \in \mathbb{Z}_p$ in $\Pi_{\mathrm{CHF},n}$ is denoted by $x_{ij} \in \{0,1\}$, and the prime and group in $\Pi_{\mathrm{LPKE},n\lambda}$ are written as $\hat{p}$ and $\hat{\mathbb{G}}$, respectively. By the signing algorithm of Π_{SIG}, a ciphertext $\mathbf{C}$ and a proof π are generated as follows.

The ciphertext $\mathbf{C}$ is generated by running $\mathbf{C} \leftarrow$ LPKE.Enc$(ek, (x_1, \cdots, x_n), \bar{m})$, where LPKE.Enc is the encryption algorithm of $\Pi_{\mathrm{LPKE},n\lambda}$. $\mathbf{C}$ is parsed as $\{\mathbf{c}_{ij}\}_{i\in[1,n],j\in[1,\lambda]}$. $\mathbf{c}_{ij}$ is parsed as $(\mathbf{y}_{ij}, z_{ij} \in \hat{\mathbb{G}}, c_{ij} \in \hat{\mathbb{G}})$. $\mathbf{y}_{ij}$ is parsed as $(y_{ij,1}, y_{ij,2}, y_{ij,3}) \in \hat{\mathbb{G}}^3$.

By using the proof-generation algorithm of the NIZK scheme Π_{NIZK}, we generate the

[3]IND-LCCA is stronger security notion than IND-wLCCA. For the details, refer to [FHN+12].

```
LPKE.Gen($1^\lambda, 1^l$):
    $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$. $a_1, \cdots, a_l, b_1, b_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $\hat{g}_0 := g, \hat{g}_1 := g^{b_1}, \hat{g}_2 := g^{b_2}, g_1 := g^{a_1}, \cdots, g_l := g^{a_l}$.
    $u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $d_1 := \hat{g}_0^{u_1} \cdot \hat{g}_1^{u_2}, d_2 := \hat{g}_0^{u_1} \cdot \hat{g}_2^{u_3}$,
    $e_1 := \hat{g}_0^{v_1} \cdot \hat{g}_1^{v_2}, e_2 := \hat{g}_0^{v_1} \cdot \hat{g}_2^{v_3}, h_1 := \hat{g}_0^{w_1} \cdot \hat{g}_1^{w_2}, h_2 := \hat{g}_0^{w_1} \cdot \hat{g}_2^{w_3}$.
    $ek := (p, \mathbb{G}, \hat{g}_0, \hat{g}_1, \hat{g}_2, g_1, \cdots, g_l, d_1, d_2, e_1, e_2, h_1, h_2)$. $dk := (u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3)$.
    Return($ek, dk$).
LPKE.Enc($ek, x \in \{0,1\}^l, L \in \{0,1\}^*$):
    For $i \in [1, l]$, the $i$-th bit of $x \in \{0,1\}^l$ is denoted by $x_i \in \{0,1\}$.
    For every $i \in [1, l]$, do:
        $r_i, s_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $\mathbf{y}_i := (y_{i,1}, y_{i,2}, y_{i,3}) := (\hat{g}_0^{r_i+s_i}, \hat{g}_1^{r_i}, \hat{g}_2^{s_i})$. $z_i := h_1^{r_i} \cdot h_2^{s_i} \cdot g_i^{x_i}$.
        $c_i := (d_1 \cdot e_1^{t_i})^{r_i} \cdot (d_2 \cdot e_2^{t_i})^{s_i}$, where $t_i := H_{CL}(\mathbf{y}_i, z_i, L)$. $\mathbf{c}_i := (\mathbf{y}_i, z_i, c_i)$.
    Return $\mathbf{C} := \{\mathbf{c}_i\}_{i \in [1,l]}$.
LPKE.Dec($ek, dk, \mathbf{C}, L$):
    For every $i \in [1, l]$, do:
        $\tilde{c}_i := y_{i,1}^{u_1+t_i v_1} \cdot y_{i,2}^{u_2+t_i v_2} \cdot y_{i,3}^{u_3+t_i v_3}$, where $t_i := H_{CL}(\mathbf{y}_i, z_i, L)$.
        If $\tilde{c}_i \neq c_i$, then return $\perp$.
        Else if $z_i/(y_{i,1}^{w_1} \cdot y_{i,2}^{w_2} \cdot y_{i,3}^{w_3}) = g_i$, then $x'_i := 1$. Else, then $x'_i := 0$.
        The $i$-th bit of $x'$ is set as $x'_i$.
    Return $x' \in \{0,1\}^l$.
```

Figure 4.2: Construction of LPKE Scheme $\Pi_{\mathrm{LPKE},l}$, where $H_{CL} : \{0,1\}^* \to \mathbb{Z}_p$ is a collision-resistant hash function.

proof π. Actually, the proof π is a proof which proves that

$$\exists \{r_{ij} \in \mathbb{Z}_{\hat{p}}, s_{ij} \in \mathbb{Z}_{\hat{p}}, x_{ij} \in \{0,1\}\}_{i \in [1,n], j \in [1,\lambda]} \textit{ such that}$$
$$\left[\prod_{i=1}^{n} \prod_{j=1}^{\lambda} g_i^{2^{j-1} \cdot x_{ij}} = y\right] \bigwedge_{i \in [1,n], j \in [1,\lambda]} \left[\left[\hat{g}_0^{r_{ij}+s_{ij}} = y_{ij,1}\right] \wedge \left[\hat{g}_1^{r_{ij}} = y_{ij,2}\right] \wedge \left[\hat{g}_2^{s_{ij}} = y_{ij,3}\right]\right.$$
$$\left. \wedge \left[h_1^{r_{ij}} \cdot h_2^{s_{ij}} \cdot g_{ij}^{x_{ij}} = z_{ij}\right] \wedge \left[(d_1 \cdot e_1^{t_{ij}})^{r_{ij}} \cdot (d_2 \cdot e_2^{t_{ij}})^{s_{ij}} = c_{ij}\right] \wedge \left[x_{ij}(1 - x_{ij}) = 0\right]\right],$$

where $y = \prod_{i=1}^{n} g_i^{x_i} \in \mathbb{G}$.

Proof of Theorem 4.4.1. We prove the theorem by proving that if we assume that there is a PPT adversary $\mathcal{A}$ which breaks the HtC-SK property for $\Pi_{\mathrm{CHF},n}$, then we are able to construct a PPT algorithm $\mathcal{S}$ which breaks the DL assumption. We prove the theorem for the case that $n \geq 2$. The theorem for the case that $n = 1$ can be proven in the same manner.

$\mathcal{S}$ is given $(p, \mathbb{G}, g, g^a)$ as an instance of the discrete logarithm problem. Then, $\mathcal{S}$ behaves as follows: Chooses $j \overset{\mathrm{U}}{\leftarrow} [1, n]$ and sets $g_j := g^a$. For every $i \in [1, n]$, chooses $x_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, then sets $\mathbf{x} := (x_1, \cdots, x_n)$. For every $i \in [1, n] \setminus \{j\}$, chooses $a_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, then sets $g_i := g^{a_i}$. Sets $y := \prod_{i \in [1,n]} g_i^{x_i}$.

$\mathcal{S}$ gives $pk := (p, \mathbb{G}, g, g_1, \cdots, g_n, y)$ to $\mathcal{A}$. After that, $\mathcal{S}$ receives $\mathbf{x}^* := (x_1^*, \cdots, x_n^*) \in \mathbb{Z}_p^n$ sent by $\mathcal{A}$. Since we are considering a PPT adversary $\mathcal{A}$ breaking the HtC-SK property for $\Pi_{\mathrm{CHF},n}$, $\mathbf{x}^*$ satisfies the relation $[\mathbf{x}^* \neq \mathbf{x}] \wedge [\prod_{i \in [1,n]} g_i^{x_i^*} = \prod_{i \in [1,n]} g_i^{x_i}]$.

Hence, we obtain $g_j^{x_j^*-x_j} = \prod_{i\in[1,n]\setminus\{j\}} g_i^{x_i-x_i^*}$ and $g_j = g^{(\sum_{i\in[1,n]\setminus\{j\}} a_i(x_i-x_i^*))/(x_j^*-x_j)} = g^a$.

$\mathcal{S}$ outputs $a = (\sum_{i\in[1,n]\setminus\{j\}} a_i(x_i - x_i^*))/(x_j^* - x_j)$. Note that $\Pr[x_j^* \neq x_j] \geq 2/n$, where $n \geq 2$. □

Proof of Theorem 4.4.2. We consider a key-pair (pk, sk), a message $m \in \mathcal{M}$, and a message $m' \in \mathcal{M}$. pk and sk are parsed as $pk = (p, \mathbb{G}, g_1, \cdots, g_n, y)$ and $sk = (x_1, \cdots, x_n)$, respectively.

$P(\hat{\mathbf{r}})$ denotes the probability that the randomness $\hat{\mathbf{r}}$ is chosen as the randomness used to compute the hash value for the message m. Hence, $P(\hat{\mathbf{r}}) = \Pr[\hat{\mathbf{r}} = \mathbf{r} \mid \mathbf{r} \overset{U}{\leftarrow} \mathbb{Z}_p^n] = \Pr[\bigwedge_{i\in[1,n]}[\hat{r}_i = r_i] \mid \mathbf{r} \overset{U}{\leftarrow} \mathbb{Z}_p^n] = \prod_{i\in[1,n]} \Pr[\hat{r}_i = r_i \mid r_i \overset{U}{\leftarrow} \mathbb{Z}_p] = 1/p^n$.

$P(\hat{\mathbf{r}'})$ denotes the probability that CHF.TC$(pk, sk, (m, \mathbf{r}), m')$ outputs $\hat{\mathbf{r}'} \in \mathbb{Z}_p^n$, where the randomness $\mathbf{r} \in \mathbb{Z}_p^n$ is chosen uniformly at random. Hence,
$P(\hat{\mathbf{r}'}) = \Pr[\hat{\mathbf{r}'} = \text{CHF.TC}(pk_1, sk_1, (m, \mathbf{r}), m') \mid \mathbf{r} \overset{U}{\leftarrow} \mathbb{Z}_p^n] = \Pr[\bigwedge_{i\in[1,n]}[\hat{r}'_i = J(m)(x_i - r_i)/J(m') - x_i] \mid \mathbf{r} \overset{U}{\leftarrow} \mathbb{Z}_p^n] = \prod_{i\in[1,n]} \Pr[\hat{r}'_i = J(m)(x_i - r_i)/J(m') - x_i \mid r_i \overset{U}{\leftarrow} \mathbb{Z}_p] = 1/p^n$.

Hence, $P(\hat{\mathbf{r}}) = P(\hat{\mathbf{r}'}) = 1/p^n$. Therefore, for any (pk, sk) and any $m, m' \in \mathcal{M}$, $\mathbf{r} \overset{U}{\leftarrow} \mathcal{R}$ and $\mathbf{r}' :=$ CHF.Eval$(pk, sk, (m, \mathbf{r}), m')$ distribute identically. □

Proof of Theorem 4.4.3. We prove the theorem by the argument of game-transformation. We use four games $\mathtt{Game}_0$, $\mathtt{Game}_1$, $\mathtt{Game}_2$ and $\mathtt{Game}_3$. Each game is defined as follows.

$\mathtt{Game}_0$. $\mathtt{Game}_0$ is the game of strong collision-resistance for the CHF scheme $\Pi_{\text{CHF},n}$ in the auxiliary leakage model w.r.t. the function class $\mathcal{F}_{\Pi_{\text{SIG}}}^{HtI}(\lambda)$. Concretely, the game is the following.

$\mathcal{CH}$ runs $(pk_1, sk_1) \leftarrow$ CHF.Gen(1^λ), $(pk_2, sk_2) \leftarrow$ CHF2.Gen(1^λ), $(ek, dk) \leftarrow$ LPKE.Gen(1^λ) and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk_1 is parsed as $(p, \mathbb{G}, g_1, \cdots, g_n, y)$. sk_1 is parsed as $(x_1, \cdots, x_n)$. pk is set as (pk_1, pk_2, ek, crs). sk is set as sk_1. $\mathcal{CH}$ sets r as $r \overset{R}{\leftarrow} \mathcal{R}$, where $\mathcal{R}$ is the randomness space of a leakage function $f \in \mathcal{F}_{\Pi_{\text{SIG}}}^{HtI}(\lambda)$, then computes $f(pk, sk; r)$. $\mathcal{CH}$ sends $(pk, sk_2, dk, td, f(pk, sk; r))$ to $\mathcal{A}$. Then, $(m, \mathbf{r})$ and $(m', \mathbf{r}')$, where $m, m' \in \{0, 1\}^*$ and $\mathbf{r}, \mathbf{r}' \in \mathbb{Z}_p^n$, are sent to $\mathcal{CH}$ by $\mathcal{A}$. $\mathbf{r}$ and $\mathbf{r}'$ are parsed as $(r_1, \cdots, r_n)$ and $(r'_1, \cdots, r'_n)$, respectively. $\mathcal{A}$ is said to win the game, if the following condition is satisfied:

$$[[m \neq m'] \vee [[m = m'] \wedge [\mathbf{r} \neq \mathbf{r}']]] \wedge \left[\left(y \cdot \prod_{i=1}^{n} g_i^{r_i}\right)^{J(m)} = \left(y \cdot \prod_{i=1}^{n} g_i^{r'_i}\right)^{J(m')}\right].$$

$\mathtt{Game}_1$. $\mathtt{Game}_1$ is the same as $\mathtt{Game}_0$ except that the winning condition by $\mathcal{A}$ is changed to the following one: $[[[m \neq m'] \wedge [J(m) \neq J(m')] \wedge [[\mathbf{x} \neq \mathbf{x}^*] \vee [\mathbf{x} = \mathbf{x}^*]]] \vee [[m = m'] \wedge [\mathbf{r} \neq \mathbf{r}']]] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]$, where, for $i \in [1, n]$, $x_i^* := (J(m)r_i - J(m')r'_i)/(J(m') - J(m))$, $\mathbf{x}^* := (x_1^*, \cdots, x_n^*)$ and $\mathbf{x} := (x_1, \cdots, x_n)$.

$\mathtt{Game}_2$. $\mathtt{Game}_2$ is the same as $\mathtt{Game}_1$ except that the winning condition by $\mathcal{A}$ is changed to the following one: $[[[m \neq m'] \wedge [J(m) \neq J(m')] \wedge [\mathbf{x} = \mathbf{x}^*]] \vee [[m = m'] \wedge [\mathbf{r} \neq \mathbf{r}']]] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]$.

$\mathtt{Game}_3$. $\mathtt{Game}_3$ is the same as $\mathtt{Game}_2$ except that the winning condition by $\mathcal{A}$ is changed to the following one: $[m = m'] \wedge [\mathbf{r} \neq \mathbf{r}'] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]$.

W_i, where $i \in \{0, 1, 2, 3\}$, denotes the event that $\mathcal{A}$ wins the game $\mathtt{Game}_i$. It holds that

$$\Pr[W_0] \leq |\Pr[W_0] - \Pr[W_1]| + |\Pr[W_1] - \Pr[W_2]| + |\Pr[W_2] - \Pr[W_3]|.$$

By the above inequality and the following lemmas, Theorem 4.4.3 is proven.

Lemma 4.4.1. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible, if* $J : \{0,1\}^* \to \mathbb{Z}_p \setminus \{0\}$ *is a collision-resistant hash function.*

Lemma 4.4.2. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible, if the discrete logarithm assumption holds.*

Lemma 4.4.3. $|\Pr[W_2] - \Pr[W_3]|$ *is negligible under the hard-to-invert property of the function* $f \in \mathcal{F}_{\Pi_{\mathrm{SIG}}}^{HtI}(\lambda)$.

Lemma 4.4.4. $\Pr[W_3]$ *is negligible, if the discrete logarithm assumption holds.*

□

<u>**Proof of Lemma 4.4.1.**</u> We prove that if there is a PPT $\mathcal{A}$ which makes $|\Pr[W_0] - \Pr[W_1]|$ non-negligible, we can construct a PPT $\mathcal{S}$ which breaks the collision-resistance of the hash function $J : \{0,1\}^* \to \mathbb{Z}_p \setminus \{0\}$. Let us consider a PPT $\mathcal{S}$ which behaves as follows.

$\mathcal{S}$ randomly generates (pk_1, sk_1), (pk_2, sk_2), (ek, dk) and (crs, td). pk_1 and sk_1 are parsed as $(p, \mathbb{G}, g_1, \cdots, g_n, y)$ and $(x_1, \cdots, x_n)$, respectively. pk and sk are set as (pk_1, pk_2, ek, crs) and sk_1, respectively. $\mathcal{S}$ sets r as $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, where $\mathcal{R}$ is the randomness space of a leakage function $f \in \mathcal{F}_{\Pi_{\mathrm{SIG}}}^{HtI}(\lambda)$, then computes $f(pk, sk; r)$. $\mathcal{S}$ sends $(pk, sk_2, dk, td, f(pk, sk; r))$ to $\mathcal{A}$. Then, $\mathcal{S}$ receives $(m, \mathbf{r})$ and $(m', \mathbf{r}')$ from $\mathcal{A}$. If $[m \neq m'] \wedge [J(m) = J(m')] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]$, then $\mathcal{S}$ outputs (m, m'). We obtain

$$\begin{aligned} |\Pr[W_0] - \Pr[W_1]| &\leq \Pr[[m \neq m'] \wedge [J(m) = J(m')] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]] \\ &= \Pr[\mathcal{S}(\cdot) \to (m, m') \ s.t.\ [m \neq m'] \wedge [J(m) = J(m')]]. \end{aligned}$$

If we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_0] - \Pr[W_1]|$ non-negligible, $\mathcal{S}$ is able to break the collision-resistance property of the hash function J. □

<u>**Proof of Lemma 4.4.2.**</u> We prove that if there is a PPT $\mathcal{A}$ which makes $|\Pr[W_1] - \Pr[W_2]|$ non-negligible, a PPT $\mathcal{S}$ which breaks the HtC-SK property for $\Pi_{\mathrm{CHF},n}$ can be constructed. Let us consider a PPT $\mathcal{S}$ which behaves as follows.

$\mathcal{S}$ is given the keys (pk_1, sk_1) of $\Pi_{\mathrm{CHF},n}$. pk_1 and sk_1 are parsed as $(p, \mathbb{G}, g_1, \cdots, g_n, y)$ and $(x_1, \cdots, x_n)$, respectively. $\mathcal{S}$ randomly generates (pk_2, sk_2), (ek, dk) and (crs, td). pk and sk are set as (pk_1, pk_2, ek, crs) and sk_1, respectively. $\mathcal{S}$ sets r as $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then computes $f(pk, sk; r)$. $\mathcal{S}$ sends $(pk, sk_2, dk, td, f(pk, sk; r))$ to $\mathcal{A}$. Then, $\mathcal{S}$ receives $(m, \mathbf{r})$ and $(m', \mathbf{r}')$ from $\mathcal{A}$. $\mathcal{S}$ computes $x_i^* := (J(m) \cdot r_i - J(m') \cdot r'_i)/(J(m') - J(m))$ for $i \in [1, n]$ and sets

$\mathbf{x}^* := (x_1^*, \cdots, x_n^*)$ and $\mathbf{x} := (x_1, \cdots, x_n)$. If $[m \neq m'] \wedge [J(m) \neq J(m')] \wedge [\mathbf{x} \neq \mathbf{x}^*] \wedge [(y \cdot \prod_{i=1}^{n} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i=1}^{n} g_i^{r'_i})^{J(m')}]$, then $\mathcal{S}$ outputs $\mathbf{x}^*$.

We obtain $\prod_{i\in[1,n]} g_i^{J(m)\cdot(x_i+r_i)} = \prod_{i\in[1,n]} g_i^{J(m')\cdot(x_i+r'_i)}$, $g^{\sum_{i\in[1,n]} a_i \cdot J(m)\cdot(x_i+r_i)} = g^{\sum_{i\in[1,n]} a_i \cdot J(m')\cdot(x_i+r'_i)}$, and $g^{\sum_{i\in[1,n]} a_i \cdot x_i} = g^{\sum_{i\in[1,n]} \frac{J(m')\cdot r'_i - J(m)\cdot r_i}{J(m)-J(m')}}$. In the transition from the first equation to the second one, we used the fact that for every $g_i \in \mathbb{G}$ where $i \in [1, n]$, there exists an integer $a_i \in \mathbb{Z}_p$ such that $g_i = g^{a_i}$.

Likewise, we obtain the following equations.

$$\prod_{i\in[1,n]} g_i^{J(m)\cdot(x_i^*+r_i)} = \prod_{i\in[1,n]} g_i^{J(m')\cdot(x_i^*+r'_i)}$$
$$g^{\sum_{i\in[1,n]} a_i \cdot J(m)\cdot(x_i^*+r_i)} = g^{\sum_{i\in[1,n]} a_i \cdot J(m')\cdot(x_i^*+r'_i)}$$
$$g^{\sum_{i\in[1,n]} a_i \cdot x_i^*} = g^{\sum_{i\in[1,n]} \frac{J(m')\cdot r'_i - J(m)\cdot r_i}{J(m)-J(m')}}$$

Hence, we obtain $g^{\sum_{i\in[1,n]} a_i \cdot x_i} = g^{\sum_{i\in[1,n]} a_i \cdot x_i^*}$, and $y = \prod_{i\in[1,n]} g_i^{x_i} = \prod_{i\in[1,n]} g_i^{x_i^*}$.

As a result, we obtain

$$\begin{aligned}
&|\Pr[W_1] - \Pr[W_2]| \\
\leq\ & \Pr\left[[m \neq m'] \wedge [J(m) \neq J(m')] \wedge [\mathbf{x} \neq \mathbf{x}^*] \wedge \left[(y \cdot \prod_{i\in[1,n]} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i\in[1,n]} g_i^{r'_i})^{J(m')}\right]\right] \\
=\ & \Pr\left[\mathcal{S}(p, \mathbb{G}, g_1, \cdots, g_n, y, \mathbf{x}) \to \mathbf{x}^* \ s.t. \left[y = \prod_{i\in[1,n]} g_i^{x_i^*}\right] \wedge [\mathbf{x} \neq \mathbf{x}^*]\right].
\end{aligned}$$

If we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_1]-\Pr[W_2]|$ non-negligible, $\mathcal{S}$ is able to break the property of HtC-SK for $\Pi_{\mathrm{CHF},n}$. We have already proven Theorem 4.4.1 which says that the property of HtC-SK for $\Pi_{\mathrm{CHF},n}$ can be proven under the discrete logarithm assumption. Hence, $|\Pr[W_1]-\Pr[W_2]|$ is negligible under the DL assumption. □

Proof of Lemma 4.4.3. Let f be a leakage function $f \in \mathcal{F}_{\Pi_{\mathrm{SIG}}}^{HtI}(\lambda)$. We prove that if there is a PPT $\mathcal{A}$ which makes $|\Pr[W_2] - \Pr[W_3]|$ non-negligible, we can construct a PPT $\mathcal{S}$ which breaks the hardness of inversion for the function. Let us consider a PPT $\mathcal{S}$ which behaves as follows.

$\mathcal{S}$ is given $(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, pk_2, ek, crs, sk_1; r))$, where (pk_1, sk_1), (pk_2, sk_2), (ek, dk) and (crs, td) are randomly generated and $r \in \mathcal{R}$ is randomly chosen. pk_1 and sk_1 are parsed as $(p, \mathbb{G}, g_1, \cdots, g_n, y)$ and $(x_1, \cdots, x_n)$, respectively. $\mathcal{S}$ sends $(pk_1, pk_2, ek, crs, f(pk_1, pk_2, ek, crs, sk_1; r))$ to $\mathcal{A}$. Then, $\mathcal{S}$ receives $(m, \mathbf{r})$ and $(m', \mathbf{r}')$ from $\mathcal{A}$. $\mathcal{S}$ computes $x_i^* := (J(m)r_i - J(m')r'_i)/(J(m') - J(m))$ for $i \in [1, n]$ and sets $sk_1^* := (x_1^*, \cdots, x_n^*)$. $\mathcal{S}$ outputs sk_1^*. We obtain

$$\begin{aligned}
&|\Pr[W_2] - \Pr[W_3]| \\
\leq\ & \Pr[[m \neq m'] \wedge [J(m) \neq J(m')] \wedge [sk_1 = sk_1^*] \wedge [(y \cdot \prod_{i\in[1,n]} g_i^{r_i})^{J(m)} = (y \cdot \prod_{i\in[1,n]} g_i^{r'_i})^{J(m')}]] \\
=\ & \Pr[\mathcal{S}(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, sk_1; r)) \to sk_1^* \ s.t. \ [sk_1^* = sk_1]] \\
=\ & \Pr[\mathcal{S}(pk_1, pk_2, ek, crs, sk_2, dk, td, f(pk_1, sk_1; r)) \to sk_1^* \\
& \quad s.t. \ [1 \leftarrow \mathsf{CHF.SKVer}(pk_1, sk_1^*)] \wedge [1 \leftarrow \mathsf{CHF.SKVer2}(pk_1, sk_1^*, sk_1)].
\end{aligned}$$

If we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_2]-\Pr[W_3]|$ non-negligible, $\mathcal{S}$ breaks the hardness of inversion of the function $f \in \mathcal{F}^{HtI}_{\Pi_{\mathrm{SIG}}}(\lambda)$. $\square$

Proof of Lemma 4.4.4. We prove that if there is a PPT $\mathcal{A}$ which makes $\Pr[W_4]$ non-negligible, we can construct a PPT $\mathcal{S}$ which breaks the n-representation assumption [Bra93, BGG94]. We give the definition of the assumption below.

Definition 22. *We say that n-representation assumption holds if for every PPT $\mathcal{A}$,*

$$\begin{aligned}&\Pr\left[\mathcal{A}(p,\mathbb{G},g_1,\cdots,g_n) \to ((x_1,\cdots,x_n),(x'_1,\cdots,x'_n))\right.\\ &\left. s.t.\ [(x_1,\cdots,x_n) \neq (x'_1,\cdots,x'_n)] \wedge \left[\prod_{i\in[1,n]} g_i^{x_i} = \prod_{i\in[1,n]} g_i^{x'_i}\right]\right]\end{aligned}$$

is negligible, where $(p,\mathbb{G}) \overset{\mathrm{R}}{\leftarrow} \mathcal{G}(1^\lambda)$, $g_1,\cdots,g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$ and $x_1,\cdots,x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$.

Validity of the assumption is guaranteed by the following theorem [Bra93, BGG94].

Theorem 4.4.5. *n-representation assumption holds under the DL assumption.*

Let us consider a PPT $\mathcal{S}$ which behaves as follows. $\mathcal{S}$ is given $(p,\mathbb{G},g_1,\cdots,g_n)$ as an instance of the n-representation problem. $\mathcal{S}$ chooses $x_1,\cdots,x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, and sets $y := \prod_{i\in[1,n]} g_i^{x_i}$. Then, $\mathcal{S}$ sets pk_1 and sk_1 as $(p,\mathbb{G},g_1,\cdots,g_n,y)$ and $(x_1,\cdots,x_n)$, respectively. $\mathcal{S}$ randomly generates (pk_2,sk_2), (ek,dk) and (crs,td). pk and sk are set as (pk_1,pk_2,ek,crs) and sk_1, respectively. $\mathcal{S}$ sets r as $r \overset{\mathrm{R}}{\leftarrow} \mathcal{R}$, then computes $f(pk,sk;r)$. $\mathcal{S}$ sends $(pk,sk_2,dk,td,f(pk,sk;r))$ to $\mathcal{A}$. Then, $\mathcal{S}$ receives $(m,\mathbf{r})$ and $(m',\mathbf{r}')$ from $\mathcal{A}$. If $[m=m'] \wedge [\mathbf{r}\neq\mathbf{r}'] \wedge [(y\cdot\prod_{i=1}^n g_i^{r_i})^{J(m)} = (y\cdot\prod_{i=1}^n g_i^{r'_i})^{J(m')}]$, then $\mathcal{S}$ outputs $(\mathbf{r},\mathbf{r}')$. We obtain $\left(y\cdot\prod_{i\in[1,n]} g_i^{r_i}\right)^{J(m)} = \left(y\cdot\prod_{i\in[1,n]} g_i^{r'_i}\right)^{J(m')} = \left(y\cdot\prod_{i\in[1,n]} g_i^{r'_i}\right)^{J(m)}$. Hence, $\prod_{i\in[1,n]} g_i^{r_i} = \prod_{i\in[1,n]} g_i^{r'_i}$. Therefore, we obtain

$$\begin{aligned}\Pr[W_4] &= \Pr\left[[m=m'] \wedge [\mathbf{r}\neq\mathbf{r}'] \wedge \left[\left(y\cdot\prod_{i\in[1,n]} g_i^{r_i}\right)^{J(m)} = \left(y\cdot\prod_{i\in[1,n]} g_i^{r'_i}\right)^{J(m')}\right]\right]\\ &= \Pr\left[\mathcal{S}(p,\mathbb{G},g_1,\cdots,g_n) \to (\mathbf{r},\mathbf{r}')\ s.t.\ [\mathbf{r}\neq\mathbf{r}'] \wedge \left[\prod_{i\in[1,n]} g_i^{r_i} = \prod_{i\in[1,n]} g_i^{r'_i}\right]\right].\end{aligned}$$

If we assume that there is a PPT adversary $\mathcal{A}$ which makes $\Pr[W_4]$ non-negligible, $\mathcal{S}$ is able to break the n-representation assumption. $\square$

4.5 Conclusion for Chapter 4

In this work, we generically constructed a digital signature scheme which is strongly existentially unforgeable and resilient to polynomially hard-to-invert leakage which can be chosen dependently on the public-key pk. Then, we instantiated it under the DLIN assumption in the standard model. Our result is meaningful because of the following two independent respects.

Firstly, our instantiation of signature is the first one resilient to polynomially hard-to-invert leakage (dependent on the public-key) under standard assumptions. Note that the existing known signature schemes secure in HL model are either one which is resilient to exponentially hard-to-invert leakage which can be dependent on the public-key under standard assumptions such as [FHN^{+}12] or one which is polynomially hard-to-invert leakage which cannot be dependent on the public-key under standard (resp. strong) assumptions [YYH12] (resp. [WMHT16]).

Secondly, our instantiation of signature is the first one which simultaneously achieves the strong unforgeability and hard-to-invert leakage-resilience. Note that every existing known signature scheme [FHN^{+}12, YYH12, WMHT16] is weakly unforgeable.

Related to this work, there are some open problems.

For instance, one of them can be presenting a more efficient scheme. Our signature scheme degrades its efficiency because of the non-interactive zero-knowledge proof (NIZK) as one of the building blocks. Thus, presenting a more efficient scheme which is not based on such inefficient schemes is an open problem.

Chapter 5

ABS/IBS Schemes with Hard-to-Invert Leakage-Resilience

5.1 Introduction for Chapter 5

5.1.1 Background

Identity-Based Signature (IBS). The concept of IBS was presented by Shamir at Crypto'84 [Sha84]. IBS is a generalization of digital signature. In IBS systems, each user whose identity information (or ID) is a bit-string $ID \in \{0,1\}^*$ receives a secret-key associated with the ID which was generated by a trusted authority. By using the secret-key, the user can generate a valid signature on any message associated with the ID. IBS is required to be existentially unforgeable. Informally, IBS is said to be existentially unforgeable when every PPT adversary cannot find with a non-negligible probability a message m^*, a signature σ^* and an ID ID^* called the *target* ID such that σ^* is a correct signature for (m^*, ID^*), even if the adversary can adaptively use *revelation* oracle which takes an ID $ID(\neq ID^*)$, then returns a valid secret-key for the ID and *signing* oracle which takes a message and an ID, then returns a correct signature for (m, ID). By the way, Shamir [Sha84] pointed out that an IBS scheme can be generically constructed from two types of digital signature scheme which sign on different-sized messages by using one of the signature scheme for secret-key generation and the other one for signature generation. Some concrete constructions of IBS with better properties such as better efficiency than the well-known generic construction were proposed in the previous works such as [PS06].

Attribute-Based Signature (ABS). The first ABS scheme was presented by Maji et al. [MPR11]. ABS is a generalization of IBS. In ABS systems, each user's secret-key is associated with an attribute $\mathbb{W}$. By using the secret-key, the user can generate a signature on any message associated with any predicate which is satisfied by $\mathbb{W}$. As IBS, ABS is required to be existentially unforgeable. Its definition is a natural extention from the definition of the one of IBS. Note that any attribute $\mathbb{W}^*$ satisfying the *target* predicate ϕ^* must not be queried to the *revelation* oracle. In addition, ABS is required to be signer-private. Informally, the property means that any signature reveals no information about the set of attributes of the signer. By the way, in the first ABS scheme [MPR11], each signer-predicate is required to be represented as a monotone span program (MSP). Following [MPR11], some works focused on ABS schemes with a more expressive signer-predicate, e.g., a non-monotone span program [OT11], a general circuit [SAH16].

5.1.2 Related Work

Signature schemes secure in BL model or CL model were proposed in previous works such as [KV09, BSW11, GV12, MTVY11].

When we construct a signature scheme existentially unforgeable in HL model, we have to keep in mind that there exists a well-known attack which makes the adversary successful in winning the security game. The attack is possible when the signing algorithm generating a valid signature on a message, and then outputting it, is included in the set of allowed leakage-functions. If the adversary outputs the signature (and the message) obtained through the leakage-function as a forged signature, he wins the game. Thus, signature schemes with HL resilience require a defence mechanism to the attack. Hereafter, in some cases, we call the attack *signature-leakage attack*.

The first signature scheme secure in HL model was presented by Faust et al. [FHN+12] at Asiacrypt'12. Their generic construction of signature uses labeled public-key encryption (LPKE) as a building block. Let dk denote its decryption-key. One of their important techniques to defend the signature-leakage attack is to include a ciphertext of the secret-key in each signature. Before considering the reason why such a defence effectively works, we need to know how the set of leakage-functions $\mathcal{F}_{FHNNZ12}(\xi(\lambda))$ is defined, where $\xi(\lambda)$ is a negligible function. The set of functions consists of every efficiently computable function f s.t. no PPT $\mathcal{B}$, given $(pk, f(pk, sk))$, can find out sk with a probability greater than $\xi(\lambda)$, where (pk, sk) is a pair of randomly generated keys. If we let $\xi(\lambda)$ satisfy $2^{-|dk|} >> \xi(\lambda)$, then any function outputting a valid signature on any message is excluded from $\mathcal{F}_{FHNNZ12}(\xi(\lambda))$ because any PPT $\mathcal{B}$ can find out sk with probability $2^{-|dk|} >> \xi(\lambda)$ by firstly finding out dk with probability $2^{-|dk|}$ and secondly decrypting the ciphertext of sk in the signature by using dk.

There are some disadvantages of Faust et al.'s signature scheme. Firstly, their signature is weakly existentially unforgeable, but not strongly existentially unforgeable. Secondly, the specific hardness parameter $\xi(\lambda)$ for their scheme is written as $1/exp(\lambda)$, where $exp(\lambda)$ is an exponential function. Thus, their signature is secure against exponentially hard-to-invert functions, but not secure against polynomially hard-to-invert functions.

Recently, Ishizaka and Matsuura [IM18] at ISC'18 proposed an improved signature scheme from Faust et al.'s one. Their signature scheme is strongly existentially unforgeable and resilient to polynomially hard-to-invert leakage functions. The signature schemes by Faust et al. [FHN+12] and Ishizaka and Matsuura [IM18] are closely related to our signature schemes in this work, so we explain the former one (resp. the latter one) in more detail in a paragraph below titled *Details of Digital Signature by Faust et al.* (resp. *Details of Digital Signature by Ishizaka and Matsuura.*)

In [YYH12, WMHT16], other signature schemes with HL resilience were presented. Their signature schemes were proven to be secure in *selective auxiliary input model* where only leakage caused by functions chosen independently of the public-key are considered.

Influenced by the works such as [KP10, GV12, TLNL14], Wu et al. [WTH16] proposed an efficient IBS scheme resilient to continual leakage in the *generic bilinear groups model* which requires a strong assumption. As far as we know, neither IBS nor ABS scheme secure in BL or HL model under standard cumputational assumptions in the standard model has been proposed.

5.1.3 Our Results

By extending a signature scheme by Ishizaka and Matsuura [IM18], we generically construct an IBS scheme existentially unforgeable in HL model. Also, we generically construct an ABS scheme, whose predicate is represented as a general circuit, existentially unforgeable in HL model and *computationally* signer-private. Previous ABS schemes (such as [MPR11, OT11, SAH16]) were proven to be *perfectly* signer-private. It must be hard to prove perfect signer-privacy of our ABS scheme because of the ciphertext of a valid secret-key included in each signature. We originally define the notion of computational signer-privacy, and prove that our ABS satisfies it. By the way, since our IBS and ABS are based on the signature in [IM18], they are resilient to *polynomially* hard-to-invert leakage.

In addition, we instantiate them under standard assumptions such as the decisional linear (DLIN) assumption [BBS04] and the symmetric external diffie-hellman assumption (SXDH) assumption. Let us emphasize that each one of the instantiations of IBS and ABS is not only the first one secure in HL model under standard assumptions, but also the first leakage-resilient one secure under standard assumptions.

As we mentioned earlier, our IBS or ABS scheme is closely related to each one of a signature scheme in [FHN$^+$12] and one in [IM18]. Before explaining the details of our schemes, we explain the details of the one in [FHN$^+$12] and the one in [IM18].

Details of Digital Signature by Faust et al. [FHN$^+$12]. Their signature scheme is generically built from three building blocks: *IND-wLCCA* secure labeled PKE (LPKE), *sound* and *zero-knowledge* non-interactive zero-knowledge proof (NIZK) [GS08] and *second pre-image resistant* hash-function (SPR-HF). Informally, second pre-image resistance means that no PPTA, given a bitstring $x \in \{0,1\}^n$, can find out any $x' \in \{0,1\}^n$ which is not equivalent to x and has the same hash value as x with a non-negliglble probability. Definitions of *IND-wLCCA*, *soundness* and *zero-knowledge* are given in Sect. **??**. A randomly chosen bitstring x is used as the secret-key sk, and its hash value, i.e., $y = H(x)$, is reserved in the public-key pk. A signature on a message m consists of (C, π), where C is an LPKE-ciphertext of sk under label m, and π is an NIZK-proof which proves that there exists sk s.t. its encryption can be C and its hash value is y and the signer has an evidence to support the statement.

Informally, its existential unforgeability is proven as follows. Firstly, initial security game is reasonably transformed to a game, namely $\mathtt{Game}_1$, where the decryption sk^* of the ciphertext C^* in the forged signature σ^* has the same hash value as the real secret-key sk because of the soundness of the NIZK. Secondly, $\mathtt{Game}_1$ is properly transformed to a game $\mathtt{Game}_2$, where the secret-key sk^* is the original secret-key sk itself, because of the SPR property of the hash function. Thirdly, $\mathtt{Game}_2$ is changed to $\mathtt{Game}_3$, where every proof in every signature on signing oracle is generated by using the trapdoor td instead of the real secret-key sk, becaused of the zero-knowledge of NIZK. Fourthly, $\mathtt{Game}_4$ is changed to $\mathtt{Game}_5$, where every ciphertext in every signature on signing oracle is generated by encrypting *all zero* bitstring, because of the ciphertext-indistinguishability of LPKE. Finally, every PPTA's winning probability in $\mathtt{Game}_5$ is proven to be negligible, because of the inversion hardness property of the leakage-function $f \in \mathcal{F}_{FHNNZ12}(2^{-\lambda-|dk|-|td|})$.

Details of Digital Signature by Ishizaka and Matsuura [IM18]. In this paragraph, we give details of a signature scheme weakly unforgeable and resilient to polynomially hard-to-invert leakage functions presented in [IM18]. Actually, that is obtained by modifying the signature scheme by Faust et al. with two steps.

At first modification, we generalize the SPR-HF to an original primitive named *PKX* scheme with *HtC-SK* (*Hard-to-Compute Secret-Key*) property. PKX consists of a key-generation algorithm which generates a pair of keys (pk, sk), a secret-key-verification algorithm SKVer which takes a secret-key sk' then outputs 1 if that is a valid secret-key under pk, and another secret-key-verification algorithm $\mathsf{SKVer2}$ which takes two secret-keys then outputs 1 if a relation holds between them. The property HtC-SK means that no PPTA, given a randomly chosen pair of keys (pk, sk), can find out a secret-key sk' s.t. $1 \leftarrow \mathsf{SKVer}(pk, sk') \wedge 0 \leftarrow \mathsf{SKVer2}(pk, sk', sk)$ with a non-negligible probability. Note that the definition of the set of leakage-functions is automatically changed as follows: the set consists of every f s.t. no PPT $\mathcal{B}$, given $(pk, f(pk, sk))$, can find out sk' s.t. $1 \leftarrow \mathsf{SKVer}(pk, sk') \wedge 1 \leftarrow \mathsf{SKVer2}(pk, sk', sk)$ with a non-negligible probability.

At second modification, we modify the definition of set of leakage-functions. In the modified definition, the PPT inverter $\mathcal{B}$ is given not only the public-key pk, but also some variables which are not included in pk but generated during running the key-generation algorithm. Specifically, the pair of variables (dk, td) is given. If we use such a definition, since the reduction simulator introduced to prove the *negligibility* of the game $\mathtt{Game}_5$ is not forced to randomly guess (dk, td), the new signature scheme can achieve the security against polynomially hard-to-invert leakage.

Details of Our IBS. Our IBS is an extension of the signature proposed by Ishizaka and Matsuura.

At first, we explain our definition of existential unforgeability in HL model for IBS schemes. Basically, its security game is defined in the same manner as the one in non-leakage setting, which means that the adversary in the game is required to use the secret-key-revelation oracle and signing oracle adaptively and then output a signature σ^*, a message m^* and an ID ID^* called the target ID such that σ^* is a valid signature on (m^*, ID^*) and no signature on (m^*, ID^*) was generated on the signing oracle, except for a leakage oracle whom $\mathcal{A}$ can use only once right before outputting the forged signature. The leakage oracle takes the target ID ID^* and a function f, then returns $f(\mathcal{L}_{ID^*})$, where the list $\mathcal{L}_{ID^*}$ includes the secret-key(s) for ID^* used to generate signature(s) on the signing oracle. Note that the function f should be chosen from a specific set of functions, namely $\mathcal{F}_{IBS}$, whose definition is given below.

Our IBS is built from three building blocks: LPKE, NIZK and *IBX*. IBX's algorithms are Setup and KeyGen which are the same algorithms as the ones of IBS, SKVer which takes (sk, ID) and outputs 1 if sk is a valid secret-key for ID under pk, and $\mathsf{SKVer2}$ which takes (sk, sk') and outputs 1 if a relation holds between the two keys. Informally, HtC-SK property for IBX is defined as follows: for any PPT $\mathcal{A}$, even if $\mathcal{A}$ can adaptively use the key-revalation oracle which takes an ID ID, then returns a secret-key for the ID, $\mathcal{A}$ cannot find out a secret-key sk^* satisfying both of the following conditions: sk^* is a valid secret-key for ID^* and the relation $\mathsf{SKVer2}$ does not hold between sk^* and every $sk_{ID^*} \in \mathcal{L}_{ID^*}$, where $\mathcal{L}_{ID^*}$ consists of all secret-keys for ID^* generated on the key-revelation oracle.

The generic construction of our IBS is as follows. We run Setup of IBX to generate a pair of keys (pk', mk'). We generate each secret-key for an ID ID by running KeyGen of IBX. We generate a signature $\sigma := (C, \pi)$ on a message m and an ID ID by using sk as follows. Firstly, we generate an LPKE-ciphertext C of sk under label $m\|ID$. Secondly, we generate an NIZK-proof π which proves that there exists sk' s.t. C is a ciphertext of sk' under label $m\|ID$ and sk' is a valid secret-key for ID.

The set of leakage functions is determined dependently of some variables such as pk', dk, td, ID^* and $\mathcal{L}_{ID^*}$ generated in the game. Informally, the set consists of every function f s.t. no PPT $\mathcal{B}$, given variables such as pk', dk, td, ID^*, f and $f(\mathcal{L}'_{ID^*})$, can find out sk^* s.t. $1 \leftarrow \mathsf{KeyGen}(pk', sk^*, ID^*)$ and there exists at least one $sk' \in \mathcal{L}'_{ID^*}$ satisfying that $1 \leftarrow \mathsf{SKVer2}(pk', sk^*, sk')$ with a non-negliglble probability, where $\mathcal{L}'_{ID^*}$ consists of $|\mathcal{L}_{ID^*}|$ number[1] of randomly generated secret-keys for ID^*.

Details of Our ABS. Our ABS whose predicate is represented as a general circuit is an extension (or generalization) of our IBS whose details were given in the last paragraph.

Our definition of existential unforgeability in HL model for ABS schemes is a generalization of the one for IBS schemes. Note that the leakage oracle takes the *target predicate* ϕ^* and a function f, then returns $f(\mathcal{L}_{\phi^*})$, where the list $\mathcal{L}_{\phi^*}$ is the union of every attribute $\mathcal{L}_{\mathbb{W}}$ for attribute $\mathbb{W}$ which satisfies ϕ^*, i.e., $\phi^*(\mathbb{W}) = 1$.

For the generic construction of ABS, we use an original primitive *ABX*. It consists of Setup and KeyGen which are the same algorithms as the ones of ABS, SKVer which takes (sk, ϕ) and outputs 1 if sk is a valid secret-key for an attribute $\mathbb{W}$ such that $\phi(\mathbb{W}) = 1$ under pk, and $\mathsf{SKVer2}$ which takes (sk, sk') and outputs 1 if a relation holds between the two keys. HtC-SK property for ABX is informally defined as follows: for any PPT $\mathcal{A}$, even if $\mathcal{A}$ can adaptively use the key-revalation oracle which takes $\mathbb{W}$, then returns a secret-key for it, $\mathcal{A}$ cannot find out (sk^*, ϕ^*) satisfying both conditions: sk^* is a valid key for ϕ^* and for every $\mathbb{W}^*$ satisfying ϕ^* and every $sk \in \mathcal{L}_{\mathbb{W}^*}$, the relation $\mathsf{SKVer2}$ does not hold between sk^* and sk, where $\mathcal{L}_{\mathbb{W}^*}$ consists of all secret-keys for $\mathbb{W}^*$ generated on the key-revelation oracle.

The generic construction of our ABS is as follows. We use Setup and KeyGen of ABX for the same purpose as the case of IBS. We generate $\sigma := (C, \pi)$ on m and a predicate ϕ by using a secret-key sk for an attribute $\mathbb{W}$ such that $\phi(\mathbb{W}) = 1$ as follows. Firstly, we generate an LPKE-ciphertext C of sk under label $m\|\phi$. Secondly, we generate an NIZK-proof π which proves that there exists sk' s.t. C is a ciphertext of sk' under label $m\|\phi$ and sk' is a valid secret-key for an attribute $\mathbb{W}$ satisfying ϕ.

[1] $|\mathcal{L}_{ID^*}|$ denotes the number of secret-keys in the list $\mathcal{L}_{ID^*}$.

$\mathtt{Expt}^{IND-wLCCA-b}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}(1^\lambda)$:

$(ek, dk) \leftarrow \mathsf{Gen}(1^\lambda)$, $(m_0, m_1, L^*, st) \leftarrow \mathcal{A}_1^{O_1^{dk}(C,L)}(ek)$, $C^* \leftarrow \mathsf{Enc}(ek, m_b, L^*)$

$b' \leftarrow \mathcal{A}_2^{O_2^{dk}(C,L)}(st, C^*)$. If $b' = b$, then return 1. Else, then return 0.

Figure 5.1: $\mathtt{Expt}^{IND-wLCCA-b}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}$

Informally, the set of leakage-functions consists of every function f s.t. no PPT $\mathcal{B}$, given variables such as pk', dk, td, ϕ^*, f and $f(\mathcal{L}'_{\phi^*})$, can find out sk^* s.t. $1 \leftarrow \mathsf{SKVer}(pk', sk^*, \phi^*)$ and there exists at least one $sk' \in \mathcal{L}'_{\phi^*}$ satisfying that $1 \leftarrow \mathsf{SKVer2}(pk', sk^*, sk')$ with a non-negliglble probability, where the list $\mathcal{L}'_{\phi^*}$ is a randomized one of the list $\mathcal{L}_{\phi^*}$ in the game.

5.1.4 Organization

This chapter is organized as follows. In Sect. 5.2, we give experiment-based definition of indistinguishability of LPKE. In the section, we also give definitions of existential unforgeability in HL model of IBS/ABS and signer-privacy of ABS. In the section, we also explain our original primitives IBX and ABX. In Sect. 5.3, our generic construction of IBS and its security proof for unforgeability are given. In Sect. 5.4, our generic construction of ABS and its security proofs for unforgeability and signer-privacy are given. In Sect. 5.5, we show that the generic constructions of IBS and ABS given in the previous sections can be instantiated under the DLIN and SXDH assumptions.

5.2 Preliminaries for Chapter 5

5.2.1 Experiment-Based Definition of Indistinguishability of LPKE

In this subsection, we give an experiment-based definition of ciphertext indistinguishability (IND-wLCCA) of LPKE scheme.

To define IND-wLCCA of an LPKE scheme $\Sigma_{\mathrm{LPKE}} = \{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$, we use the game $\mathtt{Expt}^{IND-wLCCA-b}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}(1^\lambda)$ in Fig.5.1, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ are PPT adversaries and b is a bit $b \in \{0, 1\}$. The oracles $O_1^{dk}(C, L)$ and $O_2^{dk}(C, L)$ in $\mathtt{Expt}^{IND-wLCCA-b}_{\Sigma_{LPKE},\mathcal{A}}(\lambda)$ are defined as follows: $O_1^{dk}(C, L)$ returns LPKE.$\mathsf{Dec}(dk, C, L)$. $O_2^{dk}(C, L)$ returns LPKE.$\mathsf{Dec}(dk, C, L)$ if $L \neq L^*$, and $\perp$ otherwise.

Advantage of $\mathcal{A}$ is defined as

$$\mathtt{Adv}^{IND-wLCCA}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}(\lambda) := \left|\Pr\left[\mathtt{Expt}^{IND-wLCCA-0}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}(1^\lambda)) \to 1\right] - \Pr\left[\mathtt{Expt}^{IND-wLCCA-1}_{\Sigma_{\mathrm{LPKE}},\mathcal{A}}(1^\lambda)) \to 1\right]\right|.$$

Definition 23. *If, for every PPT adversary $\mathcal{A}$, $\mathtt{Adv}^{IND-wLCCA}_{\mathcal{A},\Sigma_{\mathrm{LPKE}}}(\lambda)$ is negligible, then Σ_{LPKE} is IND-wLCCA secure.*

Remark. Given an LPKE scheme, experiment-based definition of IND-wLCCA of the LPKE scheme and game-based definition of IND-wLCCA of the LPKE scheme are equivalent. Thus, if an LPKE scheme is IND-wLCCA under the experiment-based definition, i.e., Def. 23, then the LPKE scheme is IND-wLCCA under the game-based definition, i.e., Def. 5, and vice versa. Proof of the equivalence is omitted in this paper since it is very easy.

5.2.2 Existential Unforgeability in HL Model of IBS

In non-leakage setting, it is desirable that an IBS scheme satisfies weak existential unforgeability under adaptively chosen ID/messages attack. For the concrete definition, see [PS06]. We define weak existential unforgeability under adaptively chosen ID/messages attack in HL model (HL-EUF-CMA) for IBS schemes. We consider the following game for an IBS scheme $\Sigma_{\rm IBS} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ which is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$. In the game, $\mathcal{F}_{\Sigma_{\rm IBS}}(\lambda)$ denotes a set of leakage-functions[2].

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$. $\mathcal{CH}$ initializes a list $\mathcal{L}_S$ as a set $\emptyset$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-generation oracle **Generate**, secret-key-revelation oracle **Reveal**, and signature-generation oracle **Sign** as follows.

> **Generate**$(ID \in \mathcal{I})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$. If a list $\mathcal{L}_{ID}$ for the ID has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such a list $\mathcal{L}_{ID}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.
>
> **Reveal**$(ID \in \mathcal{I}, i \in \mathbb{N})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{ID}|]$. $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{ID}$, then returns the secret-key.
>
> **Sign**$(ID \in \mathcal{I}, i \in \mathbb{N}, m \in \mathcal{M})$**:** $\mathcal{A}$ issues $ID \in \mathcal{I}$, $m \in \mathcal{M}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{ID}|]$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{ID}$, then generates $\sigma \leftarrow \mathsf{SIG.Sig}(pk, m, ID, sk)$. After that, $\mathcal{CH}$ returns σ and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\rm IBS}}(\lambda))$**.** $\mathcal{A}$ issues $ID^* \in \mathcal{I}$ which was not queried to **Reveal** and a function $f \in \mathcal{F}_{\Sigma_{\rm IBS}}$. $\mathcal{CH}$ returns $f(\mathcal{L}_{ID^*})$ [3].

Forgery$(m^* \in \mathcal{M}, \sigma^*)$**.** $\mathcal{A}$ sends a message $m^* \in \mathcal{M}$ and a signature σ^*. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{Ver}(pk, m^*, ID^*, \sigma^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]$. Its advantage $\mathsf{Adv}^{\mathcal{F}(\lambda)-HL-EUF-CMA}_{\Sigma_{\rm IBS},\mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 24. $\Sigma_{\rm IBS}$ *is HL-EUF-CMA secure with respect to the set of leakage-functions* $\mathcal{F}_{\Sigma_{\rm IBS}}(\lambda)$*, if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{\mathcal{F}(\lambda)-HL-EUF-CMA}_{\Sigma_{\rm IBS},\mathcal{A}}(\lambda)$ *is negligible.*

5.2.3 Existential Unforgeability in HL Model of ABS

In non-leakage setting, it is desirable for an ABS scheme to satisfy weak existential unforgeability under adaptively chosen predicate/messages attack [MPR11, OT11, SAH16]. We define weak existential unforgeability under adaptively chosen predicate/messages attack in HL model (HL-EUF-CMA) for ABS schemes. We consider the following game for an ABS scheme $\Sigma_{\rm ABS} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$ which is played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$. In the game, $\mathcal{F}_{\Sigma_{\rm ABS}}(\lambda)$ or $\mathcal{F}(\lambda)$ denotes a set of leakage-functions.

[2] We simply write $\mathcal{F}(\lambda)$ to indicate $\mathcal{F}_{\Sigma_{\rm IBS}}(\lambda)$ if the set of functions is obviously for $\Sigma_{\rm IBS}$.

[3] We assume that the ID ID^* was queried to **Generate** at least once and a list $\mathcal{L}_{ID^*}$ for the ID has already been generated.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$. The universal set of attributes is set as $\mathcal{U} = \{0,1\}^L$. A list $\mathcal{L}_S$ is set as a set $\emptyset$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-generation oracle **Generate**, secret-key-revelation oracle **Reveal**, and signature-generation oracle **Sign** as follows.

Generate$(\mathbb{W} \in \mathcal{U})$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W})$. If a list $\mathcal{L}_{\mathbb{W}}$ for the attribute has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such list $\mathcal{L}_{\mathbb{W}}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Reveal$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N})$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{W}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{\mathbb{W}}$, then returns it.

Sign$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$, $m \in \mathcal{M}$, a predicate ϕ and $i \in \mathbb{N}$ such that $i \in [1, |\mathcal{L}_{\mathbb{W}}|]$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$, then generates $\sigma \leftarrow \mathsf{SIG.Sig}(pk, m, \phi, sk)$. After that, $\mathcal{CH}$ returns σ, and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^*, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda))$**.** $\mathcal{A}$ issues a predicate ϕ^* such that $\phi^*(\mathbb{W}) = 0$ for every attribute $\mathbb{W}$ queried to **Reveal** and a function $f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$. $\mathcal{CH}$ returns $f(\mathcal{L}_{\phi^*})$, where the set $\mathcal{L}_{\phi^*}$ of secret-keys is set to $\bigcup_{\mathbb{W}^* \in \mathcal{U} \ s.t.\ \phi^*(\mathbb{W}^*)=1} \mathcal{L}_{\mathbb{W}^*}$[4].

Forgery$(m^* \in \mathcal{M}, \sigma^*)$**.** $\mathcal{A}$ sends a message m^* and a signature σ^*. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{Ver}(pk, m^*, \phi^*, \sigma^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$. The advantage $\mathsf{Adv}^{\mathcal{F}(\lambda)\text{-}HL\text{-}EUF\text{-}CMA}_{\Sigma_{\mathrm{ABS}},\mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 25. *Σ_{ABS} is HL-EUF-CMA-secure with respect to the set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$, if for every PPT $\mathcal{A}$, $\mathsf{Adv}^{\mathcal{F}(\lambda)\text{-}HL\text{-}EUF\text{-}CMA}_{\Sigma_{\mathrm{ABS}},\mathcal{A}}(\lambda)$ is negligible.*

5.2.4 Computational Signer-Privacy of ABS

For the existing ABS schemes [MPR11, OT11, SAH16], the authors discussed whether their scheme satisfies the information-theoretical signer-privacy, a.k.a. perfect privacy. In this paper, we originally define computational signer-privacy for ABS schemes and show that our scheme satisfies it. For the definition, we referred to the definition of semantic security for PKE scheme given by Goldwasser and Micali [GM84]. For an ABS scheme $\Sigma_{\mathrm{ABS}} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sig}, \mathsf{Ver}\}$, we use two experiments given in Fig.5.2, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ denotes a PPT adversary, $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ denotes a PPT simulator, h_1 and h_2 denote polynomial time computable functions, and $O^{pk,mk}_{CSP}$ denotes an oracle which takes an attribute $\mathbb{W} \in \mathcal{U}$ as input and returns $\mathsf{KeyGen}(pk, mk, \mathbb{W})$.

Definition 26. *An ABS scheme Σ_{ABS} is computationally signer-private, if $\forall \mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\exists \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ s.t. $\forall \mathcal{D}$, $Adv^{CSP}_{\Sigma_{\mathrm{ABS}},\mathcal{D},\mathcal{A},\mathcal{S},h_1,h_2}(\lambda) := |\Pr[\mathcal{D}(Expt^{CSP-0}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(1^\lambda, 1^L)) \to 1] - \Pr[\mathcal{D}(Expt^{CSP-1}_{\Sigma_{\mathrm{ABS}},\mathcal{S},h_1,h_2}(1^\lambda, 1^L)) \to 1]|$ is negligible.*

[4]We assume that at least one attribute $\mathbb{W}$ which satisfies the predicate ϕ^*, i.e., $\phi^*(\mathbb{W}) = 1$, was queried to **Generate** at least once and a list $\mathcal{L}_{\mathbb{W}}$ for the attribute has already been generated.

```
$\mathtt{Expt}^{\mathsf{CSP-0}}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(1^\lambda, 1^L)$:
  $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$.
  $(\mathcal{K}^*, \phi^*, m, st) \leftarrow \mathcal{A}_1^{O^{pk,mk}_{CSP}(\mathbb{W})}(pk)$, where
    $m \in \mathcal{M}$ and $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
  $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W}^*)$.
  $\sigma^* \leftarrow \mathsf{Sig}(pk, m, \phi^*, \mathbb{W}^*)$.
  $v \leftarrow \mathcal{A}_2^{O^{pk,mk}_{CSP}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$.
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
  Return $(d, \mathcal{K}^*)$.
```

```
$\mathtt{Expt}^{\mathsf{CSP-1}}_{\Sigma_{ABS},\mathcal{S},h_1,h_2}(1^\lambda, 1^L)$:
  $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$.
  $(\mathcal{K}^*, \phi^*, m, st) \leftarrow \mathcal{S}_1(pk)$, where
    $m \in \mathcal{M}$ and $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
  $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$

  $v \leftarrow \mathcal{S}_2(st, h_1(\mathbb{W}^*))$.
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
  Return $(d, \mathcal{K}^*)$.
```

Figure 5.2: Experiments $\mathtt{Expt}^{\mathsf{CSP-0}}_{\Sigma_{\mathrm{ABS}}}$ and $\mathtt{Expt}^{\mathsf{CSP-1}}_{\Sigma_{\mathrm{ABS}}}$.

5.2.5 An Original Primitive: IBX

Syntax of IBX. Ishizaka and Matsuura [IM18] introduced an original primitive named PKX which is related to signature schemes. A PKX scheme consists of the algorithm which generates a pair of public-key and secret-key and two secret-key-verification algorithms. We introduce such a primitive which is related to IBS schemes. An IBX scheme consists of Setup, KeyGen and two secret-key-verification algorithms whose definitions are given below.

$\mathsf{SKVer}(pk, sk, ID) \rightarrow 1\ /\ 0$. The (first) secret-key-verification algorithm takes pk, a secret-key sk and an ID ID as inputs and outputs 1 or 0. This algorithm outputs 1 only if a certain relationship holds among pk, sk and ID, or sk is a valid secret-key for ID under pk.

$\mathsf{SKVer2}(pk, sk, sk') \rightarrow 1\ /\ 0$. The second secret-key-verification algorithm takes pk, a secret-key sk and a secret-key sk' which may be sk as inputs and outputs 1 or 0. This algorithm outputs 1 only if a certain relationship holds between the two secret-keys.

We require that the two secret-key-verification algorithms satisfy that for every $\lambda \in \mathbb{N}$, every $l \in \mathbb{N}$, every $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$, every $ID \in \mathcal{I}$ and every $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, it holds that $\Pr[1 \leftarrow \mathsf{SKVer}(pk, sk, ID) \wedge 1 \leftarrow \mathsf{SKVer2}(pk, sk, sk)] = 1$.

Hard-to-Computer Secret-Key (HtC-SK) Property of IBX. Ishizaka and Matsuura [IM18] introduced a property for PKX named Hard-to-Compute-Secret-Key (HtC-SK). Intuitively, the property says that any PPT given a secret-key sk cannot find with a non-negligible probability a valid secret-key sk' such that a relation presented by the algorithm SKVer2 does not hold between sk and sk'. We define the property for IBX. We use the following game played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

Setup. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^l)$.

Query. $\mathcal{A}$ is allowed to adaptively use secret-key-revelation oracle **Reveal** as follows.

$\mathtt{Reveal}(ID \in \mathcal{I})$: $\mathcal{A}$ issues $ID \in \mathcal{I}$. $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, ID)$, then returns the secret-key to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

$\texttt{Compute}(ID^* \in \mathcal{I}, sk^*)$. $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathsf{SKVer}(pk, sk^*, ID^*)] \wedge [\bigwedge_{sk' \in \mathcal{L}_{ID^*}} [0 \leftarrow \mathsf{SKVer2}(pk, sk^*, sk')]]$. The advantage $\mathsf{Adv}^{HtC-SK}_{\Sigma_{\mathrm{IBS}},\mathcal{A}}(\lambda)$ is defined as probability $\Pr[\mathcal{A}\ wins.]$.

Definition 27. Σ_{IBX} *is HtC-SK, if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{HtC-SK}_{\Sigma_{\mathrm{IBX}},\mathcal{A}}(\lambda)$ *is negligible.*

5.2.6 An Original Primitive: ABX

Syntax of ABX. For IBS, we introduced a new primitive named IBX. For ABS, we introduce a primitive named ABX. An ABX scheme consists of Setup, KeyGen and the following two deterministic secret-key-verification algorithms.

$\mathsf{SKVer}(pk, sk, \phi) \to 1\ /\ 0$. The (first) secret-key-verification algorithm takes a secret-key sk and a predicate ϕ represented as a general circuit $\{L, N, I_1, I_2\}$ as inputs and outputs 1 or 0.

$\mathsf{SKVer2}(pk, sk, sk') \to 1\ /\ 0$. The second secret-key-verification algorithm takes two secret-keys sk and sk' as inputs and outputs 1 or 0.

We require that for every $\lambda \in \mathbb{N}$, every $L \in \mathbb{N}$, every $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$, every $\mathbb{W} \in \mathcal{U}$, every $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W})$ and every ϕ s.t. $\phi(\mathbb{W}) = 1$, it holds that $\Pr[1 \leftarrow \mathsf{SKVer}(pk, sk, \phi) \wedge 1 \leftarrow \mathsf{SKVer2}(pk, sk, sk)] = 1$.

HtC-SK Property of ABX. As the case of IBX, we define the HtC-SK property for ABX ??. We use the following game played by an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$.

$\texttt{Setup}$. $\mathcal{CH}$ runs $(pk, mk) \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$. The universal set of attributes is $\mathcal{U} = \{0, 1\}^L$.

$\texttt{Query}$. $\mathcal{A}$ is allowed to adaptively use secret-key-revelation oracle $\texttt{Reveal}$ as follows.

> **$\texttt{Reveal}(\mathbb{W} \in \mathcal{U})$:** $\mathcal{CH}$ generates $sk \leftarrow \mathsf{KeyGen}(pk, mk, \mathbb{W})$, then returns the secret-key to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

$\texttt{Compute}(\phi^*, sk^*)$. We say that $\mathcal{A}$ wins the game if $[1 \leftarrow \mathsf{SKVer}(pk, sk^*, \phi^*)]$ $\wedge\, [\bigwedge_{\mathbb{W} \in \mathcal{U}\ s.t.\ \phi^*(\mathbb{W})=1} [\bigwedge_{sk \in \mathcal{L}_{\mathbb{W}}} [0 \leftarrow \mathsf{SKVer2}(pk, sk^*, sk)]]]$. The advantage $\mathsf{Adv}^{HtC-SK}_{\Sigma_{\mathrm{ABX}},\mathcal{A}}(\lambda)$ is defined as $\Pr[\mathcal{A}\ wins.]$.

Definition 28. Σ_{ABX} *is HtC-SK, if for every PPT* $\mathcal{A}$*,* $\mathsf{Adv}^{HtC-SK}_{\Sigma_{\mathrm{ABX}},\mathcal{A}}(\lambda)$ *is negligible.*

5.3 Proposed IBS Scheme

We generically construct an IBS scheme in Subsect. 5.3.1, and prove that it is existentially unforgeable in HL model in Subsect. 5.3.2.

5.3.1 Generic Construction

We generically construct an IBS scheme $\Sigma_{\text{IBS}} = \{\mathsf{IBS.Setup}, \mathsf{IBS.KeyGen}, \mathsf{IBS.Sig}, \mathsf{IBS.Ver}\}$ from the following 3 building blocks:

- An LPKE scheme $\Sigma_{\text{LPKE}} = \{\mathsf{LPKE.Gen}, \mathsf{LPKE.Enc}, \mathsf{LPKE.Dec}\}$. Its plaintext space, label space and ciphertext space are denoted by $\mathcal{M}_L$, $\mathcal{L}_L$ and C_L, respectively.
- An NIZK scheme $\Sigma_{\text{NIZK}} = \{\mathsf{NIZK.Gen}, \mathsf{NIZK.Pro}, \mathsf{NIZK.Ver}\}$. $\mathcal{S}_1$ denotes the first simulator which makes Σ_{NIZK} satisfy the definition of zero-knowledge, i.e., Def. 12 in Sect. 2.12.
- An IBX scheme $\Sigma_{\text{IBX}} = \{\mathsf{IBX.Setup}, \mathsf{IBX.KeyGen}, \mathsf{IBX.SKVer}, \mathsf{IBX.SKVer2}\}$. Its ID space is denoted by $\mathcal{I}_X$. Let $\mathcal{K}_X$ denote the space of sk for every $ID \in \mathcal{I}_X$.

Specifically, each algorithm of Σ_{IBS} is defined as follows.

$\mathsf{IBS.Setup}(1^\lambda, 1^l)$**:** Run $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$ and $(pk', mk') \leftarrow \mathsf{IBX.Setup}(1^\lambda, 1^l)$ and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$.

The ID space $\mathcal{I}$ of Σ_{IBS} is equivalent to $\mathcal{I}_X$. The message space of Σ_{IBS} is the space $\mathcal{M}$ satisfying $\mathcal{L}_L = \mathcal{M} \| \mathcal{I}$.

Set $pk := (pk', ek, crs)$ and $mk := mk'$. Output (pk, mk). Language $\mathbb{L}$ is defined as

$$\begin{aligned} \mathbb{L} \;:=\; & \{(C, m, ID) \in C_L \times \mathcal{M} \times \mathcal{I}_X \mid \exists sk \in \mathcal{K}_X \; s.t. \\ & [C \leftarrow \mathsf{LPKE.Enc}(ek, sk, m\|ID)] \wedge [1 \leftarrow \mathsf{IBX.SKVer}(pk', sk, ID)]\}. \end{aligned} \quad (5.1)$$

$\mathsf{IBS.KeyGen}(pk, mk, ID)$**:** mk is written as mk'. `Return` $sk' := \mathsf{IBX.KeyGen}(pk', mk', ID)$.

$\mathsf{IBS.Sig}(pk, m, ID, sk)$**:** sk is parsed as (psk, ssk). Generate $C := \mathsf{LPKE.Enc}(ek, ssk, m\|ID)$. Set $x := (C, m, ID, psk)$ and $w := ssk$, then generate $\pi := \mathsf{NIZK.Pro}(crs, x, w)$. Output $\sigma := (C, \pi, psk)$.

$\mathsf{IBS.Ver}(pk, m, ID, \sigma)$**:** σ is parsed as (C, π, psk). Set $x := (C, m, ID, psk)$. Output $\mathsf{NIZK.Ver}(crs, x, \pi)$.

5.3.2 Proof of Existential Unforgeability in HL Model

In the definition of existential unforgeability in HL model for IBS schemes given in Subsect. **??**, the set of leakage-functions $\mathcal{F}(\lambda)$ was undefined. So, before we prove that our IBS scheme Σ_{IBS} satisfies the definition, we need to define the set of leakage-functions. In the definition given below, variables $(pk', mk', ek, dk, crs, td)$ denote the variables which were generated at **`Setup`** in the game, ID^* denotes the target ID, and an integer k denotes the total number of secret-keys in the list $\mathcal{L}_{ID^*}$ at **`Leak`** in the game.

Definition 29. *Set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda)$ consists of every polynomial time computable probabilistic (or deterministic) function $f : \{0,1\}^{\tau \cdot |sk|} \to \{0,1\}^*$ which has a randomness space $\mathcal{R}$ and satisfies that for every PPT $\mathcal{B}$,*

$$\Pr\left[\mathcal{B}(pk', mk', ek, dk, crs, td, ID^*, f, f(\{sk_i^*\}_{i\in[1,\tau]}; r)) \to sk^* \right.$$
$$\left.\text{s.t. } [1 \leftarrow \mathsf{IBX.SKVer}(pk', sk^*, ID^*)] \wedge \left[\bigvee_{i\in[1,\tau]} [1 \leftarrow \mathsf{IBX.SKVer2}(pk', sk^*, sk_i^*)]\right]\right]$$

is negligible, where $r \xleftarrow{\mathrm{R}} \mathcal{R}$ and for every $i \in [1,\tau]$, $sk_i^* \leftarrow \mathsf{IBX.KeyGen}(pk', mk', ID^*)$.

The existential unforgeability of Σ_{IBS} is guaranteed by the following theorem.

Theorem 5.3.1. *Σ_{IBS} is HL-EUF-CMA w.r.t. the set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda)$, if Σ_{LPKE} is IND-wLCCA, Σ_{NIZK} is sound and zero-knowledge, and Σ_{IBX} is HtC-SK.*

Proof of Theorem 5.3.1. Hereafter, $q_s \in \mathbb{N}$ denotes total number of times that PPT adversary $\mathcal{A}$ uses the signing oracle **Sign**. To prove Theorem 5.3.1, we use multiple games $\mathtt{Game}_i$, where $i \in \{0,1,2,3,4,4|1,\cdots,4|q_s,5\}$.

The game $\mathtt{Game}_0$ is the normal HL-EUF-CMA game w.r.t. the IBS scheme Σ_{IBS} and the set of leakage-functions $\mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda)$. Specifically, $\mathtt{Game}_0$ is the following game.

Setup. $\mathcal{CH}$ runs $(pk', mk') \leftarrow \mathsf{IBX.Setup}(1^\lambda, 1^l)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. $\mathcal{CH}$ sets $pk := (pk', ek, crs)$, and sends it to $\mathcal{A}$. $\mathcal{CH}$ sets $\mathcal{L}_S := \emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{CH}$ behaves as follows.

Generate($ID \in \mathcal{I}$)**:** $\mathcal{CH}$ generates $sk \leftarrow \mathsf{IBX.KeyGen}(pk', mk', ID)$. If a list $\mathcal{L}_{ID}$ for the ID has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. Else if such a list $\mathcal{L}_{ID}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Reveal($ID \in \mathcal{I}, i \in \mathbb{N}$)**:** $\mathcal{CH}$ retrieves i-th secret-key from $\mathcal{L}_{ID}$, then returns it.

Sign($ID \in \mathcal{I}, i \in \mathbb{N}, m \in \mathcal{M}$)**:** $\mathcal{CH}$ retrieves i-th secret-key sk from $\mathcal{L}_{ID}$. $\mathcal{CH}$ generates $C := \mathsf{LPKE.Enc}(ek, sk, m\|ID)$. $\mathcal{CH}$ sets $x := (C, m, ID)$ and $w := sk$, then generates $\pi := \mathsf{NIZK.Pro}(crs, x, w)$. After that, $\mathcal{CH}$ returns a signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Leak($ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda)$)**.** $\mathcal{CH}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). Statement x^* is set to (C^*, m^*, ID^*). $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]$.

We define the other games $\mathtt{Game}_i$, where $i \in \{1,2,3,4,4|1,\cdots,4|q_s,5\}$, as follows.

- $\mathtt{Game}_1$ is the same as $\mathtt{Game}_0$ except that $\mathcal{CH}$ generates a common reference string crs by running $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$ at **Setup**.

- $\mathtt{Game}_2$ is the same as $\mathtt{Game}_1$ except that $\mathcal{A}$'s winning condition is changed to the following one, where $sk^* := \mathsf{LPKE.Dec}(dk, C^*, m^*||ID^*)$: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{IBX.SKVer}(pk', sk^*, ID^*)]$.

- $\mathtt{Game}_3$ is the same as $\mathtt{Game}_2$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \mathsf{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathsf{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [1 \leftarrow \mathsf{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]]$.

- $\mathtt{Game}_4 (= \mathtt{Game}_{4|0})$ is the same as $\mathtt{Game}_3$ except for the following two parts. Firstly, $\mathcal{CH}$ generates a common reference string crs by running $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ at **Setup**. Secondly, when replying to a query to **Sign** at **Query**, $\mathcal{CH}$ generates a proof π by running $\pi \leftarrow \mathcal{S}_2(crs, x, td)$, where $\mathcal{S}_2$ denotes the second simulator in the definition of zero-knowledge for Σ_{NIZK}.

- $\mathtt{Game}_{4|i}$, where $i \in [1, q_s]$, is the same as $\mathtt{Game}_{4|0}$ except that when replying to j-th signing oracle query, where $j \leq i$, $\mathcal{CH}$ generates the ciphertext C_j by running $C_j \leftarrow \mathsf{LPKE.Enc}(ek, 0^{|sk|}, m||ID)$.

- $\mathtt{Game}_5$ is the following game, which is played by $\mathcal{A}$ and $\mathcal{CH}$.

  Setup. $\mathcal{CH}$ runs $(pk', mk') \leftarrow \mathsf{IBX.Setup}(1^\lambda, 1^l)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. $(pk', mk', ek, dk, crs, td)$ are sent to $\mathcal{A}$.

  Leak$(ID^* \in \mathcal{I}, \tau \in \mathbb{N}, f \in \mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\{sk_i^*\}_{i \in [1,\tau]})$, where for $i \in [1, \tau]$, $sk_i^* := \mathsf{IBX.KeyGen}(pk', mk', ID)$. Then $\mathcal{CH}$ sends it to $\mathcal{A}$.

  Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). $\mathcal{CH}$ decrypts the ciphertext C^* to get $sk^* := \mathsf{LPKE.Dec}(dk, C^*, m^*||ID^*)$. $\mathcal{A}$ is said to win the game if $[1 \leftarrow \mathsf{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{i \in [1,\tau]} [1 \leftarrow \mathsf{IBX.SKVer2}(pk', sk^*, sk_i^*)]]$.

Hereafter, for $i \in \{0, 1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$, W_i denotes the event where $\mathcal{A}$ wins the game $\mathtt{Game}_i$. Obviously, it holds that $\mathsf{Adv}_{\Sigma_{\mathrm{IBS}}, \mathcal{A}}^{\mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda)\text{-}HL\text{-}EUF\text{-}CMA}(\lambda) = \Pr[W_0] \leq \sum_{i=1}^{4} |\Pr[W_{i-1}] - [W_i]| + \sum_{i=1}^{q_s} |\Pr[W_{4|i-1}] - \Pr[W_{4|i}]| + |\Pr[W_{4|q_s}] - \Pr[W_5]| + \Pr[W_5]$. Theorem 5.3.1 is proven by the above inequality and the following lemmas. □

Lemma 5.3.1. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 5.3.2. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible if* Σ_{NIZK} *is sound.*

Lemma 5.3.3. $|\Pr[W_2] - \Pr[W_3]|$ *is negligible if* Σ_{IBX} *is HtC-SK.*

Lemma 5.3.4. $|\Pr[W_3] - \Pr[W_4]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 5.3.5. *For every* $i \in [1, q_s]$, $|\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$ *is negligible if* Σ_{LPKE} *is IND-wLCCA.*

Lemma 5.3.6. $\Pr[W_{4|q_s}]$ *is negligible if* $\Pr[W_5]$ *is negligible.*

Lemma 5.3.7. $\Pr[W_5]$ *is negligible.*

Proof of each lemma is given below.

<u>**Proof of Lemma 5.3.1.**</u> We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_0]-\Pr[W_1]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$. On one hand, the simulator $\mathcal{S}$ behaves as a PPT algorithm attempting to break the zero-knowledge for Σ_{NIZK}. On the other hand, $\mathcal{S}$ behaves as the challenger in $\texttt{Game}_0$ or $\texttt{Game}_1$. $\mathcal{S}$ is given a common reference string crs. If crs was generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow$ NIZK.Gen(1^λ)), then $\mathcal{S}$ simulates $\texttt{Game}_0$ (resp. $\texttt{Game}_1$) for the PPT adversary $\mathcal{A}$ properly. The concrete behaviour by $\mathcal{S}$ is the following.

Setup. $\mathcal{S}$ is given crs. $\mathcal{S}$ runs $(pk', mk') \leftarrow$ IBX.Setup$(1^\lambda, 1^l)$ and $(ek, dk) \leftarrow$ LPKE.Gen(1^λ). pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{S}$ behaves in the normal manner.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda))$**.** $\mathcal{S}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ID^*). $\mathcal{S}$ outputs $\beta' := 1$ if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]$. $\mathcal{S}$ outputs 0 otherwise.

It is obvious that if the common reference string is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow$ NIZK.Gen(1^λ)), then $\mathcal{S}$ simulates the game $\texttt{Game}_0$ (resp. $\texttt{Game}_1$) for $\mathcal{A}$ perfectly, and if and only if the event W_0 (resp. W_1) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain $\Pr[W_0] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]$ and $\Pr[W_1] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)]$. Hence, $|\Pr[W_0] - \Pr[W_1]| = |\Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]|$. □

<u>**Proof of Lemma 5.3.2.**</u> We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_1]-\Pr[W_2]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the soundness property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the soundness of the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given a common reference string crs of Σ_{NIZK}. $\mathcal{S}$ runs $(pk', mk') \leftarrow$ IBX.Setup$(1^\lambda, 1^l)$ and $(ek, dk) \leftarrow$ LPKE.Gen(1^λ). pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{S}$ behaves in the normal manner.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda))$**.** $\mathcal{S}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ID^*). The secret-key sk^* is set to LPKE.Dec$(dk, C^*, m^* \| ID^*)$. $\mathcal{S}$ outputs (x^*, π^*) if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [0 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)]$.

It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_1$ or $\mathtt{Game}_2$, perfectly.

By the way, the definitions of W_1 and W_2 gives us the following equations.

$$\begin{aligned}\Pr[W_1] &= \Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]] \quad (5.2)\\ \Pr[W_2] &= \Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \\ &\quad \wedge [1 \leftarrow \mathrm{IBX.SKVer}(pk', sk^*, ID^*)]] \quad (5.3)\end{aligned}$$

Hence, we obtain

$$\begin{aligned}&|\Pr[W_1] - \Pr[W_2]|\\ =\ &\Pr[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S]\\ &\wedge [0 \leftarrow \mathrm{IBX.SKVer}(pk', sk^*, ID^*)]]\\ =\ &\Pr[\mathcal{S}(crs) \rightarrow (x^*, \pi^*)\ s.t.\ [1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)]\\ &\wedge [0 \leftarrow \mathrm{IBX.SKVer}(pk', sk^*, ID^*)]]. \quad (5.4)\end{aligned}$$

By the definition of the language $\mathbb{L}$, i.e., (5.1), the following statement is true: for any $(C, m, ID) \in \mathbb{L}$, there exists $sk \in \mathcal{K}_X$ such that $[C \leftarrow \mathrm{LPKE.Enc}(ek, sk, m\|ID)] \wedge [1 \leftarrow \mathrm{IBX.SKVer}(pk', sk, ID)]$.

By the above statement, the correctness of Σ_{LPKE}, and the deterministic property of the algorithm SKVer of Σ_{IBX}, the following statement is also true: for any $(C, m, ID) \in \mathbb{L}$, it holds that $[1 \leftarrow \mathrm{IBX.SKVer}(pk', sk, ID)]$, where $sk := \mathrm{LPKE.Dec}(dk, C, m\|ID)$.

The following statement is contraposition of the above statement. Thus, it is also true. For any $C \in C_L$, any $m \in \mathcal{M}$ and any ID $ID \in \mathcal{I}_X$, if $[0 \leftarrow \mathrm{IBX.SKVer}(pk', sk, ID)]$, where $sk := \mathrm{LPKE.Dec}(dk, C, m\|ID)$, then $(C, m, ID) \notin \mathbb{L}$.

By the above statement and equation (5.4), we obtain

$$|\Pr[W_1] - \Pr[W_2]| = \Pr[\mathcal{S}(crs) \rightarrow (x^*, \pi^*)\ s.t.\ [1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [x^* \notin \mathbb{L}]].$$

□

Proof of Lemma 5.3.3. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_2] - \Pr[W_3]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the HtC-SK property for Σ_{IBX}.

We consider a PPT simulator $\mathcal{S}$ who behaves as a PPT adversary trying to break the property of HtC-SK for Σ_{IBX}. The concrete behaviour by $\mathcal{S}$ is the following.

Setup. $\mathcal{S}$ is given the keys (pk', mk') of Σ_{IBX}. $\mathcal{S}$ runs $(ek, dk) \leftarrow \mathrm{LPKE.Gen}(1^\lambda)$ and $crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Reveal** and **Sign**, $\mathcal{S}$ behaves in the normal manner. When $\mathcal{A}$ queries to **Generate**, $\mathcal{S}$ behaves in the same manner as $\mathcal{CH}$ in the definition of $\mathtt{Game}_0$ except that $\mathcal{S}$ queries ID to the oracle **Reveal** in the game for HtC-SK property of Σ_{IBX} to acquire the secret-key sk' for the ID.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda))$**.** $\mathcal{S}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ID^*). The secret-key sk^* is set to LPKE.Dec$(dk, C^*, m^*\|ID^*)$. $\mathcal{S}$ outputs (ID^*, sk^*) if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigwedge_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [0 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]]$.

It is obvious that $\mathcal{S}$ simulates Game$_2$ or Game$_3$ for $\mathcal{A}$ perfectly.

By the definitions of W_2 and W_3, we obtain

$$\begin{aligned}\Pr[W_2] &= \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \\ &\quad \wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)]] \\ \Pr[W_3] &= \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \\ &\quad \wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge \left[\bigvee_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]\right]\Bigg]\end{aligned}$$

Hence, we obtain

$$\begin{aligned}&|\Pr[W_2] - \Pr[W_3]| = \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \\ &\wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge \left[\bigwedge_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [0 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]\right]\Bigg].\end{aligned}$$

The above probability is equal to the probability by whom $\mathcal{S}$ wins the HtC-SK property game for Σ_{IBX}. □

Proof of Lemma 5.3.4. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_3]-\Pr[W_4]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the zero-knowledge property for the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given crs of Σ_{NIZK}. $\mathcal{S}$ runs $(pk', mk') \leftarrow \text{IBX.Setup}(1^\lambda, 1^l)$ and $(ek, dk) \leftarrow \text{LPKE.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the usual way. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(ID \in \mathcal{I}, i \in \mathbb{N}, m \in \mathcal{M})$**:** $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{ID}$. $\mathcal{CH}$ generates $C := \text{LPKE.Enc}(ek, sk, m\|ID)$. $\mathcal{CH}$ sets $x := (C, m, ID)$ and $w := sk$, then issues (x, w) to the oracle O_{zk} to get a proof π. After that, $\mathcal{CH}$ returns a signature $\sigma := (C, \pi)$ to $\mathcal{A}$, and sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda))$**.** $\mathcal{S}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). Statement x^* is set to (C^*, m^*, ID^*). sk^* is set to LPKE.Dec$(dk, C^*, m^*\|ID^*)$. $\mathcal{S}$ outputs 1 if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]]$.

It is obvious that if the common reference string crs is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow \text{NIZK.Gen}(1^\lambda)$) and the proof-generation oracle O_{zk} is $O_1^{crs,td}$ (resp. O_0^{crs}), then $\mathcal{S}$ simulates $\mathtt{Game}_4$ (resp. $\mathtt{Game}_3$) for $\mathcal{A}$ perfectly, and if and only if W_4 (resp. W_3) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain

$$\begin{aligned}|\Pr[W_3] - \Pr[W_4]| = \Big|&\Pr\left[1 \leftarrow \mathcal{S}^{O_0^{crs}(x,w)}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)\right] \\ &- \Pr\left[1 \leftarrow \mathcal{S}^{O_1^{crs,td}(x,w)}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)\right]\Big|.\end{aligned}$$

□

<u>**Proof of Lemma 5.3.5.**</u> We prove that for any $i \in [1, q_s]$ if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the IND-wLCCA security for Σ_{LPKE}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the IND-wLCCA security for the LPKE scheme Σ_{LPKE}. $\mathcal{CH}$ denotes the challenger in the IND-wLCCA security game. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given ek of Σ_{LPKE}. $\mathcal{S}$ runs $(pk', mk') \leftarrow \text{IBX.Setup}(1^\lambda, 1^l)$ and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the normal manner. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(ID \in \mathcal{I}, \hat{i} \in \mathbb{N}, m \in \mathcal{M})$: $\mathcal{CH}$ retrieves the $\hat{i}$-th secret-key sk from $\mathcal{L}_{ID}$. Suppose that the query is the j-th signing oracle query.

If $j \le i - 1$ (resp. $j \ge i + 1$), then $\mathcal{S}$ generates the ciphertext C as $C := \text{LPKE.Enc}(ek, 0^{|sk|}, m\|ID)$ (resp. $C := \text{LPKE.Enc}(ek, sk, m\|ID)$).

If $j = i$, then $\mathcal{S}$ issues $(0^{|sk|}, sk, m\|ID)$ as a challenge query in the IND-wLCCA game for Σ_{LPKE} to acquire a ciphertext C.

After acquiring C, $\mathcal{S}$ behaves as follows. $\mathcal{S}$ sets $x := (C, m, ID)$, then generates $\pi \leftarrow \mathcal{S}_2(crs, x, td)$. After that, $\mathcal{S}$ returns the signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{S}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda))$. $\mathcal{S}$ computes $f(\mathcal{L}_{ID^*})$, then returns it.

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). Statement x^* is set to (C^*, m^*, ID^*). $\mathcal{S}$ issues $(C^*, m^*\|ID^*)$ as a query to the decryption oracle at **Query 2** in the IND-wLCCA game for Σ_{LPKE}, and sk^* is returned. $\mathcal{S}$ outputs 1 if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, ID^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{sk_{ID^*} \in \mathcal{L}_{ID^*}} [1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_{ID^*})]]$.

Let $\beta \in \{0, 1\}$ be the challenge-bit in the IND-wLCCA security game for Π_{LPKE}. It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_{4|i-1}$ (resp. $\mathtt{Game}_{4|i}$) when $\beta = 1$ (resp. $\beta = 0$), and if and only if $W_{4|i-1}$ (resp. $W_{4|i}$) happens, $\mathcal{S}$ outputs $\beta' = 1$. It is also obvious that when $W_{4|i-1}$ or $W_{4|i}$ occurs, the label $m^*\|ID^*$ in the query $(C^*, m^*\|ID^*)$ to the oracle **Decrypt** at **Query 2** issued by $\mathcal{S}$ satisfies $m^*||ID^* \neq m_i||ID_i$, where (m_i, ID_i) is the pair of the message and ID issued as the i-th signing oracle query by $\mathcal{A}$, because of the rule in the game $\mathtt{Game}_{4|i-1}$ or $\mathtt{Game}_4$. Thus, the query $(C^*, m^*||ID^*)$ is not a forbidden query. Hence, we obtain $\Pr[W_{4|i-1}] = \Pr[\beta' = 1 \mid \beta = 1]$ and $\Pr[W_{4|i}] = \Pr[\beta' = 1|\beta = 0]$.

It is obvious that $\Pr[\beta' = \beta] = \Pr[\beta' = 0 \wedge \beta = 0] + \Pr[\beta' = 1 \wedge \beta = 1] = \frac{1}{2}(\Pr[\beta' = 0|\beta = 0] + \Pr[\beta' = 1|\beta = 1]) = \frac{1}{2}(\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0] + 1)$.

Hence, we obtain $\mathtt{Adv}^{IND-wLCCA}_{\Sigma_{\mathrm{LPKE}},\mathcal{S}} = |2 \cdot \Pr[\beta' = \beta] - 1| = |\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0]| = |\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$. □

Proof of Lemma 5.3.6. We prove that if we assume that there exists a PPT adversary $\mathcal{A}$ which wins $\mathtt{Game}_{4|q_s}$ with a non-negligible advantage, then we can construct a PPT simulator $\mathcal{S}$ which wins $\mathtt{Game}_5$ with a non-negligible advantage.

We consider a PPT simulator $\mathcal{S}$ which behaves an adversary in $\mathtt{Game}_5$. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given $(pk', mk', ek, dk, crs, td)$, which were generated by $(pk', mk') \leftarrow \mathrm{IBX.Setup}(1^\lambda, 1^l)$, $(ek, dk) \leftarrow \mathrm{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the normal manner. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(ID \in \mathcal{I}, i \in \mathbb{N}, m \in \mathcal{M})$: $\mathcal{CH}$ retrieves i-th secret-key sk from $\mathcal{L}_{ID}$. $\mathcal{S}$ generates $C := \mathrm{LPKE.Enc}(ek, 0^{|sk|}, m||ID)$. $\mathcal{S}$ sets $x := (C, m, ID)$, then generates $\pi := \mathcal{S}_2(crs, x, td)$. After that, $\mathcal{S}$ returns the signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{S}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, ID)\}$.

Leak$(ID^* \in \mathcal{I}, f \in \mathcal{F}_{\Sigma_{\mathrm{IBS}}}(\lambda))$. Let $k \in \mathbb{N}$ denote the cardinality of $\mathcal{L}_{ID^*}$. $\mathcal{S}$ issues (ID^*, k, f) as a query to the oracle **Leak** in $\mathtt{Game}_5$, then receives $f(\{sk_i^*\}_{i\in[1,k]})$, where for $i \in [1, k]$, sk_i^* is a secret-key for ID^* which was randomly generated by the challenger of $\mathtt{Game}_5$. Note that the actual secret-keys $\{sk_i^*\}_{i\in[1,k]}$ are unknown to $\mathcal{S}$. After that, $\mathcal{S}$ sends $f(\{sk_i^*\}_{i\in[1,k]})$ to $\mathcal{A}$.

Forgery(σ^*, m^*). $\mathcal{S}$ outputs (σ^*, m^*) at **Forgery** in $\mathtt{Game}_5$.

Since the list $\mathcal{L}_{ID^*}$ composed of k secret-keys and possessed by $\mathcal{S}$ and the secret-keys $\{sk_i^*\}_{i\in[1,k]}$ generated by the challenger of $\mathtt{Game}_5$ are indistinguishable for $\mathcal{A}$, $\mathcal{S}$ perfectly simulates $\mathtt{Game}_{4|q_s}$ for $\mathcal{A}$. Also obviously, if $\mathcal{A}$ wins $\mathtt{Game}_{4|q_s}$, $\mathcal{S}$ wins $\mathtt{Game}_5$. Hence, $\Pr[W_{4|q_s}] \leq \Pr[W_5]$. □

Proof of Lemma 5.3.7. Let us consider a PPT adversary $\mathcal{A}$ in $\mathtt{Game}_5$. We define the following three events.

$U^{pk',mk',ek,dk,crs,td}$ is the event where (pk', mk'), (ek, dk) and (crs, td) are randomly generated at **Setup** by $\mathrm{IBX.Setup}(1^\lambda, 1^l)$, $\mathrm{LPKE.Gen}(1^\lambda)$ and $\mathcal{S}_1(1^\lambda)$, respectively. $V^{ID^*,\tau,f}_{pk',mk',ek,dk,crs,td}$ is the event where $\mathcal{A}$, given the variables $(pk', mk', ek, dk, crs, td)$ at **Setup**, chooses $ID^* \in \mathcal{I}$, $\tau \in \mathbb{N}$ and function f at **Leak**. $W_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}$ is the event where $\mathcal{A}$ wins in the following game $\mathtt{Game}_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}$.

Setup. $\mathcal{A}$ is given $(pk', mk', ek, dk, crs, td, ID^*, f, f(\{sk_i^*\}_{i\in[1,\tau]}))$, where, for $i \in [1,\tau]$, $sk_i^* \leftarrow$ IBX.KeyGen(pk', mk', ID^*).

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). sk^* is set to LPKE.Dec$(dk, C^*, m^*\|ID^*)$. $\mathcal{A}$ wins the game if it holds that $[1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{i\in[1,\tau]}[1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_i^*)]]$.

By the definition of W_5 and the definitions of the above three events, we obtain

$$\Pr[W_5] = \sum_{(pk',mk',ek,dk,crs,td)} \sum_{(ID^*,\tau,f)} \Pr\left[W_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}\right] \cdot \Pr\left[V^{ID^*,\tau,f}_{pk',mk',ek,dk,crs,td}\right] \cdot \Pr\left[U^{pk',mk',ek,dk,crs,td}\right].$$

By the above equation and Lemma 5.3.8, there exists a negligible function $\epsilon(\lambda)$ such that

$$\Pr[W_5] < \sum_{(pk',mk',ek,dk,crs,td)} \sum_{(ID^*,\tau,f)} \epsilon(\lambda) \cdot \Pr\left[V^{ID^*,\tau,f}_{pk',mk',ek,dk,crs,td}\right] \cdot \Pr\left[U^{pk',mk',ek,dk,crs,td}\right]. \quad (5.5)$$

We give two facts. It obviously holds that $\sum_{(pk',mk',ek,dk,crs,td)} \Pr\left[U^{pk',mk',ek,dk,crs,td}\right] = 1$. It also obviously holds that for any $(pk', mk', ek, dk, crs, td)$, $\sum_{(ID^*,\tau,f)} \Pr\left[V^{ID^*,\tau,f}_{pk',mk',ek,dk,crs,td}\right] = 1$. By these facts and (5.5), we obtain $\Pr[W_5] < \epsilon(\lambda)$. □

Lemma 5.3.8. *For any PPT $\mathcal{A}$, any (pk', mk'), any (ek, dk), any (crs, td), any $ID^* \in \mathcal{I}$, any $\tau \in \mathbb{N}$, and any $f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda)$, $\Pr\left[W_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}\right]$ is negligible.*

Proof of Lemma 5.3.8. We prove that if we assume that there exists a PPT adversary $\mathcal{A}$ which wins the game Game$_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}$ with a non-negligible probability, we can construct a PPT simulator which leads us to a contradiction to the fact that the function f is in $\mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda)$. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given $(pk', mk', ek, dk, crs, td, ID^*, f, f(\{sk_i^*\}_{i\in[1,\tau]}))$, where, for $i \in [1,\tau]$, $sk_i^* \leftarrow$ IBX.KeyGen(pk', mk', ID^*). $\mathcal{S}$ gives the variables to $\mathcal{A}$.

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). sk^* is set to LPKE.Dec$(dk, C^*, m^*\|ID^*)$. $\mathcal{S}$ outputs sk^*.

It is obvious that $\mathcal{S}$ simulates the game Game$_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}$ for $\mathcal{A}$ perfectly. If $\mathcal{A}$ wins the game, then $\mathcal{S}$ is able to acquire and output a secret-key sk^* such that $[1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge [\bigvee_{i\in[1,\tau]}[1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_i^*)]]$. Hence, we obtain

$$\begin{aligned}
&\Pr[\mathcal{S}(pk', mk', ek, dk, crs, td, ID^*, f, f(\{sk_i^*\}_{i\in[1,\tau]})) \to sk^* \\
&\quad \text{s.t. } [1 \leftarrow \text{IBX.SKVer}(pk', sk^*, ID^*)] \wedge \left[\bigvee_{i\in[1,\tau]} [1 \leftarrow \text{IBX.SKVer2}(pk', sk^*, sk_i^*)]\right]\Bigg] \\
&= \Pr\left[W_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}\right],
\end{aligned}$$

where, for $i \in [1,\tau]$, $sk_i^* \leftarrow$ IBX.KeyGen(pk', mk', ID^*). Assuming that there exists a polynomial function $poly(\lambda)$ such that $\Pr[W_{pk',mk',ek,dk,crs,td,ID^*,\tau,f}] \geq 1/poly(\lambda)$ leads us to a contradiction to the fact that $f \in \mathcal{F}_{\Sigma_{\text{IBS}}}(\lambda)$. □

5.4 Proposed ABS Scheme for a General Circuit

We generically construct an ABS scheme for a general circuit in Subsect. 5.4.1. We prove that it is existentially unforgeable in HL model and computationally signer-private in Subsect. 5.4.2 and Subsect. 5.4.3, respectively.

5.4.1 Generic Construction

We generically construct an ABS scheme $\Sigma_{\text{ABS}} = \{\text{ABS.}\mathsf{Setup}, \text{ABS.}\mathsf{KeyGen}, \text{ABS.}\mathsf{Sig}, \text{ABS.}\mathsf{Ver}\}$ whose predicate is represented as a general circuit by using the following 3 building blocks.

- An LPKE scheme $\Sigma_{\text{LPKE}} = \{\text{LPKE.}\mathsf{Gen}, \text{LPKE.}\mathsf{Enc}, \text{LPKE.}\mathsf{Dec}\}$. For notations $\mathcal{M}_L$, $\mathcal{L}_L$ and C_L, see Subsect. 5.3.1.
- An NIZK scheme $\Sigma_{\text{NIZK}} = \{\text{NIZK.}\mathsf{Gen}, \text{NIZK.}\mathsf{Pro}, \text{NIZK.}\mathsf{Ver}\}$. For a notation $\mathcal{S}_1$, see Subsect. 5.3.1.
- An ABX scheme $\Sigma_{\text{ABX}} = \{\text{ABX.}\mathsf{Setup}, \text{ABX.}\mathsf{KeyGen}, \text{ABX.}\mathsf{SKVer}, \text{ABX.}\mathsf{SKVer2}\}$ whose predicate is represented as a circuit. Its universal set of attributes and predicate space are denoted by $\mathcal{U}_X$ and $\mathcal{P}_X$, respectively. Let $\mathcal{K}_X$ denote the space of sk for every $\mathbb{W} \in \mathcal{U}_X$.

Specifically, each algorithm of Σ_{ABS} is defined as follows.

ABS.$\mathsf{Setup}(1^\lambda, 1^L)$: It runs $(ek, dk) \leftarrow \text{LPKE.}\mathsf{Gen}(1^\lambda)$, $(pk', mk') \leftarrow \text{ABX.}\mathsf{Setup}(1^\lambda, 1^L)$ and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$.

Universal set of attributes $\mathcal{U}$, secret-key space $\mathcal{K}$ and predicate space $\mathcal{P}$ of Σ_{ABS} are equivalent to $\mathcal{U}_X$, $\mathcal{K}_X$ and $\mathcal{P}_X$, respectively. Message space $\mathcal{M}$ of Σ_{ABS} is the space satisfying $\mathcal{L}_L = \mathcal{M} \| \mathcal{P}$.

It sets $pk := (pk', ek, crs)$ and $mk := mk'$, and outputs (pk, mk). The language $\mathbb{L}$ is defined as

$$\begin{aligned}\mathbb{L} := \{&(C, m, \phi) \in C_L \times \mathcal{M} \times \mathcal{P}_X \mid \exists sk \in \mathcal{K}_X \ s.t. \\ &[C \leftarrow \text{LPKE.}\mathsf{Enc}(ek, sk, m\|\phi)] \wedge [1 \leftarrow \text{ABX.}\mathsf{SKVer}(pk', sk, \phi)]\}. \end{aligned} \tag{5.6}$$

ABS.$\mathsf{KeyGen}(pk, mk, \mathbb{W} \in \mathcal{U})$: mk is written as mk'. It outputs $sk := \text{ABX.}\mathsf{KeyGen}(pk', mk', \mathbb{W})$.

ABS.$\mathsf{Sig}(pk, m \in \mathcal{M}, \phi \in \mathcal{P}, sk)$: ϕ is represented as a circuit $\{L, N, I_1, I_2\}$. It generates $C := \text{LPKE.}\mathsf{Enc}(ek, sk, m\|\phi)$. It sets $x := (C, m, \phi)$ and $w := sk$, then generates $\pi := \text{NIZK.}\mathsf{Pro}(crs, x, w)$. It outputs $\sigma := (C, \pi)$.

ABS.$\mathsf{Ver}(pk, m \in \mathcal{M}, \phi \in \mathcal{P}, \sigma)$: σ is parsed as (C, π). It sets $x := (C, m, \phi)$, then outputs $\text{NIZK.}\mathsf{Ver}(crs, x, \pi)$.

5.4.2 Proof of Existential Unforgeability in HL Model

Before we prove the existential unforgeability of our ABS scheme Σ_{ABS}, we define the set of leakage-functions $\mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$. In the definition given below, variables $(pk', mk', ek, dk, crs, td)$ denote the variables which were generated at **Setup** in the game, ϕ^* denotes the target predicate, for an attribute $\mathbb{W} \in \mathcal{U}$, $\mathcal{L}_{\mathbb{W}}$ denotes the list for $\mathbb{W}$ at **Leak** in the game, and a set $\mathcal{L}_{\widehat{\mathbb{W}}}$ denotes a multiset which is generated as follows: we initialize $\mathcal{L}_{\widehat{\mathbb{W}}}$ as en empty set, and then for every $\mathbb{W} \in \mathcal{U}$ s.t. $\phi^*(\mathbb{W}) = 1$, we add $|\mathcal{L}_{\mathbb{W}}|$ number of the attribute $\mathbb{W}$ into $\mathcal{L}_{\widehat{\mathbb{W}}}$.

Definition 30. *Set of leakage-functions $\mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$ consists of every polynomial time computable probabilistic (or deterministic) function $f : \{0,1\}^{\sum_{sk' \in \mathcal{L}'_{\phi^*}} |sk'|} \to \{0,1\}^*$ which has a randomness space $\mathcal{R}$ and satisfies that for every PPT $\mathcal{B}$,*

$$\Pr\Big[\mathcal{B}\left(pk', mk', ek, dk, crs, td, \phi^*, \mathcal{L}_{\widehat{\mathbb{W}}}, f, f\left(\mathcal{L}'_{\phi^*}; r\right)\right) \to sk^*$$
$$s.t.\ [1 \leftarrow \mathsf{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge \left[\bigvee_{sk' \in \mathcal{L}'_{\phi^*}} [1 \leftarrow \mathsf{ABX.SKVer2}(pk', sk^*, sk')]\right]\Big]$$

is negligible, where $r \xleftarrow{R} \mathcal{R}$ and for every $\mathbb{W} \in \mathcal{L}_{\widehat{\mathbb{W}}}$, $sk' := \mathsf{ABX.KeyGen}(pk, mk, \mathbb{W})$ and $\mathcal{L}'_{\phi^*} := \mathcal{L}'_{\phi^*} \cup \{sk'\}$.

The existential unforgeability of Σ_{ABS} is guaranteed by the following theorem.

Theorem 5.4.1. *Σ_{ABS} is HL-EUF-CMA w.r.t. $\mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$, if Σ_{LPKE} is IND-wLCCA, Σ_{NIZK} is sound and zero-knowledge, and Σ_{ABX} is HtC-SK.*

Proof of Theorem 5.4.1. Hereafter, $q_s \in \mathbb{N}$ denotes the number of times that PPT adversary $\mathcal{A}$ uses the signing oracle **Sign**. To prove Theorem 5.4.1, we use multiple games $\texttt{Game}_i$, where $i \in \{0, 1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$.

The game $\texttt{Game}_0$ is the normal HL-EUF-CMA game w.r.t. Σ_{ABS} and $\mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$. Specifically, $\texttt{Game}_0$ is the following game.

Setup. $\mathcal{U} := \{0,1\}^L$. $\mathcal{CH}$ runs $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$, $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk and mk are set to $pk := (pk', ek, crs)$ and $mk := mk'$, respectively. pk is given to $\mathcal{A}$. $\mathcal{L}_S$ is set to $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{CH}$ behaves as follows.

Generate($\mathbb{W} \in \mathcal{U}$)**:** $\mathcal{CH}$ generates $sk \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W})$. If the list $\mathcal{L}_{\mathbb{W}}$ for the attribute has not been generated, $\mathcal{CH}$ generates it and sets it to $\{sk\}$. If the list $\mathcal{L}_{\mathbb{W}}$ has already been generated, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Reveal($\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}$)**:** $\mathcal{CH}$ retrieves the i-th secret-key from $\mathcal{L}_{\mathbb{W}}$, then returns the secret-key.

Sign($\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}, m \in \mathcal{M}, \phi$)**:** ϕ is represented as a circuit $\{L, N, I_1, I_2\}$. $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$. $\mathcal{CH}$ generates $C :=$ LPKE.Enc($ek, sk, m\|\phi$). $\mathcal{CH}$ sets $x := (C, m, \phi)$ and $w := sk$, then generates $\pi :=$ NIZK.Pro(crs, x, w). After that, $\mathcal{CH}$ returns a signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak($\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$)**.** ϕ^* is represented as a circuit. $\mathcal{CH}$ returns $f(\mathcal{L}_{\phi^*})$, where a set $\mathcal{L}_{\phi^*}$ of secret-keys is set as $\bigcup_{\mathbb{W} \in \mathcal{U} \ s.t. \ \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{A}$ is said to win the game if $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$

We define the other games $\texttt{Game}_i$, where $i \in \{1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$, as follows.

- $\texttt{Game}_1$ is the same as $\texttt{Game}_0$ except that $\mathcal{CH}$ generates a common reference string crs by running $crs \leftarrow$ NIZK.Gen(1^λ) at **Setup**.
- $\texttt{Game}_2$ is the same as $\texttt{Game}_1$ except that $\mathcal{A}$'s winning condition is changed to the following one, where $sk^* :=$ LPKE.Dec($dk, C^*, m^*\|\phi^*$): $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)]$.
- $\texttt{Game}_3$ is the same as $\texttt{Game}_2$ except that $\mathcal{A}$'s winning condition is changed to the following one: $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{\mathbb{W} \in \mathcal{U} \ s.t. \ \phi^*(\mathbb{W})=1} [\bigvee_{sk \in \mathcal{L}_{\mathbb{W}}} [1 \leftarrow \text{ABX.SKVer2}(pk', sk^*, sk)]]]$.
- $\texttt{Game}_4 (= \texttt{Game}_{4|0})$ is the same as $\texttt{Game}_3$ except that $\mathcal{CH}$ generates a common reference string crs by running $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ at **Setup**. When replying to a query to **Sign** at **Query**, $\mathcal{CH}$ generates a proof π by running $\pi \leftarrow \mathcal{S}_2(crs, x, td)$, where $\mathcal{S}_2$ denotes the second simulator in the definition of zero-knowledge for Σ_{NIZK}.
- $\texttt{Game}_{4|i}$, where $i \in [1, q_s]$, is the same as $\texttt{Game}_{4|0}$ except that when replying to the j-th signing oracle query, where $j \leq i$, $\mathcal{CH}$ generates the ciphertext C_j by running $C_j \leftarrow$ LPKE.Enc($ek, 0^{|sk|}, m\|\phi$).
- $\texttt{Game}_5$ is the following game, which is played by $\mathcal{A}$ and $\mathcal{CH}$.

  Setup. $\mathcal{CH}$ runs $(pk', mk') \leftarrow$ ABX.Setup($1^\lambda, 1^L$), $(ek, dk) \leftarrow$ LPKE.Gen(1^λ), and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. $(pk', mk', ek, dk, crs, td)$ are sent to $\mathcal{A}$.

  Leak($\phi^*, \mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^*$ $s.t.$ $\phi^*(\mathbb{W}_1^*) = \cdots = \phi^*(\mathbb{W}_\tau^*) = 1, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)$)**.** $\mathcal{CH}$ computes $f(\{sk_i^*\}_{i\in[1,\tau]})$, where for $i \in [1, \tau]$, the secret-key sk_i^* is generated as $sk_i^* :=$ ABX.KeyGen($pk', mk', \mathbb{W}_i^*$). Then $\mathcal{CH}$ sends it to $\mathcal{A}$.

  Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). By decrypting C^*, we get $sk^* :=$ LPKE.Dec($dk, C^*, m^*\|\phi^*$). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{A}$ is said to win the game if $[1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{i \in [1,\tau]} [1 \leftarrow \text{ABX.SKVer2}(pk', sk^*, sk_i^*)]]$.

Hereafter, for $i \in \{0, 1, 2, 3, 4, 4|1, \cdots, 4|q_s, 5\}$, W_i denotes the event where $\mathcal{A}$ wins the game $\texttt{Game}_i$. Obviously, it holds that $\texttt{Adv}_{\Sigma_{\text{ABS}},\mathcal{A}}^{\mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda)\text{-}HL\text{-}EUF\text{-}CMA}(\lambda) = \Pr[W_0] \leq \sum_{i=1}^{4} |\Pr[W_{i-1}] - [W_i]| + \sum_{i=1}^{q_s} |\Pr[W_{4|i-1}] - \Pr[W_{4|i}]| + |\Pr[W_{4|q_s}] - \Pr[W_5]| + \Pr[W_5]$.

Theorem 5.4.1 is proven by the above inequality and the following lemmas. □

Lemma 5.4.1. $|\Pr[W_0] - \Pr[W_1]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 5.4.2. $|\Pr[W_1] - \Pr[W_2]|$ *is negligible if* Σ_{NIZK} *is sound.*

Lemma 5.4.3. $|\Pr[W_2] - \Pr[W_3]|$ *is negligible if* Σ_{ABX} *is HtC-SK.*

Lemma 5.4.4. $|\Pr[W_3] - \Pr[W_4]|$ *is negligible if* Σ_{NIZK} *is zero-knowledge.*

Lemma 5.4.5. *For every* $i \in [1, q_s]$, $|\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$ *is negligible if* Σ_{LPKE} *is IND-wLCCA.*

Lemma 5.4.6. $\Pr[W_{4|q_s}]$ *is negligible if* $\Pr[W_5]$ *is negligible.*

Lemma 5.4.7. $\Pr[W_5]$ *is negligible.*

Proof of each lemma is given below.

Proof of Lemma 5.4.1. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_0]-\Pr[W_1]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$. On one hand, the simulator $\mathcal{S}$ behaves as a PPT algorithm attempting to break the zero-knowledge for Σ_{NIZK}. On the other hand, $\mathcal{S}$ behaves as the challenger in $\mathtt{Game}_0$ or $\mathtt{Game}_1$. $\mathcal{S}$ is given a common reference string crs. If crs was generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)$), then $\mathcal{S}$ simulates $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) for the PPT adversary $\mathcal{A}$ properly. The concrete behaviour by $\mathcal{S}$ is the following.

Setup. $\mathcal{S}$ is given crs. $\mathcal{S}$ runs $(pk', mk') \leftarrow \mathrm{ABX.Setup}(1^\lambda, 1^L)$ and $(ek, dk) \leftarrow \mathrm{LPKE.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{S}$ behaves in the usual manner.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda))$. $\mathcal{CH}$ computes $f(\mathcal{L}_{\phi^*})$ and returns it, where $\mathcal{L}_{\phi^*} := \bigcup_{\mathbb{W}\ s.t.\ \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{S}$ outputs 1 if it holds that $[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]$. $\mathcal{S}$ outputs 0 otherwise.

It is obvious that if the common reference string is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)$), then $\mathcal{S}$ simulates the game $\mathtt{Game}_0$ (resp. $\mathtt{Game}_1$) for $\mathcal{A}$ perfectly, and if and only if the event W_0 (resp. W_1) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain $\Pr[W_0] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]$ and $\Pr[W_1] = \Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)]$. Hence, $|\Pr[W_0] - \Pr[W_1]| = |\Pr[1 \leftarrow \mathcal{S}(crs) \mid crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)] - \Pr[1 \leftarrow \mathcal{S}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)]|$. □

Proof of Lemma 5.4.2. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_1]-\Pr[W_2]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the soundness property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the soundness of the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given a common reference string crs of Σ_{NIZK}. $\mathcal{S}$ runs $(pk', mk') \leftarrow \text{ABX.Setup}(1^\lambda, 1^L)$ and $(ek, dk) \leftarrow \text{LPKE.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate**, **Reveal** and **Sign**, $\mathcal{S}$ behaves in the normal manner.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\mathcal{L}_{\phi^*})$ and returns it, where $\mathcal{L}_{\phi^*} := \bigcup_{\mathbb{W}\ s.t.\ \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{S}$ sets $sk^* := \text{LPKE.Dec}(dk, C^*, m^*\|\phi^*)$. $\mathcal{S}$ outputs (x^*, π^*) if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [0 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)]$.

It is obvious that $\mathcal{S}$ simulates $\texttt{Game}_1$ or $\texttt{Game}_2$, perfectly.

By the way, the definitions of W_1 and W_2 gives us the following equations.

$$\begin{aligned} \Pr[W_1] &= \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S]] && (5.7)\\ \Pr[W_2] &= \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \\ &\quad \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)]] && (5.8)\end{aligned}$$

Hence, we obtain

$$\begin{aligned} &|\Pr[W_1] - \Pr[W_2]| \\ =\ & \Pr[[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \\ & [0 \leftarrow \text{ABX.SKVer}(pk', sk^*, \mathbb{S}^*)]] \\ =\ & \Pr[\mathcal{S}(crs) \to (x^*, \pi^*)\ s.t.\ [1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \\ & \wedge [0 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)]]\,. && (5.9)\end{aligned}$$

By the definition of the language $\mathbb{L}$, i.e., (5.6), the following statement is true: for any $(C, m, \phi) \in \mathbb{L}$, there exists $sk \in \mathcal{K}_X$ such that $[C \leftarrow \text{LPKE.Enc}(ek, sk, m\|\phi)] \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk, \phi)]$.

By the above statement, the correctness of Σ_{LPKE}, and the deterministic property of the algorithm SKVer of Σ_{ABX}, the following statement is true: for any $(C, m, \phi) \in \mathbb{L}$, it holds that $[1 \leftarrow \text{ABX.SKVer}(pk', sk, \phi)]$, where $sk := \text{LPKE.Dec}(dk, C, m\|\phi)$.

The contraposition of the above statement is the following statement: for any $C \in C_L$, any $m \in \mathcal{M}$ and any $\phi \in \mathcal{P}_X$, if $[0 \leftarrow \text{ABX.SKVer}(pk', sk, \phi)]$, where $sk := \text{LPKE.Dec}(dk, C, m\|\phi)$, then $(C, m, \phi) \notin \mathbb{L}$.

By the above statement and the equation (5.9), we obtain

$$|\Pr[W_1] - \Pr[W_2]| = \Pr[\mathcal{S}(crs) \to (x^*, \pi^*)\ s.t.\ [1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [x^* \notin \mathbb{L}]]\,.$$

□

Proof of Lemma 5.4.3. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_2]-\Pr[W_3]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the HtC-SK property for Σ_{ABX}.

We consider a PPT simulator $\mathcal{S}$ who behaves as a PPT adversary trying to break the property of HtC-SK for Σ_{ABX}. The concrete behaviour by $\mathcal{S}$ is the following.

Setup. $\mathcal{S}$ is given the keys (pk', mk') of Σ_{ABX}. $\mathcal{S}$ runs $(ek, dk) \leftarrow \mathrm{LPKE.Gen}(1^\lambda)$ and $crs \leftarrow \mathrm{NIZK.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Reveal** and **Sign**, $\mathcal{S}$ behaves in the same manner as the challenger $\mathcal{CH}$ in the definition of $\mathtt{Game}_0$. When $\mathcal{A}$ queries $\mathbb{W}$ to **Generate**, $\mathcal{S}$ behaves in the same manner as $\mathcal{CH}$ in the definition of $\mathtt{Game}_0$ except that $\mathcal{S}$ queries $\mathbb{W}$ to the oracle **Reveal** in the game for HtC-SK property of Σ_{ABX} to acquire the secret-key sk for the attribute $\mathbb{W}$.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\mathcal{L}_{\phi^*})$ and returns it, where $\mathcal{L}_{\phi^*} := \bigcup_{\mathbb{W} \text{ s.t. } \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{S}$ sets $sk^* := \mathrm{LPKE.Dec}(dk, C^*, m^*\|\phi^*)$. $\mathcal{S}$ outputs (ϕ^*, sk^*) if it holds that $[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)] \bigwedge_{sk \in \mathcal{L}_{\phi^*}} [0 \leftarrow \mathrm{ABX.SKVer2}(pk', sk^*, sk)]$.

It is obvious that $\mathcal{S}$ simulates $\mathtt{Game}_2$ or $\mathtt{Game}_3$ for $\mathcal{A}$ perfectly.

By the definitions of W_2 and W_3, we obtain

$$\begin{aligned}
\Pr[W_2] &= \Pr\big[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)]\big] \\
\Pr[W_3] &= \Pr\big[[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \\
&\quad \wedge [1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge \left[\bigvee_{sk \in \mathcal{L}_{\phi^*}} [1 \leftarrow \mathrm{ABX.SKVer2}(pk', sk^*, sk)]\right]\Bigg]
\end{aligned}$$

Hence, we obtain

$$\begin{aligned}
|\Pr[W_2] - \Pr[W_3]| = \Pr\big[&[1 \leftarrow \mathrm{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \\
\wedge [1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)] &\bigwedge_{sk \in \mathcal{L}_{\phi^*}} [0 \leftarrow \mathrm{ABX.SKVer2}(pk', sk^*, sk)]\Bigg]
\end{aligned} \tag{5.10}$$

The above probability is equal to the probability by whom $\mathcal{S}$ wins the HtC-SK property game for Σ_{ABX}. □

Proof of Lemma 5.4.4. We prove that if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_3]-\Pr[W_4]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the zero-knowledge property for Σ_{NIZK}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the zero-knowledge property for the NIZK scheme Σ_{NIZK}. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given crs of Σ_{NIZK}. $\mathcal{S}$ runs $(pk', mk') \leftarrow \text{ABX.Setup}(1^\lambda, 1^L)$ and $(ek, dk) \leftarrow \text{LPKE.Gen}(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the normal manner. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** ϕ is represented as a circuit $\{L, N, I_1, I_2\}$. $\mathcal{S}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$. $\mathcal{S}$ generates $C := \text{LPKE.Enc}(ek, sk, m\|\phi)$. $\mathcal{CH}$ sets $x := (C, m, \phi)$ and $w := sk$, then issues (x, w) to the oracle O_{zk}, then receives a proof π. After that, $\mathcal{CH}$ returns a signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\mathcal{L}_{\phi^*})$ and returns it, where $\mathcal{L}_{\phi^*} := \bigcup_{\mathbb{W} \ s.t.\ \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{S}$ sets $sk^* := \text{LPKE.Dec}(dk, C^*, m^*\|\phi^*)$. $\mathcal{S}$ outputs 1 if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{sk \in \mathcal{L}_{\phi^*}} [1 \leftarrow \text{ABX.SKVer2}(pk', sk^*, sk)]]$.

It is obvious that if the common reference string crs is generated by $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$ (resp. $crs \leftarrow \text{NIZK.Gen}(1^\lambda)$) and the proof-generation oracle O_{zk} is $O_1^{crs,td}$ (resp. O_0^{crs}), then $\mathcal{S}$ simulates Game_4 (resp. Game_3) for $\mathcal{A}$ perfectly, and if and only if W_4 (resp. W_3) occurs, $\mathcal{S}$ outputs 1. Hence, we obtain

$$\begin{aligned}|\Pr[W_3] - \Pr[W_4]| = \Big|&\Pr\left[1 \leftarrow \mathcal{S}^{O_0^{crs}(x,w)}(crs) \mid crs \leftarrow \text{NIZK.Gen}(1^\lambda)\right] \\ &- \Pr\left[1 \leftarrow \mathcal{S}^{O_1^{crs,td}(x,w)}(crs) \mid (crs, td) \leftarrow \mathcal{S}_1(1^\lambda)\right]\Big|.\end{aligned}$$

□

<u>**Proof of Lemma 5.4.5.**</u> We prove that for any $i \in [1, q_s]$ if we assume that there is a PPT adversary $\mathcal{A}$ which makes $|\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$ non-negligible, then we are able to construct a PPT algorithm which breaks the IND-wLCCA security for Σ_{LPKE}.

We consider a PPT simulator $\mathcal{S}$ attempting to break the IND-wLCCA security for the LPKE scheme Σ_{LPKE}. $\mathcal{CH}$ denotes the challenger in the IND-wLCCA security game. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given ek of Σ_{LPKE}. $\mathcal{S}$ runs $(pk', mk') \leftarrow \text{ABX.Setup}(1^\lambda, 1^L)$ and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the usual way. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(\mathbb{W} \in \mathcal{U}, \hat{i} \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{CH}$ retrieves the $\hat{i}$-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$. Suppose that this query is the j-th signing oracle query.

If $j \leq i - 1$ (resp. $j \geq i + 1$), then $\mathcal{S}$ generates the ciphertext C as $C := \text{LPKE.Enc}(ek, 0^{|sk|}, m\|\phi)$ (resp. $C := \text{LPKE.Enc}(ek, sk, m\|ID)$).

If $j = i$, then $\mathcal{S}$ issues $(0^{|sk|}, sk, m||\phi)$ as a challenge query in the IND-wLCCA game for Σ_{LPKE} to acquire a ciphertext C.

After acquiring the ciphertext C, $\mathcal{S}$ behaves as follows. $\mathcal{S}$ sets $x := (C, m, \phi)$, then generates $\pi \leftarrow \mathcal{S}_2(crs, x, td)$. After that, $\mathcal{S}$ returns the signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{S}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda))$**.** $\mathcal{CH}$ computes $f(\mathcal{L}_{\phi^*})$ and returns it, where $\mathcal{L}_{\phi^*} := \bigcup_{\mathbb{W} \text{ s.t. } \phi^*(\mathbb{W})=1} \mathcal{L}_{\mathbb{W}}$.

Forgery(σ^*, m^*)**.** σ^* is parsed as (C^*, π^*). The statement x^* is set to (C^*, m^*, ϕ^*). $\mathcal{S}$ issues $(C^*, m^*||\phi^*)$ as a query to the decryption oracle at **Query 2** in the IND-wLCCA game for Σ_{LPKE}, and sk^* is returned. $\mathcal{S}$ outputs 1 if it holds that $[1 \leftarrow \text{NIZK.Ver}(crs, x^*, \pi^*)] \wedge [(m^*, \phi^*) \notin \mathcal{L}_S] \wedge [1 \leftarrow \text{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{sk \in \mathcal{L}_{\phi^*}} [1 \leftarrow \text{ABX.SKVer2}(pk', sk^*, sk)]]$.

Let $\beta \in \{0, 1\}$ be the challenge-bit in the IND-wLCCA security game for Σ_{LPKE}. It is obvious that $\mathcal{S}$ simulates $\texttt{Game}_{4|i-1}$ (resp. $\texttt{Game}_{4|i}$) when $\beta = 1$ (resp. $\beta = 0$), and if and only if $W_{4|i-1}$ (resp. $W_{4|i}$) happens, $\mathcal{S}$ outputs $\beta' = 1$. It is also obvious that when $W_{4|i-1}$ or $W_{4|i}$ occurs, the label $m^*||\phi^*$ in the query $(C^*, m^*||\phi^*)$ to the oracle **Decrypt** at **Query 2** issued by $\mathcal{S}$ satisfies $(m^*, \phi^*) \neq (m_i, \phi_i)$, where (m_i, ϕ_i) is the pair of the message and predicate issued as the i-th signing oracle query by $\mathcal{A}$, so the query $(C^*, m^*||\phi^*)$ is not a forbidden query. Hence, we obtain $\Pr[W_{4|i-1}] = \Pr[\beta' = 1 \mid \beta = 1]$ and $\Pr[W_{4|i}] = \Pr[\beta' = 1|\beta = 0]$.

It holds that $\Pr[\beta' = \beta] = \Pr[\beta' = 0 \wedge \beta = 0] + \Pr[\beta' = 1 \wedge \beta = 1] = \frac{1}{2}(\Pr[\beta' = 0|\beta = 0] + \Pr[\beta' = 1|\beta = 1]) = \frac{1}{2}(\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0] + 1)$.

Hence, we obtain $\texttt{Adv}^{IND-wLCCA}_{\Sigma_{\text{LPKE}}, \mathcal{S}} = |2 \cdot \Pr[\beta' = \beta] - 1| = |\Pr[\beta' = 1|\beta = 1] - \Pr[\beta' = 1|\beta = 0]| = |\Pr[W_{4|i-1}] - \Pr[W_{4|i}]|$. □

Proof of Lemma 5.4.6. We prove that if we assume that there exists a PPT adversary $\mathcal{A}$ which wins $\texttt{Game}_{4|q_s}$ with a non-negligible advantage, then we can construct a PPT simulator $\mathcal{S}$ which wins $\texttt{Game}_5$ with a non-negligible advantage.

We consider a PPT simulator $\mathcal{S}$ which behaves as an adversary in $\texttt{Game}_5$. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{S}$ is given $(pk', mk', ek, dk, crs, td)$, which were generated by $(pk', mk') \leftarrow \text{ABX.Setup}(1^\lambda, 1^L)$, $(ek, dk) \leftarrow \text{LPKE.Gen}(1^\lambda)$, and $(crs, td) \leftarrow \mathcal{S}_1(1^\lambda)$. pk is set to $pk := (pk', ek, crs)$. $\mathcal{S}$ sends pk to $\mathcal{A}$. $\mathcal{S}$ initializes $\mathcal{L}_S$ as $\emptyset$.

Query. When $\mathcal{A}$ queries to either one of the oracles **Generate** and **Reveal**, $\mathcal{S}$ behaves in the normal manner. When $\mathcal{A}$ queries to **Sign**, $\mathcal{S}$ behaves as follows.

Sign$(\mathbb{W} \in \mathcal{U}, i \in \mathbb{N}, m \in \mathcal{M}, \phi)$**:** $\mathcal{CH}$ retrieves the i-th secret-key sk from $\mathcal{L}_{\mathbb{W}}$. $\mathcal{S}$ generates the ciphertext C as $C := \text{LPKE.Enc}(ek, 0^{|sk|}, m||\phi)$. $\mathcal{S}$ sets $x := (C, m, \phi)$, then generates $\pi \leftarrow \mathcal{S}_2(crs, x, td)$. After that, $\mathcal{S}$ returns the signature $\sigma := (C, \pi)$ to $\mathcal{A}$. After that, $\mathcal{S}$ sets $\mathcal{L}_S := \mathcal{L}_S \cup \{(m, \phi)\}$.

Leak$(\phi^* \in \mathcal{P}, f \in \mathcal{F}_{\Sigma_{\text{ABS}}}(\lambda))$**.** We generate a multiset $\mathcal{L}^\spadesuit$ as follows: initialize it as $\emptyset$, then for every attribute $\mathbb{W} \in \mathcal{U}$ such that $\phi^*(\mathbb{W}) = 1$ and list of secret-keys $\mathcal{L}_{\mathbb{W}}$ for the attribute $\mathbb{W}$ has already been generated, add $|\mathcal{L}_{\mathbb{W}}| \in \mathbb{N}$ number of the attribute $\mathbb{W}$ to $\mathcal{L}^\spadesuit$.

$\mathcal{S}$ issues $(\phi^*, \mathcal{L}^\spadesuit, f)$ as a query to the oracle **Leak** in $\texttt{Game}_5$, then receives $f(\mathcal{L}'_{\mathbb{W}})$. $\mathcal{S}$ sends $f(\mathcal{L}'_{\mathbb{W}})$ to $\mathcal{A}$.

Forgery(σ^*, m^*). $\mathcal{S}$ outputs (σ^*, m^*) at **Forgery** in $\texttt{Game}_5$.

Obviously, $\mathcal{S}$ perfectly simulates $\texttt{Game}_{4|q_s}$ for $\mathcal{A}$. Also obviously, if $\mathcal{A}$ wins $\texttt{Game}_{4|q_s}$, $\mathcal{S}$ wins $\texttt{Game}_5$. Hence, $\Pr[W_{4|q_s}] \leq \Pr[W_5]$. □

Proof of Lemma 5.4.7. Let us consider a PPT adversary $\mathcal{A}$ in $\texttt{Game}_5$. We define the following three events.

$U^{pk',mk',ek,dk,crs,td}$ is the event where (pk', mk'), (ek, dk) and (crs, td) are randomly generated at **Setup** by ABX.Setup$(1^\lambda, 1^L)$, LPKE.Gen(1^λ) and $\mathcal{S}_1(1^\lambda)$, respectively. $V^{\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}_{pk',mk',ek,dk,crs,td}$ is the event where $\mathcal{A}$, given the variables $(pk', mk', ek, dk, crs, td)$ at **Setup**, chooses a predicate $\phi^* \in \mathcal{P}$, attributes $\mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^* \in \mathcal{U}$ *s.t.* $\phi^*(\mathbb{W}_1^*) = \cdots = \phi^*(\mathbb{W}_\tau^*) = 1$ for integer $\tau \in \mathbb{N}$, and a function f at **Leak**. $W_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}$ is the event where $\mathcal{A}$ wins the following game $\texttt{Game}_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}$.

Setup. $\mathcal{A}$ is given $(pk', mk', ek, dk, crs, td, \phi^*, \mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^*, f, f(\{sk_i^*\}_{i\in[1,\tau]}))$, where, for $i \in [1, \tau]$, $sk_i^* \leftarrow$ ABX.KeyGen$(pk', mk', \mathbb{W}_i^*)$.

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). sk^* is set to LPKE.Dec$(dk, C^*, m^* \| \phi^*)$. $\mathcal{A}$ wins the game if it holds that $[1 \leftarrow$ ABX.SKVer$(pk', sk^*, \phi^*)] \wedge [\bigvee_{i\in[1,k^*]} [1 \leftarrow$ ABX.SKVer2$(pk', sk^*, sk_i^*)]]$.

By the definition of W_5 and the definitions of the above three events, we obtain

$$\Pr[W_5] = \sum_{(pk',mk',ek,dk,crs,td)} \sum_{(\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f)} \Pr\left[W_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}\right] \cdot \Pr\left[V^{\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}_{pk',mk',ek,dk,crs,td}\right] \cdot \Pr\left[U^{pk',mk',ek,dk,crs,td}\right].$$

By the above equation and Lemma 5.4.8, there exists a negligible function $\epsilon(\lambda)$ such that

$$\Pr[W_5] < \sum_{(pk',mk',ek,dk,crs,td)} \sum_{(\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f)} \epsilon(\lambda) \cdot \Pr\left[V^{\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}_{pk',mk',ek,dk,crs,td}\right] \cdot \Pr\left[U^{pk',mk',ek,dk,crs,td}\right]. \tag{5.11}$$

We give two facts. It obviously holds that $\sum_{(pk',mk',ek,dk,crs,td)} \Pr\left[U^{pk',mk',ek,dk,crs,td}\right] = 1$. It also obviously holds that for any $(pk', mk', ek, dk, crs, td)$, $\sum_{(\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f)} \Pr\left[V^{\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}_{pk',mk',ek,dk,crs,td}\right] = 1$. By these facts and (5.11), we obtain $\Pr[W_5] < \epsilon(\lambda)$. □

Lemma 5.4.8. *For any PPT $\mathcal{A}$, any (pk', mk'), any (ek, dk), any (crs, td), any $\phi^* \in \mathcal{P}$, any $\mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^* \in \mathcal{U}$ s.t. $\phi^*(\mathbb{W}_1^*) = \cdots = \phi^*(\mathbb{W}_\tau^*) = 1$ for integer $\tau \in \mathbb{N}$, and any $f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$, $\Pr\left[W_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}\right]$ is negligible.*

Proof of Lemma 5.4.8. We prove that if we assume that there exists a PPT adversary $\mathcal{A}$ which wins the game $\mathtt{Game}_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}$ with a non-negligible probability, we can construct a PPT simulator which leads us to a contradiction to the fact that $f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$. Specifically, $\mathcal{S}$ behaves as follows.

Setup. $\mathcal{A}$ is given $(pk', mk', ek, dk, crs, td, \phi^*, \mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^*, f, f(\{sk_i^*\}_{i\in[1,\tau]}))$, where, for $i \in [1,\tau]$, $sk_i^* \leftarrow \mathrm{ABX.KeyGen}(pk', mk', \mathbb{W}_i^*)$. $\mathcal{S}$ gives the variables to $\mathcal{A}$.

Forgery(σ^*, m^*). σ^* is parsed as (C^*, π^*). sk^* is set to $\mathrm{LPKE.Dec}(dk, C^*, m^*\|\phi^*)$. $\mathcal{S}$ outputs sk^*.

It is obvious that $\mathcal{S}$ simulates the game $\mathtt{Game}_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}$ for $\mathcal{A}$ perfectly. If $\mathcal{A}$ wins the game, then $\mathcal{S}$ is able to acquire a secret-key sk^* which satisfies $[1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge [\bigvee_{i\in[1,\tau]}[1 \leftarrow \mathrm{ABX.SKVer2}(pk', sk^*, sk_i^*)]]$. Hence, we obtain

$$\begin{aligned}&\Pr\Big[\mathcal{S}(pk', mk', ek, dk, crs, td, \phi^*, \mathbb{W}_1^*, \cdots, \mathbb{W}_\tau^*, f, f(\{sk_i^*\}_{i\in[1,\tau]})) \rightarrow sk^* \\ &\quad \text{s.t. } [1 \leftarrow \mathrm{ABX.SKVer}(pk', sk^*, \phi^*)] \wedge \Big[\bigvee_{i\in[1,\tau]}[1 \leftarrow \mathrm{ABX.SKVer2}(pk', sk^*, sk_i^*)]\Big]\Big] \\ &= \Pr\left[W_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}\right],\end{aligned}$$

where, for $i \in [1,\tau]$, $sk_i^* \leftarrow \mathrm{ABX.KeyGen}(pk', mk', \mathbb{W}_i^*)$. Assuming that there exists a polynomial function $poly(\lambda)$ such that $\Pr[W_{pk',mk',ek,dk,crs,td,\phi^*,\mathbb{W}_1^*,\cdots,\mathbb{W}_\tau^*,f}] \geq 1/poly(\lambda)$ leads us to a contradiction to the fact that $f \in \mathcal{F}_{\Sigma_{\mathrm{ABS}}}(\lambda)$. □

5.4.3 Proof of Computational Signer-Privacy

Previous ABS schemes [MPR11, OT11, SAH16] were proven to be perfectly signer-private. On the other hand, it must be hard to prove that our ABS scheme Σ_{ABS} is perfectly signer-private since any signature on any predicate ϕ generated by Σ_{ABS} includes a ciphertext of a secret-key for an attribute satisfying the predicate. So, we prove the following theorem which guarantees that Σ_{ABS} is computationally signer-private.

Theorem 5.4.2. *If Σ_{LPKE} is IND-wLCCA and Σ_{NIZK} is zero-knowledge, then Σ_{ABS} is computationally signer-private.*

Proof of Theorem 5.4.2. We define six experiments $\mathtt{Expt}_i$, where $i \in \{0,1,2,3,4,5\}$, to prove the theorem. Hereafter, $\mathcal{A}_1$ and $\mathcal{A}_2$ denote PPT adversaries, $\mathcal{S}_1$ and $\mathcal{S}_2$ denote PPT simulators, $\mathcal{B}_1$ and $\mathcal{B}_2$ denote PPT simulators which makes Σ_{NIZK} satisfy the definition of zero-knowledge, and h_1, h_2 are polynomial time computable functions.

$\mathtt{Expt}_0$, given in Fig.5.3, is the experiment $\mathtt{Expt}^{\mathtt{CSP-0}}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(1^\lambda, 1^L)$ w.r.t. PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and functions h_1 and h_2. $\mathtt{Expt}_5$, given in Fig.5.3, is the experiment $\mathtt{Expt}^{\mathtt{CSP-1}}_{\Sigma_{\mathrm{ABS}},\mathcal{S},h_1,h_2}(1^\lambda, 1^L)$ w.r.t. PPT simulators $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$, and functions h_1 and h_2. Let us consider a PPT distinguisher $\mathcal{D}$, whose advantage is defined as follows: $\mathtt{Adv}^{\mathtt{CSP}}_{\Sigma_{\mathrm{ABS}},\mathcal{D},\mathcal{A},\mathcal{S},h}(\lambda) := |\Pr[\mathcal{D}(\mathtt{Expt}_0(1^\lambda, 1^L)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_5(1^\lambda, 1^L)) \rightarrow 1]|$.

We define the other experiments $\mathtt{Expt}_i$, where $i \in \{1,2,3,4\}$. We describe each experiment in Fig.5.4 and Fig.5.5. Specifically, each experiment is defined as follows.

```
$\mathtt{Expt}_0(1^\lambda, 1^L)(= \mathtt{Expt}^{\mathsf{CSP\text{-}0}}_{\Sigma_{\mathrm{ABS}},\mathcal{A},h_1,h_2}(1^\lambda, 1^L))$:
  $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
  $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
  $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$
  $pk := (pk', ek, crs),\ mk := mk'$
  $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O^{pk,mk}_{CSP}(\mathbb{W})}(pk)$,
    where $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
  $\mathbb{W}^* \overset{U}{\leftarrow} \mathcal{K}^*,\ sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
  $C^* \leftarrow \mathsf{LPKE.Enc}(ek, sk^*, m^* \| \phi^*)$
  $x^* := (C^*, m^*, \phi^*),\ w := sk^*$
  $\pi^* \leftarrow \mathsf{NIZK.Pro}(crs, x^*, w),\ \sigma^* := (C^*, \pi^*)$
  $v \leftarrow \mathcal{A}_2^{O^{pk,mk}_{CSP}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
  Return $(d, \mathcal{K}^*)$.
```

```
$\mathtt{Expt}_5(1^\lambda, 1^L)(= \mathtt{Expt}^{\mathsf{CSP\text{-}1}}_{\Sigma_{\mathrm{ABS}},\mathcal{S},h_1,h_2}(1^\lambda, 1^L))$:
  $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
  $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
  $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$
  $pk := (pk', ek, crs),\ mk := mk'$
  $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{S}_1(pk)$,
    where $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
  $\mathbb{W}^* \overset{U}{\leftarrow} \mathcal{K}^*$
  $v \leftarrow \mathcal{S}_2(st, h_1(\mathbb{W}^*))$
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
  Return $(d, \mathcal{K}^*)$.
```

Figure 5.3: Experiments $\mathtt{Expt}_0$ and $\mathtt{Expt}_5$.

- $\mathtt{Expt}_1$ is the same as $\mathtt{Expt}_0$ except that the common reference string crs is generated by running $crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)$.
- $\mathtt{Expt}_2$ is the same as $\mathtt{Expt}_1$ except that the common reference string crs is generated by running $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$ and the NIZK-proof π^* in the signature $\sigma^* = (C^*, \pi^*)$ is generated by running $\pi^* \leftarrow \mathcal{B}_2(crs, x^*, td)$.
- $\mathtt{Expt}_3$ is the same as $\mathtt{Expt}_2$ except that the LPKE-ciphertext C^* in the signature σ^* is generated from a randomly generated secret-key for an attribute $\mathbb{W}^\spadesuit$ which is randomly chosen independently of the attribute $\mathbb{W}^*$.
- $\mathtt{Expt}_4$ is basically the same as $\mathtt{Expt}_3$. In the experiment, we generate not only the pair of (pk, mk), but also another pair of $(\widetilde{pk}, \widetilde{mk})$. After generating them, we use only the latter pair. For instance, the public-key $\widetilde{pk}$ is given to the adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, and the key-revelation oracle $O^{\widetilde{pk},\widetilde{mk}}_{CSP}(\widetilde{\mathbb{W}})$ generates a secret-key for an attribute $\widetilde{\mathbb{W}}$ by using the master-key $\widetilde{mk}$ and returns it. Every variable which is dependent on the key-pair $(\widetilde{pk}, \widetilde{mk})$ is given the wide tilde symbol ($\widetilde{\ }$).

We obtain the following inequality: $\mathsf{Adv}^{\mathsf{CSP}}_{\Sigma_{\mathrm{ABS}},\mathcal{D},\mathcal{A},\mathcal{S},h_1,h_2}(\lambda) \leq \sum_{i=1}^{5} |\Pr[\mathcal{D}(\mathtt{Expt}_{i-1}(1^\lambda, 1^L)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_i(1^\lambda, 1^L)) \rightarrow 1]|$.

Theorem 5.4.2 is proven true by the above inequality and the following 5 lemmas. □

Lemma 5.4.9. *If Σ_{NIZK} is ZK, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\mathtt{Expt}_0(1^\lambda, 1^L)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_1(1^\lambda, 1^L)) \rightarrow 1]|$ is negligible.*

Lemma 5.4.10. *If Σ_{NIZK} is ZK, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\mathtt{Expt}_1(1^\lambda, 1^L)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_2(1^\lambda, 1^L)) \rightarrow 1]|$ is negligible.*

Lemma 5.4.11. *If Σ_{LPKE} is IND-wLCCA secure, then $\forall\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall\mathcal{D}$, $|\Pr[\mathcal{D}(\mathtt{Expt}_2(1^\lambda, 1^L)) \rightarrow 1] - \Pr[\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L)) \rightarrow 1]|$ is negligible.*

```
$\mathtt{Expt}_1(1^\lambda, 1^L)$:
    $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
    $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $\underline{crs \leftarrow \mathsf{NIZK.Gen}(1^\lambda)}$
    $pk := (pk', ek, crs),\ mk := mk'$
    $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$,
      where $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
    $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*,\ sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
    $C^* \leftarrow \mathsf{LPKE.Enc}(ek, sk^*, m^* || \phi^*)$
    $x^* := (C^*, m^*, \phi^*),\ w := sk^*$
    $\pi^* \leftarrow \mathsf{NIZK.Pro}(crs, x^*, w),\ \sigma^* := (C^*, \pi^*)$
    $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
    If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
    Return $(d, \mathcal{K}^*)$.
```

```
$\mathtt{Expt}_2(1^\lambda, 1^L)$:
    $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
    $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $\underline{(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)}$
    $pk := (pk', ek, crs),\ mk := mk'$
    $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$,
      where $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
    $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*,\ sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
    $C^* \leftarrow \mathsf{LPKE.Enc}(ek, sk^*, m^* || \phi^*)$
    $x^* := (C^*, m^*, \phi^*),\ w := sk^*$
    $\underline{\pi^* \leftarrow \mathcal{B}_2(crs, x^*, td)},\ \sigma^* := (C^*, \pi^*)$
    $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
    If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
    Return $(d, \mathcal{K}^*)$.
```

Figure 5.4: Experiments $\mathtt{Expt}_1$ and $\mathtt{Expt}_2$. The underlined parts indicate the parts which are changed from the previous experiment.

```
$\mathtt{Expt}_3(1^\lambda, 1^L)$:
    $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
    $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$
    $pk := (pk', ek, crs),\ mk := mk'$
    $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$,
      where $\mathcal{K}^* = \{\mathbb{W} | \mathbb{W} \in \mathcal{U} \wedge \phi^*(\mathbb{W}) = 1\}$.
    $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$
    $\mathbb{W}^\spadesuit \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*,\ sk^\spadesuit \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^\spadesuit)$
    $C^* \leftarrow \mathsf{LPKE.Enc}(ek, sk^\spadesuit, m^* || \phi^*)$
    $x^* := (C^*, m^*, \phi^*),\ w := sk^\spadesuit$
    $\pi^* \leftarrow \mathcal{B}_2(crs, x^*, td),\ \sigma^* := (C^*, \pi^*)$
    $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
    If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$.
    Return $(d, \mathcal{K}^*)$.
```

```
$\mathtt{Expt}_4(1^\lambda, 1^L)$:
    $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
    $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$
    $pk := (pk', ek, crs),\ mk := mk'$
    $(\widetilde{ek}, \widetilde{dk}) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$
    $(\widetilde{pk}', \widetilde{mk}') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $(\widetilde{crs}, \widetilde{td}) \leftarrow \mathcal{B}_1(1^\lambda)$
    $\widetilde{pk} := (\widetilde{pk}', \widetilde{ek}, \widetilde{crs}),\ \widetilde{mk} := \widetilde{mk}'$
    $(\widetilde{\mathcal{K}}^*, \widetilde{\phi}^* \in \widetilde{\mathcal{P}}, \widetilde{m}^* \in \widetilde{\mathcal{M}}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{\widetilde{pk},\widetilde{mk}}(\widetilde{\mathbb{W}})}(\widetilde{pk})$,
      where $\widetilde{\mathcal{K}}^* = \{\widetilde{\mathbb{W}} | \widetilde{\mathbb{W}} \in \widetilde{\mathcal{U}} \wedge \widetilde{\phi}^*(\widetilde{\mathbb{W}}) = 1\}$.
    $\widetilde{\mathbb{W}}^* \overset{\mathrm{U}}{\leftarrow} \widetilde{\mathcal{K}}^*$
    $\widetilde{\mathbb{W}}^\spadesuit \overset{\mathrm{U}}{\leftarrow} \widetilde{\mathcal{K}}^*,\ \widetilde{sk}^\spadesuit \leftarrow \mathsf{ABX.KeyGen}(\widetilde{pk}', \widetilde{mk}', \widetilde{\mathbb{W}}^\spadesuit)$
    $\widetilde{C}^* \leftarrow \mathsf{LPKE.Enc}(\widetilde{ek}, \widetilde{sk}^\spadesuit, \widetilde{m}^* || \widetilde{\phi}^*)$
    $\widetilde{x}^* := (\widetilde{C}^*, \widetilde{m}^*, \widetilde{\phi}^*),\ \widetilde{w} := \widetilde{sk}^\spadesuit$
    $\widetilde{\pi}^* \leftarrow \mathcal{B}_2(\widetilde{crs}, \widetilde{x}^*, \widetilde{td}),\ \widetilde{\sigma}^* := (\widetilde{C}^*, \widetilde{\pi}^*)$
    $\widetilde{v} \leftarrow \mathcal{A}_2^{O_{CSP}^{\widetilde{pk},\widetilde{mk}}(\widetilde{\mathbb{W}})}(st, h_1(\widetilde{\mathbb{W}}^*), \widetilde{\sigma}^*)$
    If $\bar{v} = h_2(\widetilde{\mathbb{W}}^*)$, then $d := 1$. Else, then $d := 0$.
    Return $(d, \mathcal{K}^*)$.
```

Figure 5.5: Experiments $\mathtt{Expt}_3$ and $\mathtt{Expt}_4$.

```
$\underline{\mathcal{A}^*(crs)}$:
  $(ek, dk) \leftarrow \mathsf{LPKE.Gen}(1^\lambda)$, $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
  $pk := (pk', ek, crs)$, $mk := mk'$, $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$
  $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
  $C^* \leftarrow \mathsf{LPKE.Enc}(ek, sk^*, m^* \| \phi^*)$
  $x^* := (C^*, m^*, \phi^*)$, $w := sk^*$, $\pi^* \leftarrow \mathsf{NIZK.Pro}(crs, x^*, w)$, $\sigma^* := (C^*, \pi^*)$
  $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$. Return $\mathcal{D}(d, \mathcal{K}^*)$.
```

Figure 5.6: PPT adversary $\mathcal{A}^*$ which is used in the proof of Lemma 5.4.9

Lemma 5.4.12. *$\forall \mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\forall \mathcal{D}$,* $\Pr[\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L)) \to 1] = \Pr[\mathcal{D}(\mathtt{Expt}_4(1^\lambda, 1^L)) \to 1]$.

Lemma 5.4.13. *$\forall \mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, $\forall h_1$, $\forall h_2$, $\exists \mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ s.t. $\forall \mathcal{D}$,* $\Pr[\mathcal{D}(\mathtt{Expt}_4(1^\lambda, 1^L)) \to 1] = \Pr[\mathcal{D}(\mathtt{Expt}_5(1^\lambda, 1^L)) \to 1]$.

Proof of each lemma is given below.

Proof of Lemma 5.4.9. We prove the lemma by contradiction. We show that assuming that the lemma doesn't hold leads us to a contradiction to the fact that Σ_{NIZK} is zero-knowledge.

If we assume that the lemma doesn't hold, we can say that there exist a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, functions h_1, h_2, a PPT distinguisher $\mathcal{D}$, and a polynomial function $poly(\lambda)$ such that the following inequality holds.

$$\left|\Pr\left[\mathcal{D}(\mathtt{Expt}_0(1^\lambda, 1^L)) \to 1\right] - \Pr\left[\mathcal{D}(\mathtt{Expt}_1(1^\lambda, 1^L)) \to 1\right]\right| \geq 1/poly(\lambda). \tag{5.12}$$

Let us consider a PPT adversary $\mathcal{A}^*$ attempting to break the zero-knowldge property of Σ_{NIZK}. $\mathcal{A}^*$ uses $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2), h_1, h_2$ and $\mathcal{D}$ defined above. $\mathcal{A}^*$ behaves as shown in Fig.5.6. The common reference string given to $\mathcal{A}^*$ is generated by running $\mathsf{NIZK.Gen}(1^\lambda) \to crs$ or $\mathcal{B}_1(1^\lambda) \to (crs, td)$. If $\mathbb{W} \in \mathcal{U}$ is issued as a query to oracle $O_{CSP}^{pk,mk}$ by $\mathcal{A}_1$ or $\mathcal{A}_2$, then $\mathcal{A}^*$ computes $\mathsf{ABX.KeyGen}(pk', mk', \mathbb{W})$ and returns it. When crs is generated by running $\mathsf{NIZK.Gen}(1^\lambda) \to crs$ (resp. $\mathcal{B}_1(1^\lambda) \to (crs, td)$), $\mathcal{A}^*$ perfectly simulates $\mathtt{Expt}_1(1^\lambda, 1^L)$ (resp. $\mathtt{Expt}_0(1^\lambda, 1^L)$) for $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{D}$. Hence, we obtain the following equations.

$$\Pr\left[\mathcal{A}^*(crs) \to 1 \,\middle|\, \mathsf{NIZK.Gen}(1^\lambda) \to crs\right] = \Pr\left[\mathcal{D}(\mathtt{Expt}_1(1^\lambda, 1^L)) \to 1\right] \tag{5.13}$$

$$\Pr\left[\mathcal{A}^*(crs) \to 1 \,\middle|\, \mathcal{B}_1(1^\lambda) \to (crs, td)\right] = \Pr\left[\mathcal{D}(\mathtt{Expt}_0(1^\lambda, 1^L)) \to 1\right] \tag{5.14}$$

By (5.12), (5.13) and (5.14), we obtain the following inequality.

$$\left|\Pr\left[\mathcal{A}^*(crs) \to 1 \,\middle|\, \mathsf{NIZK.Gen}(1^\lambda) \to crs\right] - \Pr\left[\mathcal{A}^*(crs) \to 1 \,\middle|\, \mathcal{B}_1(1^\lambda) \to (crs, td)\right]\right| \\ \geq 1/poly(\lambda)$$

□

```
$\mathcal{A}^*(crs)$:
  $(ek, dk) \leftarrow \text{LPKE.Gen}(1^\lambda)$, $(pk', mk') \leftarrow \text{ABX.Setup}(1^\lambda, 1^L)$
  $pk := (pk', ek, crs)$, $mk := mk'$, $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$
  $\mathbb{W}^* \overset{\text{U}}{\leftarrow} \mathcal{K}^*$, $sk^* \leftarrow \text{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
  $C^* \leftarrow \text{LPKE.Enc}(ek, sk^*, m^*\|\phi^*)$
  $x^* := (C^*, m^*, \phi^*)$, $w := sk^*$, $\pi^* \leftarrow O^{ZK}(x^*, w)$, $\sigma^* := (C^*, \pi^*)$
  $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
  If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$. Return $\mathcal{D}(d, \mathcal{K}^*)$.
```

Figure 5.7: PPT adversary $\mathcal{A}^*$ which is used in the proof of Lemma 5.4.10

Proof of Lemma 5.4.10. We prove the lemma by contradiction. We show that assuming that the lemma doesn't hold true leads us to a contradiction to the fact that Σ_{NIZK} is zero-knowledge.

If we assume that the lemma doesn't hold true, we can say that there exist a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, functions h_1, h_2, a PPT distinguisher $\mathcal{D}$, and a polynomial function $poly(\lambda)$ such that

$$\left|\Pr\left[\mathcal{D}(\texttt{Expt}_1(1^\lambda, 1^L)) \to 1\right] - \Pr\left[\mathcal{D}(\texttt{Expt}_2(1^\lambda, 1^L)) \to 1\right]\right| \geq 1/poly(\lambda). \tag{5.15}$$

Let us consider a PPT adversary $\mathcal{A}^*$ attempting to break the zero-knowledge property of Σ_{NIZK}. $\mathcal{A}^*$ uses $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2), h_1, h_2$ and $\mathcal{D}$ defined above. $\mathcal{A}^*$ behaves as shown in Fig.5.7. The common reference string given to $\mathcal{A}^*$ is generated by running $\text{NIZK.Gen}(1^\lambda) \to crs$ or $\mathcal{B}_1(1^\lambda) \to (crs, td)$. If $\mathbb{W} \in \mathcal{U}$ is issued as a query to oracle $O_{CSP}^{pk,mk}$ by $\mathcal{A}_1$ or $\mathcal{A}_2$, then $\mathcal{A}^*$ computes $\text{ABX.KeyGen}(pk', mk', \mathbb{W})$ and returns it. The oracle $O^{ZK}(x, w)$ which is used by $\mathcal{A}^*$ is equivalent to the oracle O_0^{crs} (resp. $O_1^{crs,td}$) in the definition of zero-knowledge property for Σ_{NIZK} when crs is generated by running $\text{NIZK.Gen}(1^\lambda) \to crs$ (resp. $\mathcal{B}_1(1^\lambda) \to (crs, td)$). $\mathcal{A}^*$ perfectly simulates $\texttt{Expt}_1(1^\lambda, 1^L)$ (resp. $\texttt{Expt}_2(1^\lambda, 1^L)$) for $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{D}$ when crs is generated by running $\text{NIZK.Gen}(1^\lambda) \to crs$ (resp. $\mathcal{B}_1(1^\lambda) \to (crs, td)$). Hence, we obtain the following equations.

$$\Pr\left[\mathcal{A}^{*O_0^{crs}(x,w)}(crs) \to 1 \,\middle|\, \text{NIZK.Gen}(1^\lambda) \to crs\right] = \Pr\left[\mathcal{D}(\texttt{Expt}_1(1^\lambda, 1^L)) \to 1\right] \tag{5.16}$$

$$\Pr\left[\mathcal{A}^{*O_1^{crs,td}(x,w)}(crs) \to 1 \,\middle|\, \mathcal{B}_1(1^\lambda) \to (crs, td)\right] = \Pr\left[\mathcal{D}(\texttt{Expt}_2(1^\lambda, 1^L)) \to 1\right] \tag{5.17}$$

By (5.15), (5.16) and (5.17), we obtain the following inequality.

$$\Bigg|\Pr\left[\mathcal{A}^{*O_0^{crs}(x,w)}(crs) \to 1 \,\middle|\, \text{NIZK.Gen}(1^\lambda) \to crs\right] - \Pr\left[\mathcal{A}^{*O_1^{crs,td}(x,w)}(crs) \to 1 \,\middle|\, \mathcal{B}_1(1^\lambda) \to (crs, td)\right]\Bigg| \geq 1/poly(\lambda)$$

□

Proof of Lemma 5.4.11. We prove the lemma by contradiction. We show that assuming that the lemma does not hold leads us to a contradiction to the fact that Σ_{LPKE} is IND-wLCCA.

If we assume that the lemma does not hold, then we can say that there exist a PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, functions h_1, h_2, a PPT distinguisher $\mathcal{D}$, and a polynomial function $poly(\lambda)$ such that

$$\left|\Pr\left[\mathcal{D}(\mathtt{Expt}_2(1^\lambda, 1^L)) \to 1\right] - \Pr\left[\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L)) \to 1\right]\right| \geq 1/poly(\lambda). \tag{5.18}$$

Let us consider a PPT adversary $\mathcal{A}^* = (\mathcal{A}_1^*, \mathcal{A}_2^*)$ in the experimental definition of IND-wLCCA of Σ_{LPKE} desribed in Subsect. 5.2.1. $\mathcal{A}^*$ uses $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2), h_1, h_2$ and $\mathcal{D}$ defined above. $\mathcal{A}^* = (\mathcal{A}_1^*, \mathcal{A}_2^*)$ behaves as shown in Fig. 5.8. If $\mathbb{W} \in \mathcal{U}_X$ is issued as a query to oracle

```
$\underline{\mathcal{A}_1^*(ek)}$:
    $(crs, td) \leftarrow \mathcal{B}_1(1^\lambda)$, $(pk', mk') \leftarrow \mathsf{ABX.Setup}(1^\lambda, 1^L)$
    $pk := (pk', ek, crs)$, $mk := mk'$, $(\mathcal{K}^*, \phi^* \in \mathcal{P}, m^* \in \mathcal{M}, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{pk,mk}(\mathbb{W})}(pk)$
    $\mathbb{W}^* \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$, $sk^* \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^*)$
    $\mathbb{W}^\spadesuit \overset{\mathrm{U}}{\leftarrow} \mathcal{K}^*$, $sk^\spadesuit \leftarrow \mathsf{ABX.KeyGen}(pk', mk', \mathbb{W}^\spadesuit)$
    $st' := (st, pk, mk, m^*, \phi^*, \mathbb{W}^*, sk^*, sk^\spadesuit)$, Return $(sk^*, sk^\spadesuit, m^* \| \phi^*, st')$
$\underline{\mathcal{A}_2^*(st', C^*)}$:
    $x^* := (C^*, m^*, \phi^*)$, $w := sk^\spadesuit$, $\pi^* \leftarrow \mathcal{B}_2(crs, x^*, td)$
    $\sigma^* := (C^*, \pi^*)$, $v \leftarrow \mathcal{A}_2^{O_{CSP}^{pk,mk}(\mathbb{W})}(st, h_1(\mathbb{W}^*), \sigma^*)$
    If $v = h_2(\mathbb{W}^*)$, then $d := 1$. Else, then $d := 0$. Return $\mathcal{D}(d, \mathcal{K}^*)$.
```

Figure 5.8: PPT adversary $\mathcal{A}^* = (\mathcal{A}_1^*, \mathcal{A}_2^*)$ which is used in the proof of Lemma 5.4.11

$O_{CSP}^{pk,mk}$ by $\mathcal{A}_1$ or $\mathcal{A}_2$, then $\mathcal{A}^*$ computes $\mathsf{ABX.KeyGen}(pk', mk', \mathbb{W})$ and returns it. If C^* is a ciphertext generated by encrypting sk^* (resp. $sk^\spadesuit$), then $\mathcal{A}^*$ perfectly simulates $\mathtt{Expt}_2(1^\lambda, 1^L)$ (resp. $\mathtt{Expt}_3(1^\lambda, 1^L)$) for $\mathcal{A}_1, \mathcal{A}_2$ and $\mathcal{D}$. So, we obtain the following equations.

$$\Pr\left[\mathtt{Expt}_{\Sigma_{\mathrm{LPKE}}, \mathcal{A}^*}^{IND\text{-}wLCCA\text{-}0}(1^\lambda) \to 1\right] = \Pr\left[\mathcal{D}(\mathtt{Expt}_2(1^\lambda, 1^L)) \to 1\right] \tag{5.19}$$

$$\Pr\left[\mathtt{Expt}_{\Sigma_{\mathrm{LPKE}}, \mathcal{A}^*}^{IND\text{-}wLCCA\text{-}1}(1^\lambda) \to 1\right] = \Pr\left[\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L)) \to 1\right] \tag{5.20}$$

By (5.18), (5.19) and (5.20), the following inequality is obtained.

$$\left| \Pr\left[\mathtt{Expt}_{\Sigma_{\mathrm{LPKE}}, \mathcal{A}^*}^{IND\text{-}wLCCA\text{-}0}(1^\lambda) \to 1\right] - \Pr\left[\mathtt{Expt}_{\Sigma_{\mathrm{LPKE}}, \mathcal{A}^*}^{IND\text{-}wLCCA\text{-}1}(1^\lambda) \to 1\right] \right| \geq 1/poly(\lambda) \tag{5.21}$$

□

<u>**Proof of Lemma 5.4.12.**</u> In $\mathtt{Expt}_3$ and $\mathtt{Expt}_4$, the variables $(ek, dk, pk', mk', crs, td)$ and the variables $(\widetilde{ek}, \widetilde{dk}, \widetilde{pk}', \widetilde{mk}', \widetilde{crs}, \widetilde{td})$ identically distribute. Hence, for every PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, every h_1 and every h_2, the output $(d, \mathcal{K}^*)$ of $\mathtt{Expt}_3(1^\lambda, 1^L)$ and the output $(d, \hat{\mathcal{K}}^*)$ of $\mathtt{Expt}_4(1^\lambda, 1^L)$ also identically distribute. Therefore, for every PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, every h_1, every h_2 and every PPT distinguisher $\mathcal{D}$, $\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L))$ and $\mathcal{D}(\mathtt{Expt}_4(1^\lambda, 1^L))$ also indentically distribute, and it holds that $\Pr[\mathcal{D}(\mathtt{Expt}_3(1^\lambda, 1^L)) \to 1] = \Pr[\mathcal{D}(\mathtt{Expt}_4(1^\lambda, 1^L)) \to 1]$. □

```
$\underline{\mathcal{S}_1(pk)}$:
    $(\widetilde{ek}, \widetilde{dk}) \leftarrow$ LPKE.Gen$(1^\lambda)$
    $(\widetilde{pk}', \widetilde{mk}') \leftarrow$ ABX.Setup$(1^\lambda, 1^L)$
    $(\widetilde{crs}, \widetilde{td}) \leftarrow \mathcal{B}_1(1^\lambda)$
    $\widetilde{pk} := (\widetilde{pk}', \widetilde{ek}, \widetilde{crs}), \widetilde{mk} := \widetilde{mk}'$
    $(\widetilde{\mathcal{K}}^*, \widetilde{\phi}^*, \widetilde{m}^*, st) \leftarrow \mathcal{A}_1^{O_{CSP}^{\widetilde{pk},\widetilde{mk}}(\widetilde{\mathbb{W}})}(\widetilde{pk})$
    $st' := (\widetilde{pk}, \widetilde{mk}, \widetilde{td}, st)$
    Return $(\widetilde{\mathcal{K}}^*, \widetilde{\phi}^*, \widetilde{m}^*, st')$.
```

```
$\underline{\mathcal{S}_2(st', h_1(\mathbb{W}^*))}$:
    $st'$ is parsed as $(\widetilde{pk}, \widetilde{mk}, \widetilde{td}, st)$.
    $\widetilde{\mathbb{W}}^* \overset{\mathrm{U}}{\leftarrow} \widetilde{\mathcal{K}}^*$
    $\widetilde{\mathbb{W}}^\spadesuit \overset{\mathrm{U}}{\leftarrow} \widetilde{\mathcal{K}}^*$, $\widetilde{sk}^\spadesuit \leftarrow$ ABX.KeyGen$(\widetilde{pk}', \widetilde{mk}', \widetilde{\mathbb{W}}^\spadesuit)$
    $\widetilde{C}^* \leftarrow$ LPKE.Enc$(\widetilde{ek}, \widetilde{sk}^\spadesuit, \widetilde{m}^* \| \widetilde{\phi}^*)$
    $\widetilde{x}^* := (\widetilde{C}^*, \widetilde{m}^*, \widetilde{\phi}^*), \widetilde{\pi}^* \leftarrow \mathcal{B}_2(\widetilde{crs}, \widetilde{x}^*, \widetilde{td})$
    $\widetilde{\sigma}^* := (\widetilde{C}^*, \widetilde{\pi}^*)$, Return $v := \mathcal{A}_2^{O_{CSP}^{\widetilde{pk},\widetilde{mk}}(\widetilde{\mathbb{W}})}(st, h_1(\widetilde{\mathbb{W}}^*), \widetilde{\sigma}^*)$.
```

Figure 5.9: PPT simulator $\mathcal{S}$ which is used in the proof of Lemma 5.4.13

Proof of Lemma 5.4.13. $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ denotes a PPT adversary in $\mathtt{Expt}_4$. $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ denotes a PPT simulator in $\mathtt{Expt}_5$. $\mathcal{S}_1$ (resp. $\mathcal{S}_2$) uses $\mathcal{A}_1$ (resp. $\mathcal{A}_2$) as a subroutine. If the adversary queries an attribute $\widetilde{\mathbb{W}}$ to the key-reveration oracle, the simulator generates a secret-key for $\widetilde{\mathbb{W}}$ by using the master-key $\widetilde{mk}$ and returns it. If we let $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ behave as shown in Fig.5.9, for every PPT adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, every function h_1, and every function h_2, the output of $\mathtt{Expt}_4(1^\lambda, 1^L)$ and the output of $\mathtt{Expt}_5(1^\lambda, 1^L)$ identically distribute. Hence, for every $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, every h_1 and every h_2, there exists $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for every $\mathcal{D}$, it holds that $\Pr[\mathcal{D}(\mathtt{Expt}_4(1^\lambda, 1^L)) \to 1] = \Pr[\mathcal{D}(\mathtt{Expt}_5(1^\lambda, 1^L)) \to 1]$. □

5.5 Instantiations Secure under Standard Assumptions

In this section, we instantiate our IBS and ABS under the DLIN and SXDH assumptions. On the instantiations, we use EUF-CMA-secure (structure-preserving) signature schemes.

5.5.1 An Instantiation of IBS

As LPKE scheme, we use a modified variant of the LPKE scheme by Camenisch et al. [CCS09] which is IND-LCCA under the DLIN assumption and the collision-resistance of the hash function H_{CL}. For the difference between IND-LCCA notion and IND-wLCCA notion, refer to [FHN+12]. The security of the LPKE scheme Π_{LPKE} is guaranteed by the following theorem.

Theorem 5.5.1. *For any (polynomially bounded) intergers $n, \kappa, \tau \in \mathbb{N}$, $\Pi_{\text{LPKE},n,\kappa,\tau}$ is IND-wLCCA under the DLIN assumption and the collision resistance of the hash function H_{CL} : $\{0,1\}^\tau \to \mathbb{Z}_p$.*

As NIZK scheme, we adopt the NIZK system by Groth and Sahai [GS08]. Thus, we obtain the following theorem.

Theorem 5.5.2. *NIZK system Π_{NIZK} by Groth and Sahai [GS08] is sound and zero-knowledge under the DLIN assumption.*

We use the IBX scheme given in Fig. 5.11. Theorem 5.5.3 guarantees its HtC-SK property. The theorem is proven in the latter half of this subsection.

Theorem 5.5.3. *For any (polynomially bounded) intergers $n \in \mathbb{N}$, $\Sigma_{\mathrm{IBX},n}$ is HtC-SK under the DL assumption and the EUF-CMA security of* Σ_{SPSIG}.

For the (structure-preserving) signature scheme Π_{SPSIG} as a building block in the IBX scheme $\Sigma_{\mathrm{IBX},n}$, we use the scheme by Kiltz et al. [KPW15]. Details of its construction can been seen in [KPW15] or [SAH16]. Actually, in the construction, bilinear pairing with groups of prime order is used. Hereafter, we denote its description by $(p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g})$. According to [KPW15, SAH16], we obtain the following theorem.

Theorem 5.5.4. Π_{SPSIG} *is EUF-CMA secure under the SXDH assumption.*

```
$\underline{\mathsf{LPKE.Gen}(1^\lambda, 1^n, 1^\kappa)}$: $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$, $g_1, \cdots, g_n, h_0, h_1, h_2 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$
 $u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3 \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, $d_1 := h_0^{u_1} \cdot h_1^{u_2}, d_2 := h_0^{u_1} \cdot h_2^{u_3}$
 $e_1 := h_0^{v_1} \cdot h_1^{v_2}, e_2 := h_0^{v_1} \cdot h_2^{v_3}, b_1 := h_0^{w_1} \cdot h_1^{w_2}, b_2 := h_0^{w_1} \cdot h_2^{w_3}$
 $ek := (p, \mathbb{G}, g_1, \cdots, g_n, h_0, h_1, h_2, d_1, d_2, e_1, e_2, b_1, b_2)$
 $dk := (u_1, u_2, u_3, v_1, v_2, v_3, w_1, w_2, w_3)$, Return $(ek, dk)$
$\underline{\mathsf{LPKE.Enc}(ek, x_1 \in \mathbb{Z}_p, \cdots, x_n \in \mathbb{Z}_p, \theta_1 \in \mathbb{G}, \cdots, \theta_\kappa \in \mathbb{G}, L \in \{0,1\}^\tau)\text{:}}$
 For $i \in [1, n]$ and for $j \in [1, \lambda]$, the $j$-th bit of $x_i \in \mathbb{Z}_p$ is denoted by $x_{ij}$.
 For every $i \in [1, n]$ and every $j \in [1, \lambda]$, do:
   $r_{ij}, s_{ij} \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, $\mathbf{y}_{ij} := (y_{ij,1}, y_{ij,2}, y_{ij,3}) := (h_0^{r_{ij}+s_{ij}}, h_1^{r_{ij}}, h_2^{s_{ij}})$, $z_{ij} := b_1^{r_{ij}} \cdot b_2^{s_{ij}} \cdot g_i^{2^j \cdot x_{ij}}$
   $t_{ij} := H_{CL}(\mathbf{y}_{ij}, z_{ij}, L)$, $c_{ij} := (d_1 \cdot e_1^{t_{ij}})^{r_{ij}} \cdot (d_2 \cdot e_2^{t_{ij}})^{s_{ij}}$, $\mathbf{c}_{ij} := (\mathbf{y}_{ij}, z_{ij}, c_{ij})$
 For every $i \in [1, \kappa]$, do:
   $r_i, s_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, $\mathbf{y}_i := (y_{i,1}, y_{i,2}, y_{i,3}) := (h_0^{r_i+s_i}, h_1^{r_i}, h_2^{s_i})$, $z_i := b_1^{r_i} \cdot b_2^{s_i} \cdot \theta_i$
   $t_i := H_{CL}(\mathbf{y}_i, z_i, L)$, $c_i := (d_1 \cdot e_1^{t_i})^{r_i} \cdot (d_2 \cdot e_2^{t_i})^{s_i}$, $\mathbf{c}_i := (\mathbf{y}_i, z_i, c_i)$
 Return $\mathbf{C} := (\{\mathbf{c}_{ij}\}_{i \in [1,n], j \in [1,\lambda]}, \{\mathbf{c}_i\}_{i \in [1,\kappa]})$
$\underline{\mathsf{LPKE.Dec}(ek, dk, \mathbf{C}, L \in \{0,1\}^\tau)\text{:}}$
 For every $i \in [1, n]$ and every $j \in [1, \lambda]$, do:
   $t_{ij} := H_{CL}(\mathbf{y}_{ij}, z_{ij}, L)$, $\tilde{c}_{ij} := y_{ij,1}^{u_1+t_{ij}v_1} \cdot y_{ij,2}^{u_2+t_{ij}v_2} \cdot y_{ij,3}^{u_3+t_{ij}v_3}$
   If $\tilde{c}_{ij} \neq c_{ij}$, then return $\perp$. Else if $z_{ij}/(y_{ij,1}^{w_1} \cdot y_{ij,2}^{w_2} \cdot y_{ij,3}^{w_3}) = g_i^{2^j}$, then $x'_{ij} := 1$. Else, then $x'_{ij} := 0$.
   The $j$-th bit of $x'_i$ is set as $x'_{ij}$.
 For every $i \in [1, n]$, $x'_i := \sum_{j=1}^{\lambda} 2^j \cdot x'_{ij}$
 For every $i \in [1, \kappa]$, do:
   $t_i := H_{CL}(\mathbf{y}_i, z_i, L)$, $\tilde{c}_i := y_{i,1}^{u_1+t_iv_1} \cdot y_{i,2}^{u_2+t_iv_2} \cdot y_{i,3}^{u_3+t_iv_3}$
   If $\tilde{c}_i \neq c_i$, then return $\perp$. Else, then $\sigma'_i := z_i/(y_{i,1}^{w_1} \cdot y_{i,2}^{w_2} \cdot y_{i,3}^{w_3})$.
 Return $(x'_1, \cdots, x'_n, \sigma'_1, \cdots, \sigma'_\kappa)$
```

Figure 5.10: Construction of LPKE scheme $\Pi_{\mathrm{LPKE},n,\kappa,\tau}$, where $n, \kappa, \tau \in \mathbb{N}$ and $H_{CL} : \{0,1\}^* \to \mathbb{Z}_p$ is a collision-resistant hash function.

Since we use Groth-Sahai NIZK system [GS08], we have to compute the commitments to the witness, and then prove that the commitments satisfy some equations such as pairing equations which are needed to generate a signature. As Maji et al. [MPR11], we commit to an element Z of the group $\mathbb{G}$ or the group $\tilde{\mathbb{G}}$, and make $\langle Z \rangle$ denote the commitment value.

<u>IBX.Setup$(1^\lambda, 1^n)$</u>: $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$. $g_0, g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. $\mathcal{I} := \mathbb{Z}_p$ $(vk, mk) \leftarrow$ SPSIG.Gen(gk). `Return` $pk := (gk, vk, g_0, g_1, \cdots, g_n)$ and mk.
<u>IBX.KeyGen$(pk, mk, ID \in \mathcal{I})$</u>: $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $y := \prod_{i=1}^n g_i^{x_i}$. $\theta \leftarrow$ SPSIG.Sig$(vk, mk, y \cdot g_0^{ID})$. `Return` $sk := (x_1, \cdots, x_n, y, \theta)$.
<u>IBX.SKVer$(pk, sk^*, ID \in \mathcal{I})$</u>: sk^* is parsed as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$.   `Return` 1, if $\left[y = \prod_{i=1}^n g_i^{x_i^*}\right] \wedge \left[1 \leftarrow \text{SPSIG.Ver}(vk, \theta^*, y^* \cdot g_0^{ID})\right]$. `Return` 0, otherwise;
<u>IBX.SKVer2(pk, sk^*, sk')</u>: sk^* and sk' are parsed as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$ and $(x_1', \cdots, x_n', y', \theta')$, respectively.   `Return` 1, if $\bigwedge_{i=1}^n \left[x_i^* = x_i'\right]$. `Return` 0, otherwise;

Figure 5.11: Construction of IBX scheme $\Sigma_{\text{IBX},n}$, where $n \in \mathbb{N}$ and $\Sigma_{\text{SPSIG}} = \{$SPSIG.Gen, SPSIG.Sig, SPSIG.Ver$\}$ is a structure-preserving signature scheme whose message space is $\mathbb{G}$.

We use the above specific primitives to instantiate the IBS scheme Σ_{IBS}. Let us consider the case when we generate a signature σ on a message $m \in \{0,1\}^\tau$ for an ID $ID \in \mathbb{Z}_p$ by using a secret-key sk_{ID} for the ID. Note that the secret-key is parsed as $(x_1, \cdots, x_n, y, \theta)$.

Firstly, we generate a ciphertext of the plaintext sk_{ID} under the label $m \| ID$. The ciphertext is written as $\mathbf{C} := (\{\mathbf{c}_{ij}\}_{i \in [1,n], j \in [1,\lambda]}, \{\mathbf{c}_i\}_{i \in [1,2]})$. We parse $\mathbf{c}_{ij}$ as $(\mathbf{y}_{ij}, z_{ij}, c_{ij})$, and parse $\mathbf{y}_{ij}$ as $(y_{ij,1}, y_{ij,2}, y_{ij,3})$. Also, we parse $\mathbf{c}_i$ as $(\mathbf{y}_i, z_i, c_i)$, and parse $\mathbf{y}_i$ as $(y_{i,1}, y_{i,2}, y_{i,3})$.

After that, we generate NIZK proofs as follows.

Firstly, we compute commitments whom we need. Specifically, we compute the following commitments:

- $\langle g_i^{x_{ij}} \rangle$, $\langle \tilde{g}^{x_{ij}} \rangle$, $\langle g_i^{2^j \cdot x_{ij}} \rangle$, $\langle h_0^{r_{ij}} \rangle$, $\langle h_0^{s_{ij}} \rangle$, $\langle h_1^{r_{ij}} \rangle$, $\langle h_2^{s_{ij}} \rangle$, $\langle b_1^{r_{ij}} \rangle$, $\langle b_2^{s_{ij}} \rangle$, $\langle e_1^{r_{ij}} \rangle$, $\langle e_2^{s_{ij}} \rangle$, $\langle e_1^{t_{ij} \cdot r_{ij}} \rangle$, $\langle e_2^{t_{ij} \cdot s_{ij}} \rangle$, $\langle d_1^{r_{ij}} \rangle$ and $\langle d_2^{s_{ij}} \rangle$, where $i \in [1,n]$ and $j \in [1,\lambda]$.

- $\langle y \rangle$, $\langle y \cdot g_0^{ID} \rangle$, $\langle h_0^{r_1} \rangle$, $\langle h_0^{s_1} \rangle$, $\langle h_1^{r_1} \rangle$, $\langle h_2^{s_1} \rangle$, $\langle b_1^{r_1} \rangle$, $\langle b_2^{s_1} \rangle$, $\langle e_1^{r_1} \rangle$, $\langle e_2^{s_1} \rangle$, $\langle e_1^{t_1 \cdot r_1} \rangle$, $\langle e_2^{t_1 \cdot s_1} \rangle$, $\langle d_1^{r_1} \rangle$ and $\langle d_2^{s_1} \rangle$.

- $\langle \theta \rangle$, $\langle h_0^{r_2} \rangle$, $\langle h_0^{s_2} \rangle$, $\langle h_1^{r_2} \rangle$, $\langle h_2^{s_2} \rangle$, $\langle b_1^{r_2} \rangle$, $\langle b_2^{s_2} \rangle$, $\langle e_1^{r_2} \rangle$, $\langle e_2^{s_2} \rangle$, $\langle e_1^{t_2 \cdot r_2} \rangle$, $\langle e_2^{t_2 \cdot s_2} \rangle$, $\langle d_1^{r_2} \rangle$ and $\langle d_2^{s_2} \rangle$.

After that, we prove that the commitments satisfy some equations.

We prove that for $i \in [1,n]$ and $j \in [1,\lambda]$, x_{ij} is a bit and x_{ij} of the commitment $\langle g^{x_{ij}} \rangle$ and that of $\langle \tilde{g}^{x_{ij}} \rangle$ are consistent by generating proofs for the following equations: $[\hat{e}(\langle g_i^{x_{ij}} \rangle, \langle \tilde{g}^{x_{ij}} \rangle) = \hat{e}(g_i, \langle \tilde{g}^{x_{ij}} \rangle) = \hat{e}(\langle g_i^{x_{ij}} \rangle, \tilde{g})]$.

We prove that the ciphertext $\{\mathbf{c}_{ij}\}_{i \in [1,n], j \in [1,\lambda]}$ is a valid ciphertext of the plaintext $\{x_i\}_{i \in [1,n]}$ by generating proofs for the following equations: $[y_{ij,1} = \langle h_0^{r_{ij}} \rangle \cdot \langle h_0^{s_{ij}} \rangle]$, $[y_{ij,2} = \langle h_1^{r_{ij}} \rangle]$, $[y_{ij,3} = \langle h_2^{s_{ij}} \rangle]$, $[\hat{e}(\langle g_i^{2^j \cdot x_{ij}} \rangle, \tilde{g}) = \hat{e}(\langle g_i^{x_{ij}} \rangle, \tilde{g}^{2^j})]$, $[z_{ij} = \langle b_1^{r_{ij}} \rangle \cdot \langle b_2^{s_{ij}} \rangle \cdot \langle g_i^{2^j \cdot x_{ij}} \rangle]$, $[\hat{e}(\langle e_1^{t_{ij} \cdot r_{ij}} \rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_{ij}} \rangle, \tilde{g}^{t_{ij}})]$, $[\hat{e}(\langle e_2^{t_{ij} \cdot s_{ij}} \rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_{ij}} \rangle, \tilde{g}^{t_{ij}})]$, and $[c_{ij} = \langle d_1^{r_{ij}} \rangle \cdot \langle e_1^{t_{ij} \cdot r_{ij}} \rangle \cdot \langle d_2^{s_{ij}} \rangle \cdot \langle e_2^{t_{ij} \cdot s_{ij}} \rangle]$, where $i \in [1,n]$ and $j \in [1,\lambda]$.

We prove that the ciphertext $\mathbf{c}_1$ is a valid ciphertext of the plaintext y by generating proofs for the following equations: $[y_{1,1} = \langle h_0^{r_1} \rangle \cdot \langle h_0^{s_1} \rangle]$, $[y_{1,2} = \langle h_1^{r_1} \rangle]$, $[y_{1,3} = \langle h_2^{s_1} \rangle]$, $[z_1 =$

$\langle b_1^{r_1}\rangle \cdot \langle b_2^{s_1}\rangle \cdot \langle y\rangle]$, $[\hat{e}(\langle e_1^{t_1 \cdot r_1}\rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_1}\rangle, \tilde{g}^{t_1})]$, $[\hat{e}(\langle e_2^{t_1 \cdot s_1}\rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_1}\rangle, \tilde{g}^{t_1})]$, and $[c_1 = \langle d_1^{r_1}\rangle \cdot \langle e_1^{t_1 \cdot r_1}\rangle \cdot \langle d_2^{s_1}\rangle \cdot \langle e_2^{t_1 \cdot s_1}\rangle]$.

We prove that the ciphertext $\mathbf{c}_2$ is a valid ciphertext of the plaintext θ by generating proofs for the following equations: $[y_{2,1} = \langle h_0^{r_2}\rangle \cdot \langle h_0^{s_2}\rangle]$, $[y_{2,2} = \langle h_1^{r_2}\rangle]$, $[y_{2,3} = \langle h_2^{s_2}\rangle]$, $[z_2 = \langle b_1^{r_2}\rangle \cdot \langle b_2^{s_2}\rangle \cdot \langle \theta\rangle]$, $[\hat{e}(\langle e_1^{t_2 \cdot r_2}\rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_2}\rangle, \tilde{g}^{t_2})]$, $[\hat{e}(\langle e_2^{t_2 \cdot s_2}\rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_2}\rangle, \tilde{g}^{t_2})]$, and $[c_1 = \langle d_1^{r_2}\rangle \cdot \langle e_1^{t_2 \cdot r_2}\rangle \cdot \langle d_2^{s_2}\rangle \cdot \langle e_2^{t_2 \cdot s_2}\rangle]$.

We prove that the witness variables $(x_1, \cdots, x_n, y)$ satisfy the relation $y = \prod_{i=1}^{n} g_i^{x_i}$ by generating a proof for the following equation: $[\langle y\rangle = \prod_{i=1}^{n} \prod_{j=1}^{\lambda} \langle g_i^{2^j \cdot x_{ij}}\rangle]$.

We prove that the witness variables (y, θ) satisfy the relation $1 \leftarrow \mathrm{SPSIG.Ver}(vk, \theta, y \cdot g_0^{ID})$ by generating proofs for the following equations: $[1 \leftarrow \mathrm{SPSIG.Ver}(vk, \langle\theta\rangle, \langle y \cdot g_0^{ID}\rangle)]$ and $[\langle y \cdot g_0^{ID}\rangle = \langle y\rangle \cdot g_0^{ID}]$.

Proof of Theorem 5.5.3. Specifically, the HtC-SK game for Σ_{IBX} played by $\mathcal{A}$ and $\mathcal{CH}$ is the following game.

Setup. $\mathcal{CH}$ generates $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$ and $(vk, mk) \leftarrow \mathrm{SPSIG.Gen}(gk)$. $\mathcal{CH}$ chooses $g_0, g_1, \cdots, g_n \xleftarrow{U} \mathbb{G}$. $\mathcal{CH}$ sets $pk := (gk, vk, g_0, g_1, \cdots, g_n)$, then sends it to $\mathcal{A}$.

Query. When $\mathcal{A}$ queries to the oracle **Reveal**, $\mathcal{CH}$ behaves as follows.

> **Reveal**$(ID \in \mathbb{Z}_p)$**:** $\mathcal{CH}$ chooses $x_1, \cdots, x_n \xleftarrow{U} \mathbb{Z}_p$, then calculates $y := \prod_{i=1}^{n} g_i^{x_i}$. After that, $\mathcal{CH}$ runs $\theta \leftarrow \mathrm{SPSIG.Sig}(vk, mk, y \cdot g_0^{ID})$, then returns the secret-key $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Compute$(ID^* \in \mathbb{Z}_p, sk^*)$**.** sk^* is parsed as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$.

Let W denote the event where $\mathcal{A}$ wins the game. Thus,

$$W := \left[\left[y^* = \prod_{i=1}^{n} g_i^{x_i^*}\right] \wedge \left[1 \leftarrow \mathrm{SPSIG.Ver}(vk, \theta^*, y^* \cdot g_0^{ID^*})\right] \bigwedge_{sk^\dagger \in \mathcal{L}_{ID^*}} \left[\bigvee_{i \in [1,n]} \left[x_i^\dagger \neq x_i^*\right]\right]\right],$$

where $sk^\dagger$ is parsed as $(x_1^\dagger, \cdots, x_n^\dagger, y^\dagger, \theta^\dagger)$. Let us introduce two events A_1, A_2. They are defined as follows.

$$A_1 := \left[\bigvee_{ID \text{ s.t. } [\mathcal{L}_{ID} \neq \emptyset] \wedge [ID \neq ID^*]} \bigvee_{sk \in \mathcal{L}_{ID}} \left[y \cdot g_0^{ID} = y^* \cdot g_0^{ID^*}\right]\right] \text{ and } A_2 := \left[\bigvee_{sk \in \mathcal{L}_{ID^*}} [y = y^*]\right],$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. Obviously, the following equation holds: $\Pr[W] = \Pr[W \wedge A_1] + \Pr[W \wedge \bar{A}_1 \wedge A_2] + \Pr[W \wedge \bar{A}_1 \wedge \bar{A}_2]$. Hence, by the following three lemmas, $\Pr[W]$ is negligible. □

Lemma 5.5.1. $\Pr[W \wedge A_1]$ *is negligible under the DL assumption.*

Lemma 5.5.2. $\Pr\left[W \wedge \bar{A}_1 \wedge A_2\right]$ *is negligible under the DL assumption.*

Lemma 5.5.3. $\Pr\left[W \wedge \bar{A}_1 \wedge \bar{A}_2\right]$ *is negligible under the EUF-CMA security of* Σ_{SPSIG}.

Proof of each lemma is given below.

Proof of Lemma 5.5.1. We prove that if there exists an PPT adversary $\mathcal{A}$ who makes $\Pr[W \wedge A_1]$ non-negligible, we can construct a PPT simulator $\mathcal{S}$ who breaks the DL assumption. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{U}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. As an instance of the DL problem, $\mathcal{S}$ is given (h, h^α), where $\alpha \overset{U}{\leftarrow} \mathbb{Z}_p$. She sets $g_0 := h^\alpha$. She runs $(vk, mk) \leftarrow \text{SPSIG.Gen}(gk)$. For every $i \in [1, n]$, She chooses $\beta_i \overset{U}{\leftarrow} \mathbb{Z}_p$ and sets $g_i := h^{\beta_i}$. She sends $pk := (gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $ID \in \mathbb{Z}_p$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{U}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^n g_i^{x_i}$. She generates $\theta \leftarrow \text{SPSIG.Sig}(vk, mk, y \cdot g_0^{ID})$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Compute$(ID^* \in \mathbb{Z}_p, sk^*)$**.** $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. $\mathcal{A}$ is assumed to make $\Pr[W \wedge A_1]$ non-negligible, so the following event happens with a non-negligible probability:

$$\left[y^* = \prod_{i=1}^n g_i^{x_i^*}\right] \wedge \left[1 \leftarrow \text{SPSIG.Ver}(vk, \theta^*, y^* \cdot g_0^{ID^*})\right] \bigwedge_{sk^\dagger \in \mathcal{L}_{ID^*}} \left[\bigvee_{i \in [1,n]} \left[x_i^\dagger \neq x_i^*\right]\right]$$
$$\wedge \left[\bigvee_{ID \ s.t.\ [\mathcal{L}_{ID} \neq \emptyset] \wedge [ID \neq ID^*]} \left[\bigvee_{sk \in \mathcal{L}_{ID}} \left[y \cdot g_0^{ID} = y^* \cdot g_0^{ID^*}\right]\right]\right],$$

where $sk^\dagger$ and sk are parsed as $(x_1^\dagger, \cdots, x_n^\dagger, y^\dagger, \theta^\dagger)$ and $(x_1, \cdots, x_n, y, \theta)$, respectively. $\mathcal{S}$ computes $\alpha' := (\sum_{i=1}^n \beta_i \cdot (x_i - x_i^*))/(ID^* - ID)$ and outputs it.

Let us explain why $\mathcal{S}$ breaks the DL assumption. Applying the fact that $g_0 = h^\alpha$ and $g_i = h^{\beta_i}$ to the equation $y \cdot g_0^{ID} = y^* \cdot g_0^{ID^*}$, we obtain $\alpha = (\sum_{i=1}^n \beta_i \cdot (x_i - x_i^*))/(ID^* - ID) = \alpha'$. Since $ID \neq ID^*$, it holds that $y \neq y^*$. Hence, it also holds that $\bigvee_{i \in [1,n]} \left[x_i \neq x_i^*\right]$. Hence, she can compute $\alpha' (= \alpha)$ correctly. □

Proof of Lemma 5.5.2. We prove the lemma by using n-representation assumption.

Definition 31. *n-representation assumption holds if for every PPT $\mathcal{A}$,* $\Pr[\mathcal{A}(p, \mathbb{G}, g_1, \cdots, g_n) \rightarrow ((x_1, \cdots, x_n), (x_1', \cdots, x_n'))$ *s.t.* $[(x_1, \cdots, x_n) \neq (x_1', \cdots, x_n')] \wedge [\prod_{i \in [1,n]} g_i^{x_i} = \prod_{i \in [1,n]} g_i^{x_i'}]]$ *is negligible, where $(p, \mathbb{G}, g) \leftarrow \mathcal{G}(1^\lambda)$ and $g_1, \cdots, g_n \overset{U}{\leftarrow} \mathbb{G}$.*

Validity of the assumption is guaranteed by the following theorem [Bra93, BGG94].

Theorem 5.5.5. *For any $n \in \mathbb{N}$, n-representation assumption holds under the DL assumption.*

We prove that if there exists a PPT adversary $\mathcal{A}$ who makes $\Pr[W \wedge \bar{A}_1 \wedge A_2]$ non-negligible, then there exists a PPT simulator $\mathcal{S}$ who breaks the n-representation assumption. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{\mathrm{U}}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. As an instance of the problem of the n-representation assumption, $\mathcal{S}$ is given $(g_1, \cdots, g_n)$, where $g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. She sets $g_0 \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. She runs $(vk, mk) \leftarrow \mathrm{SPSIG.Gen}(gk)$. She sends $pk := (gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $ID \in \mathbb{Z}_p$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^n g_i^{x_i}$. She generates $\theta \leftarrow \mathrm{SPSIG.Sig}(vk, mk, y \cdot g_0^{ID})$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Compute$(ID^* \in \mathbb{Z}_p, sk^*)$**.** $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. Assuming that the event $W \wedge \bar{A}_1 \wedge A_2$ occurs implies that $\exists sk^\dagger \in \mathcal{L}_{ID^*}$ *s.t.* $[y^\dagger = \prod_{i=1}^n g_i^{x_i^\dagger} = y^* = \prod_{i=1}^n g_i^{x_i^*}] \wedge [\bigvee_{i\in[1,n]}[x_i^\dagger \neq x_i^*]]$, where $sk^\dagger$ is parsed as $(x_1^\dagger, \cdots, x_n^\dagger, y^\dagger, \theta^\dagger)$. $\mathcal{S}$ outputs $(x_1^\dagger, \cdots, x_n^\dagger)$, $(x_1^*, \cdots, x_n^*)$.

Obviously, $\mathcal{S}$ breaks the n-representation assumption. □

Proof of Lemma 5.5.3. We prove that if there exists a PPT adversary $\mathcal{A}$, then there exists a PPT simulator $\mathcal{S}$ who breaks the EUF-CMA security of Σ_{SPSIG}. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{\mathrm{U}}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. $\mathcal{S}$ chooses $g_0, g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. She is given the verification-key vk of Σ_{SPSIG}. She sends $pk := (gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $ID \in \mathbb{Z}_p$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^n g_i^{x_i}$. She issues $y \cdot g_0^{ID}$ to the sigining oracle of the EUF-CMA game of Σ_{SPSIG}, then receives $\theta \leftarrow \mathrm{SPSIG.Sig}(vk, mk, y \cdot g_0^{ID})$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{ID} := \mathcal{L}_{ID} \cup \{sk\}$.

Compute$(ID^* \in \mathbb{Z}_p, sk^*)$**.** $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. $\mathcal{S}$ outputs θ^* as a forged signature on a message $y^* \cdot g_0^{ID^*}$.

Let us explain why $\mathcal{S}$ breaks the EUF-CMA of Σ_{SPSIG}. Assuming that the event $W \wedge \bar{A}_1 \wedge \bar{A}_2$ occurs implies that $[y^* = \prod_{i=1}^n g_i^{x_i^*}] \wedge [1 \leftarrow \mathrm{SPSIG.Ver}(vk, \theta^*, y^* \cdot g_0^{ID^*})] \bigwedge_{ID \text{ s.t. } [\mathcal{L}_{ID} \neq \emptyset]} [\bigwedge_{sk \in \mathcal{L}_{ID}} [y \cdot g_0^{ID} \neq y^* \cdot g_0^{ID^*}]]$, where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. Thus, $\mathcal{S}$ wins the EUF-CMA game by outputting $(\theta^*, y^* \cdot g_0^{ID^*})$. □

5.5.2 An Instantiation of ABS for a General Circuit

As ABX scheme, we use the scheme described in Fig. 5.12. Its HtC-SK property is guaranteed by the following theorem whose proof is given in the latter half of this subsection.

Theorem 5.5.6. *For any (polynomially bounded) integers $L, n \in \mathbb{N}$, $\Sigma_{\mathrm{ABX},L,n}$ is HtC-SK under the DL assumption and the EUF-CMA security of* Σ_{SPSIG}.

The ABX scheme needs a structure-preserving signature scheme as a building block. We use the same structure-preserving signature scheme as the one used to instantiate our IBS scheme, i.e., the scheme by Kiltz et al. [KPW15]. We also use the same LPKE and NIZK schemes in the last subsection, i.e., the LPKE scheme in Fig. 5.10 and Groth-Sahai NIZK system [GS08], respectively.

$\underline{\mathsf{ABX.Setup}(1^\lambda, 1^L, 1^n)}$: $\mathcal{U} := \{0,1\}^L$. $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$. $g_0, g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. $(vk, mk) \leftarrow \mathsf{SPSIG.Gen}(gk, 1^L)$. $\mathtt{Return}$ $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$ and mk.
$\underline{\mathsf{ABX.KeyGen}(pk, mk, \mathbb{W} \in \mathcal{U})}$: For $i \in [1, L]$, the i-th bit of $\mathbb{W}$ is denoted by w_i. $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. $y := \prod_{i=1}^n g_i^{x_i}$. $\theta \leftarrow \mathsf{SPSIG.Sig}(vk, mk, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))$. $\mathtt{Return}$ $sk := (x_1, \cdots, x_n, y, \theta)$.
$\underline{\mathsf{ABX.SKVer}(pk, sk, \phi)}$: sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. ϕ is represented as a circuit $\{L, N, I_1, I_2\}$. $\mathtt{Return}$ 1, if $[y = \prod_{i=1}^n g_i^{x_i}] \wedge [\exists \mathbb{W} \in \{0,1\}^L$ s.t. $[\phi(\mathbb{W}) = 1]$ $\wedge\ [1 \leftarrow \mathsf{SPSIG.Ver}(vk, \theta, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))]]$. $\mathtt{Return}$ 0, otherwise.
$\underline{\mathsf{ABX.SKVer2}(pk, sk, sk^*)}$: sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. sk^* is parsed as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$ $\mathtt{Return}$ 1, if $\bigwedge_{i=1}^n \left[x_i = x_i^*\right]$. $\mathtt{Return}$ 0, otherwise.

Figure 5.12: Construction of ABX scheme $\Sigma_{\mathrm{ABX},L,n}$, where $L, n \in \mathbb{N}$ and $\Sigma_{\mathrm{SPSIG}} = \{\mathsf{SPSIG.Gen}, \mathsf{SPSIG.Sig}, \mathsf{SPSIG.Ver}\}$ is a structure-preserving signature scheme whose message space is $\mathbb{G}^L$.

Let us consider the case when we generate a signature σ on a message $m \in \{0,1\}^\tau$ for a predicate as a circuit $\phi = \{L, N, I_1, I_2\}$ by using a secret-key sk for an attribute $\mathbb{W} \in \{0,1\}^L$ such that $\phi(\mathbb{W}) = 1$. Note that the secret-key is parsed as $(x_1, \cdots, x_n, y, \theta)$.

Firstly, we generate a ciphertext of the plaintext sk under the label $m \| \phi$. The ciphertext is written as $\mathbf{C} := (\{\mathbf{c}_{ij}\}_{i \in [1,n], j \in [1,\lambda]}, \{\mathbf{c}_i\}_{i \in [1,2]})$. We parse $\mathbf{c}_{ij}$ as $(\mathbf{y}_{ij}, z_{ij}, c_{ij})$, and parse $\mathbf{y}_{ij}$ as $(y_{ij,1}, y_{ij,2}, y_{ij,3})$. Also, we parse $\mathbf{c}_i$ as $(\mathbf{y}_i, z_i, c_i)$, and parse $\mathbf{y}_i$ as $(y_{i,1}, y_{i,2}, y_{i,3})$.

After that, we generate NIZK proofs as follows.

Firstly, for every $i \in [L+1, L+N-1]$, we compute $w_i := 1 - w_{I_1(i)} \cdot w_{I_2(i)}$.

Secondly, we compute commitments whom we need. Specifically, we compute the following commitments:

- $\langle g_i^{x_{ij}} \rangle$, $\langle \tilde{g}^{x_{ij}} \rangle$, $\langle g_i^{2^j \cdot x_{ij}} \rangle$, $\langle h_0^{r_{ij}} \rangle$, $\langle h_0^{s_{ij}} \rangle$, $\langle h_1^{r_{ij}} \rangle$, $\langle h_2^{s_{ij}} \rangle$, $\langle b_1^{r_{ij}} \rangle$, $\langle b_2^{s_{ij}} \rangle$, $\langle e_1^{r_{ij}} \rangle$, $\langle e_2^{s_{ij}} \rangle$, $\langle e_1^{t_{ij} \cdot r_{ij}} \rangle$, $\langle e_2^{t_{ij} \cdot s_{ij}} \rangle$, $\langle d_1^{r_{ij}} \rangle$ and $\langle d_2^{s_{ij}} \rangle$, where $i \in [1, n]$ and $j \in [1, \lambda]$.
- $\langle y \rangle$, $\langle g_0^{w_1} \rangle$, $\langle \tilde{g}^{w_1} \rangle$, $\cdots$, $\langle g_0^{w_{L+N-1}} \rangle$, $\langle \tilde{g}^{w_{L+N-1}} \rangle$, $\langle y \cdot g_0^{w_1} \rangle$, $\cdots$, $\langle y \cdot g_0^{w_L} \rangle$, $\langle h_0^{r_1} \rangle$, $\langle h_0^{s_1} \rangle$, $\langle h_1^{r_1} \rangle$, $\langle h_2^{s_1} \rangle$, $\langle b_1^{r_1} \rangle$, $\langle b_2^{s_1} \rangle$, $\langle e_1^{r_1} \rangle$, $\langle e_2^{s_1} \rangle$, $\langle e_1^{t_1 \cdot r_1} \rangle$, $\langle e_2^{t_1 \cdot s_1} \rangle$, $\langle d_1^{r_1} \rangle$ and $\langle d_2^{s_1} \rangle$.
- $\langle \theta \rangle$, $\langle h_0^{r_2} \rangle$, $\langle h_0^{s_2} \rangle$, $\langle h_1^{r_2} \rangle$, $\langle h_2^{s_2} \rangle$, $\langle b_1^{r_2} \rangle$, $\langle b_2^{s_2} \rangle$, $\langle e_1^{r_2} \rangle$, $\langle e_2^{s_2} \rangle$, $\langle e_1^{t_2 \cdot r_2} \rangle$, $\langle e_2^{t_2 \cdot s_2} \rangle$, $\langle d_1^{r_2} \rangle$ and $\langle d_2^{s_2} \rangle$.

After that, we prove that the commitments satisfy some equations.

We prove that for $i \in [1, n]$ and $j \in [1, \lambda]$, x_{ij} is a bit and x_{ij} of the commitment $\langle g^{x_{ij}} \rangle$ and that of $\langle \tilde{g}^{x_{ij}} \rangle$ are consistent by generating proofs for the following equations: $[\hat{e}(\langle g_i^{x_{ij}} \rangle, \langle \tilde{g}^{x_{ij}} \rangle) = \hat{e}(g_i, \langle \tilde{g}^{x_{ij}} \rangle) = \hat{e}(\langle g_i^{x_{ij}} \rangle, \tilde{g})]$.

We prove that the ciphertext $\{\mathbf{c}_{ij}\}_{i\in[1,n],j\in[1,\lambda]}$ is a valid ciphertext of the plaintext $\{x_i\}_{i\in[1,n]}$ by generating proofs for the following equations: $[y_{ij,1} = \langle h_0^{r_{ij}}\rangle \cdot \langle h_0^{s_{ij}}\rangle]$, $[y_{ij,2} = \langle h_1^{r_{ij}}\rangle]$, $[y_{ij,3} = \langle h_2^{s_{ij}}\rangle]$, $[\hat{e}(\langle g_i^{2^j\cdot x_{ij}}\rangle, \tilde{g}) = \hat{e}(\langle g_i^{x_{ij}}\rangle, \tilde{g}^{2^j})]$, $[z_{ij} = \langle b_1^{r_{ij}}\rangle \cdot \langle b_2^{s_{ij}}\rangle \cdot \langle g_i^{2^j\cdot x_{ij}}\rangle]$, $[\hat{e}(\langle e_1^{t_{ij}\cdot r_{ij}}\rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_{ij}}\rangle, \tilde{g}^{t_{ij}})]$, $[\hat{e}(\langle e_2^{t_{ij}\cdot s_{ij}}\rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_{ij}}\rangle, \tilde{g}^{t_{ij}})]$, and $[c_{ij} = \langle d_1^{r_{ij}}\rangle \cdot \langle e_1^{t_{ij}\cdot r_{ij}}\rangle \cdot \langle d_2^{s_{ij}}\rangle \cdot \langle e_2^{t_{ij}\cdot s_{ij}}\rangle]$, where $i \in [1,n]$ and $j \in [1,\lambda]$.

We prove that the ciphertext $\mathbf{c}_1$ is a valid ciphertext of the plaintext y by generating proofs for the following equations: $[y_{1,1} = \langle h_0^{r_1}\rangle \cdot \langle h_0^{s_1}\rangle]$, $[y_{1,2} = \langle h_1^{r_1}\rangle]$, $[y_{1,3} = \langle h_2^{s_1}\rangle]$, $[z_1 = \langle b_1^{r_1}\rangle \cdot \langle b_2^{s_1}\rangle \cdot \langle y\rangle]$, $[\hat{e}(\langle e_1^{t_1\cdot r_1}\rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_1}\rangle, \tilde{g}^{t_1})]$, $[\hat{e}(\langle e_2^{t_1\cdot s_1}\rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_1}\rangle, \tilde{g}^{t_1})]$, and $[c_1 = \langle d_1^{r_1}\rangle \cdot \langle e_1^{t_1\cdot r_1}\rangle \cdot \langle d_2^{s_1}\rangle \cdot \langle e_2^{t_1\cdot s_1}\rangle]$.

We prove that the ciphertext $\mathbf{c}_2$ is a valid ciphertext of the plaintext θ by generating proofs for the following equations: $[y_{2,1} = \langle h_0^{r_2}\rangle \cdot \langle h_0^{s_2}\rangle]$, $[y_{2,2} = \langle h_1^{r_2}\rangle]$, $[y_{2,3} = \langle h_2^{s_2}\rangle]$, $[z_2 = \langle b_1^{r_2}\rangle \cdot \langle b_2^{s_2}\rangle \cdot \langle \theta\rangle]$, $[\hat{e}(\langle e_1^{t_2\cdot r_2}\rangle, \tilde{g}) = \hat{e}(\langle e_1^{r_2}\rangle, \tilde{g}^{t_2})]$, $[\hat{e}(\langle e_2^{t_2\cdot s_2}\rangle, \tilde{g}) = \hat{e}(\langle e_2^{s_2}\rangle, \tilde{g}^{t_2})]$, and
$[c_1 = \langle d_1^{r_2}\rangle \cdot \langle e_1^{t_2\cdot r_2}\rangle \cdot \langle d_2^{s_2}\rangle \cdot \langle e_2^{t_2\cdot s_2}\rangle]$.

We prove that the witness variables $(x_1, \cdots, x_n, y)$ satisfy the relation $y = \prod_{i=1}^n g_i^{x_i}$ by generating a proof for the following equation: $\left[\langle y\rangle = \prod_{i=1}^n \prod_{j=1}^\lambda \langle g_i^{2^j\cdot x_{ij}}\rangle\right]$.

We prove that the witness variables (y, θ) satisfy the relation $\exists \mathbb{W} \in \{0,1\}^L$ *s.t.* $[\phi(\mathbb{W}) = 1] \wedge [1 \leftarrow \mathsf{SPSIG.Ver}(vk, \theta, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))]$ by generating proofs for the following equations:

- $[\hat{e}(g_0, \langle \tilde{g}^{w_i}\rangle) = \hat{e}(\langle g_0^{w_i}\rangle, \tilde{g})]$, where $i \in [1, L]$.
- $[\hat{e}(g_0, \langle \tilde{g}^{w_i}\rangle) = \hat{e}(\langle g_0^{w_i}\rangle, \tilde{g}) = \hat{e}(\langle g_0^{w_{I_1(i)}}\rangle, \langle \tilde{g}^{w_{I_2(i)}}\rangle)^{-1} \cdot \hat{e}(g_0, \tilde{g})]$, where $i \in [L+1, L+N-1]$.
- $\hat{e}(\langle g_0^{w_{I_1(L+N)}}\rangle, \langle \tilde{g}^{w_{I_2(L+N)}}\rangle) = 1_{\mathbb{G}_T}]$, where $1_{\mathbb{G}_T}$ denotes the identity element of $\mathbb{G}_T$.
- $[\langle y \cdot g_0^{w_1}\rangle = \langle y\rangle \cdot \langle g_0^{w_1}\rangle]$, $\cdots$, $[\langle y \cdot g_0^{w_L}\rangle = \langle y\rangle \cdot \langle g_0^{w_L}\rangle]$ and $[1 \leftarrow \mathsf{SPSIG.Ver}(vk, \langle\theta\rangle, (\langle y \cdot g_0^{w_1}\rangle, \cdots, \langle y \cdot g_0^{w_L}\rangle))]$.

Proof of Theorem 5.5.6. Specifically, the HtC-SK game for Σ_{ABX} played by $\mathcal{A}$ and $\mathcal{CH}$ is the following game.

Setup. $\mathcal{CH}$ sets $\mathcal{U} := \{0,1\}^L$. $\mathcal{CH}$ generates $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, g, \tilde{g}) \leftarrow \mathcal{G}_{pg}(1^\lambda)$ and $(vk, mk) \leftarrow \mathsf{SPSIG.Gen}(gk, 1^L)$. $\mathcal{CH}$ chooses $g_0, g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. $\mathcal{CH}$ sets $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$, then sends it to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $\mathbb{W} \in \mathcal{U}$ to the oracle **Reveal**, $\mathcal{CH}$ behaves as follows. $\mathcal{CH}$ chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$, then calculates $y := \prod_{i=1}^n g_i^{x_i}$. After that, $\mathcal{CH}$ runs $\theta \leftarrow \mathsf{SPSIG.Sig}(vk, mk, (y\cdot g_0^{w_1}, \cdots, y\cdot g_0^{w_L}))$, then returns the secret-key $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. After that, $\mathcal{CH}$ sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Compute(ϕ^*, sk^*). sk^* is parsed as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$.

Let W denote the event where $\mathcal{A}$ wins the game. Thus,

$$\begin{aligned}W \ :=\ & \Bigg[\Bigg[y^* = \prod_{i=1}^{n} g_i^{x_i^*}\Bigg] \bigwedge_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=1]} \Bigg[\bigwedge_{sk\in\mathcal{L}_{\mathbb{W}}} \Bigg[\bigvee_{i\in[1,n]} [x_i \neq x_i^*]\Bigg]\Bigg] \\ & \wedge \Big[\exists \mathbb{W}^* \in \{0,1\}^L \ s.t.\ [\phi^*(\mathbb{W}^*) = 1] \wedge \Big[1 \leftarrow \mathsf{SPSIG.Ver}(vk, \theta^*, (y\cdot g_0^{w_1^*}, \cdots, y\cdot g_0^{w_L^*}))\Big]\Big]\Bigg],\end{aligned}$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. Related to the event W, we define a function $F_W(X)$ which takes an event X as input and outputs an event as follows.

$$\begin{aligned}F_W(X) \ :=\ & \Bigg[\Bigg[y^* = \prod_{i=1}^{n} g_i^{x_i^*}\Bigg] \bigwedge_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=1]} \Bigg[\bigwedge_{sk\in\mathcal{L}_{\mathbb{W}}} \Bigg[\bigvee_{i\in[1,n]} [x_i \neq x_i^*]\Bigg]\Bigg] \wedge \Big[\exists \mathbb{W}^* \in \{0,1\}^L \\ & s.t.\ [\phi^*(\mathbb{W}^*) = 1] \wedge \Big[1 \leftarrow \mathsf{SPSIG.Ver}(vk, \theta^*, (y\cdot g_0^{w_1^*}, \cdots, y\cdot g_0^{w_L^*}))\Big] \wedge X\Big]\Bigg],\end{aligned}$$

For instance, the input X can be the following event A_1.

$$A_1 := \Bigg[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=0]} \Bigg[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \Bigg[\bigwedge_{i\in[1,L]} \Big[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*}\Big]\Bigg]\Bigg]\Bigg],$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. By the definitions of the events W, A_1 and the function $F_W(\cdot)$, we obtain

$$\Pr[W] \leq \Pr[F_W(A_1)] + \Pr[F_W(\bar{A_1})]. \tag{5.22}$$

Related to the event A_1, we define another event A_2 as

$$A_2 := \Bigg[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=1]} \Bigg[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \Bigg[\bigwedge_{i\in[1,L]} \Big[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*}\Big]\Bigg]\Bigg]\Bigg],$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. By the definitions of $A_1, A_2, F_W(\cdot)$, we obtain

$$\Pr[F_W(\bar{A_1})] \leq \Pr[F_W(\bar{A_1} \wedge A_2)] + \Pr[F_W(\bar{A_1} \wedge \bar{A_2})]. \tag{5.23}$$

By (5.22) and (5.23), we obtain

$$\Pr[W] \leq \Pr[F_W(A_1)] + \Pr[F_W(\bar{A_1} \wedge A_2)] + \Pr[F_W(\bar{A_1} \wedge \bar{A_2})]. \tag{5.24}$$

We define the other 5 events $\{B_1, B_2, C_1, C_2, C_3\}$ as follows, where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$.

$$
\begin{aligned}
B_1 &:= \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=0]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y = y^*] \bigwedge_{i\in[1,L]} \left[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*} \right] \right] \right] \right] \\
B_2 &:= \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=0]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y \neq y^*] \bigwedge_{i\in[1,L]} \left[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*} \right] \right] \right] \right] \\
C_1 &:= \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=1]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y = y^*] \bigwedge_{i\in[1,L]} \left[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*} \right] \right] \right] \right] \\
C_2 &:= \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=1]\wedge[\mathbb{W}\neq\mathbb{W}^*]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y \neq y^*] \bigwedge_{i\in[1,L]} \left[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*} \right] \right] \right] \right] \\
C_3 &:= \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\mathbb{W}=\mathbb{W}^*]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y \neq y^*] \bigwedge_{i\in[1,L]} \left[y\cdot g_0^{w_i} = y^*\cdot g_0^{w_i^*} \right] \right] \right] \right]
\end{aligned}
$$

By the definitions of the events, we obtain

$$
\begin{aligned}
\Pr[F_W(A_1)] &\leq \Pr[F_W(B_1)] + \Pr[F_W(B_2)] && (5.25) \\
\Pr[F_W(\bar{A}_1 \wedge A_2)] &\leq \Pr[F_W(\bar{A}_1 \wedge C_1)] + \Pr[F_W(\bar{A}_1 \wedge C_2)] + \Pr[F_W(\bar{A}_1 \wedge C_3)] && (5.26)
\end{aligned}
$$

By (5.24), (5.25), (5.26), we obtain

$$
\begin{aligned}
\Pr[W] \leq\ & \Pr[F_W(B_1)] + \Pr[F_W(B_2)] + \Pr[F_W(\bar{A}_1 \wedge C_1)] + \Pr[F_W(\bar{A}_1 \wedge C_2)] \\
& + \Pr[F_W(\bar{A}_1 \wedge C_3)] + \Pr[F_W(\bar{A}_1 \wedge \bar{A}_2)] \qquad (5.27)
\end{aligned}
$$

If the following 6 lemmas are true, by (5.27), $\Pr[W]$ is negligible. □

Lemma 5.5.4. $\Pr[F_W(\bar{A}_1 \wedge \bar{A}_2)]$ *is negligible under the EUF-CMA of* Σ_{SPSIG}.

Lemma 5.5.5. *The event $F_W(B_1)$ never occurs, i.e.,* $\Pr[F_W(B_1)] = 0$.

Lemma 5.5.6. $\Pr[F_W(B_2)]$ *is negligible under the DL assumption.*

Lemma 5.5.7. $\Pr\left[F_W(\bar{A}_1 \wedge C_1)\right]$ *is negligible under the DL assumption.*

Lemma 5.5.8. $\Pr\left[F_W(\bar{A}_1 \wedge C_2)\right]$ *is negligible under the DL assumption.*

Lemma 5.5.9. *The event $F_W(\bar{A}_1 \wedge C_3)$ never occurs, i.e.,* $\Pr\left[F_W(\bar{A}_1 \wedge C_3)\right] = 0$.

Proof of each lemma is given below.

Proof of Lemma 5.5.4. We prove that if there exists a PPT adversary $\mathcal{A}$, then there exists a PPT simulator $\mathcal{S}$ who breaks the EUF-CMA security of Σ_{SPSIG}. $\mathcal{S}$ behaves as follows.

Setup. Let $\mathcal{U} := \{0,1\}^L$. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{\mathrm{U}}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. $\mathcal{S}$ chooses $g_0, g_1, \cdots, g_n \overset{\mathrm{U}}{\leftarrow} \mathbb{G}$. She is given the verification-key vk of Σ_{SPSIG}. She sends $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $\mathbb{W} \in \{0,1\}^L$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^n g_i^{x_i}$. She issues $(y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L})$ to the sigining oracle of the EUF-CMA game of Σ_{SPSIG}, then receives $\theta \leftarrow \mathrm{SPSIG.Sig}(vk, mk, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Compute(ϕ^*, sk^*). $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. If we assume that the event $F_W(\bar{A}_1 \wedge \bar{A}_2)$ occurs, then $\exists \mathbb{W}^* \in \{0,1\}^L$ *s.t.* $[\phi^*(\mathbb{W}^*) = 1] \wedge [1 \leftarrow \mathrm{SPSIG.Ver}(vk, \theta^*, (y \cdot g_0^{w_1^*}, \cdots, y \cdot g_0^{w_L^*}))] \bigwedge_{\mathbb{W} \in \{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}} \neq \emptyset]} [\bigwedge_{sk \in \mathcal{L}_{\mathbb{W}}} [\bigvee_{i \in [1,L]} [y \cdot g_0^{x_i} \neq y^* \cdot g_0^{x_i^*}]]]$, where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. $\mathcal{S}$ outputs θ^* as a forged signature on a message $(y^* \cdot g_0^{w_1^*}, \cdots, y^* \cdot g_0^{w_L^*})$.

$\square$

Proof of Lemma 5.5.5. For an attribute $\mathbb{W} \in \{0,1\}^L$, if $\phi^*(\mathbb{W}) = 0$, then there exists $i \in [1, L]$ such that $w_i \neq w_i^*$. Additionally, if $y = y^*$, then there exists $i \in [1, L]$ such that $y \cdot w_i \neq y^* \cdot w_i^*$. Thus, the event $F_W(B_1)$ never occurs. $\square$

Proof of Lemma 5.5.6. We prove that if there exists an PPT adversary $\mathcal{A}$ who makes $\Pr[F_W(B_2)]$ non-negligible, we can construct a PPT simulator $\mathcal{S}$ who breaks the DL assumption. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{\mathrm{U}}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. As an instance of the DL problem, $\mathcal{S}$ is given (h, h^α), where $\alpha \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. She sets $g_0 := h^\alpha$. She runs $(vk, mk) \leftarrow \mathrm{SPSIG.Gen}(gk, 1^L)$. For every $i \in [1, n]$, She chooses $\beta_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and sets $g_i := h^{\beta_i}$. She sends $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $\mathbb{W} \in \{0,1\}^L$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^n g_i^{x_i}$. She generates $\theta \leftarrow \mathrm{SPSIG.Sig}(vk, mk, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Compute(ϕ^*, sk^*). $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. Assuming that $F_W(B_2)$ occurs implies that the following event occurs.

$$\left[y^* = \prod_{i=1}^{n} g_i^{x_i^*}\right] \wedge \left[\exists \mathbb{W}^* \ s.t.\ [\phi^*(\mathbb{W}^*) = 1] \wedge \left[1 \leftarrow \text{SPSIG.}\mathsf{Ver}(vk, \theta^*, (y^* \cdot g_0^{w_1^*}, \cdots, y^* \cdot g_0^{w_L^*}))\right]\right.$$
$$\left.\wedge \left[\bigvee_{\mathbb{W}\in\{0,1\}^L \ s.t.\ [\mathcal{L}_{\mathbb{W}}\neq\emptyset]\wedge[\phi^*(\mathbb{W})=0]} \left[\bigvee_{sk\in\mathcal{L}_{\mathbb{W}}} \left[[y \neq y^*] \bigwedge_{i\in[1,L]} \left[y \cdot g_0^{w_i} = y^* \cdot g_0^{w_i^*}\right]\right]\right]\right]\right],$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. Since $\phi^*(\mathbb{W}^*) = 1$ and $\phi^*(\mathbb{W}) = 0$, there exists $i \in [1, L]$ such that $w_i \neq w_i^*$. Given the integer i, $\mathcal{S}$ computes $\alpha' := (\sum_{j=1}^{n} \beta_j \cdot (x_j - x_j^*))/(w_i^* - w_i)$ and outputs it.

Let us explain why $\mathcal{S}$ breaks the DL assumption. Applying the fact that $g_0 = h^{\alpha}$ and $g_i = h^{\beta_i}$ to the equation $y \cdot g_0^{w_i} = y^* \cdot g_0^{w_i^*}$, we obtain $\alpha = (\sum_{j=1}^{n} \beta_j \cdot (x_j - x_j^*))/(w_i^* - w_i) = \alpha'$. Since $y \neq y^*$, $\bigvee_{j\in[1,n]} \left[x_j \neq x_j^*\right]$. Hence, she can compute $\alpha'(= \alpha)$ correctly. □

<u>Proof of Lemma 5.5.7.</u> We prove the lemma by using the n-representation assumption whose definition was given in the proof of Lemma 5.5.2.

We prove that if there exists a PPT adversary $\mathcal{A}$ who makes $\Pr[F_W(\bar{A}_1 \wedge C_1)]$ non-negligible, then there exists a PPT simulator $\mathcal{S}$ who breaks the n-representation assumption. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{U}{\leftarrow} \mathcal{G}_{pg}(1^{\lambda})$. As an instance of the problem of the n-representation assumption, $\mathcal{S}$ is given $(g_1, \cdots, g_n)$, where $g_1, \cdots, g_n \overset{U}{\leftarrow} \mathbb{G}$. She sets $g_0 \overset{U}{\leftarrow} \mathbb{G}$. She runs $(vk, mk) \leftarrow \text{SPSIG.}\mathsf{Gen}(gk, 1^L)$. She sends $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $\mathbb{W} \in \{0, 1\}^L$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{U}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^{n} g_i^{x_i}$. She generates $\theta \leftarrow \text{SPSIG.}\mathsf{Sig}(vk, mk, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Compute(ϕ^*, sk^*). $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. Assuming that the event $F_W(\bar{A}_1 \wedge C_1)$ occurs implies that $\exists \mathbb{W} \in \{0, 1\}^L$ such that $[\mathcal{L}_{\mathbb{W}} \neq \emptyset] \wedge [\phi^*(\mathbb{W}) = 1]$, $\exists sk \in \mathcal{L}_{\mathbb{W}}$ such that $[y = \prod_{i=1}^{n} g_i^{x_i} = y^* = \prod_{i=1}^{n} g_i^{x_i^*}] \wedge [\bigvee_{i\in[1,n]}[x_i \neq x_i^*]]$, where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. $\mathcal{S}$ outputs $(x_1, \cdots, x_n)$ and $(x_1^*, \cdots, x_n^*)$.

Obviously, $\mathcal{S}$ breaks the n-representation assumption. □

<u>Proof of Lemma 5.5.8.</u> We prove that if there exists an PPT adversary $\mathcal{A}$ who makes $\Pr[F_W(\bar{A}_1 \wedge C_2)]$ non-negligible, we can construct a PPT simulator $\mathcal{S}$ who breaks the DL assumption. $\mathcal{S}$ behaves as follows.

Setup. Let $gk := (p, \mathbb{G}, \tilde{\mathbb{G}}, \mathbb{G}_T, \hat{e}, h, \tilde{h}) \overset{\mathrm{U}}{\leftarrow} \mathcal{G}_{pg}(1^\lambda)$. As an instance of the DL problem, $\mathcal{S}$ is given (h, h^α), where $\alpha \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$. She sets $g_0 := h^\alpha$. She runs $(vk, mk) \leftarrow \mathsf{SPSIG.Gen}(gk, 1^L)$. For every $i \in [1, n]$, She chooses $\beta_i \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and sets $g_i := h^{\beta_i}$. She sends $pk := (1^L, gk, vk, g_0, g_1, \cdots, g_n)$ to $\mathcal{A}$.

Query. When $\mathcal{A}$ issues $\mathbb{W} \in \{0,1\}^L$ as a query to the oracle **Reveal**, $\mathcal{S}$ behaves as follows. She chooses $x_1, \cdots, x_n \overset{\mathrm{U}}{\leftarrow} \mathbb{Z}_p$ and computes $y := \prod_{i=1}^{n} g_i^{x_i}$. She generates $\theta \leftarrow \mathsf{SPSIG.Sig}(vk, mk, (y \cdot g_0^{w_1}, \cdots, y \cdot g_0^{w_L}))$. She sends $sk := (x_1, \cdots, x_n, y, \theta)$ to $\mathcal{A}$. She also sets $\mathcal{L}_{\mathbb{W}} := \mathcal{L}_{\mathbb{W}} \cup \{sk\}$.

Compute(ϕ^*, sk^*)**.** $\mathcal{S}$ parses sk^* as $(x_1^*, \cdots, x_n^*, y^*, \theta^*)$. Assuming that $F_W(\bar{A}_1 \wedge C_2)$ occurs implies that the following event occurs.

$$\left[y^* = \prod_{i=1}^{n} g_i^{x_i^*}\right] \wedge \left[\exists \mathbb{W}^* \ s.t. \ [\phi^*(\mathbb{W}^*) = 1] \wedge \left[1 \leftarrow \mathsf{SPSIG.Ver}(vk, \theta^*, (y^* \cdot g_0^{w_1^*}, \cdots, y^* \cdot g_0^{w_L^*}))\right]\right.$$
$$\left.\wedge \left[\bigvee_{\mathbb{W} \in \{0,1\}^L \ s.t. \ [\mathcal{L}_{\mathbb{W}} \neq \emptyset] \wedge [\phi^*(\mathbb{W}) = 1] \wedge [\mathbb{W} \neq \mathbb{W}^*]} \left[\bigvee_{sk \in \mathcal{L}_{\mathbb{W}}} \left[[y \neq y^*] \bigwedge_{i \in [1,L]} \left[y \cdot g_0^{w_i} = y^* \cdot g_0^{w_i^*}\right]\right]\right]\right]\right],$$

where sk is parsed as $(x_1, \cdots, x_n, y, \theta)$. Since $\mathbb{W} \neq \mathbb{W}^*$, there exists $i \in [1, L]$ such that $w_i \neq w_i^*$. Given the integer i, $\mathcal{S}$ computes $\alpha' := (\sum_{j=1}^{n} \beta_j \cdot (x_j - x_j^*))/(w_i^* - w_i)$ and outputs it.

Let us explain why $\mathcal{S}$ breaks the DL assumption. Applying the fact that $g_0 = h^\alpha$ and $g_i = h^{\beta_i}$ to the equation $y \cdot g_0^{w_i} = y^* \cdot g_0^{w_i^*}$, we obtain $\alpha = (\sum_{j=1}^{n} \beta_j \cdot (x_j - x_j^*))/(w_i^* - w_i) = \alpha'$. Since $y \neq y^*$, $\bigvee_{j \in [1,n]} \left[x_j \neq x_j^*\right]$. Hence, she can compute $\alpha' (= \alpha)$ correctly. □

Proof of Lemma 5.5.9. For every $sk \in \mathcal{L}_{\mathbb{W}^*}$, if $y \neq y^*$, then for every $i \in [1, L]$, $y \cdot g_0^{w_i^*} \neq y^* \cdot g_0^{w_i^*}$. Thus, the event $F_W(\bar{A}_1 \wedge C_3)$ never occurs. □

5.6 Conclusion for Chapter 5

No IBS/ABS schemes whose hard-to-invert leakage-resilience is guaranteed under standard assumptions have been known. More generally, no IBS/ABS schemes whose leakage-resilience is guaranteed under standard assumptions have been known.

In this work, we defined existential unforgeability considering hard-to-invert leakage-resilience for IBS schemes. Then, we generically constructed an IBS scheme, and proved that it is secure under our definition. Additionally, we instantiated it under the DLIN and SXDH assumptions. The instantiation is the first one whose leakage-resilience is guaranteed under standard assumptions.

Similarly, we achieved a result on attribute-based signature whose predicate is represented as a general circuit. Specifically, we defined existential unforgeability considering hard-to-invert leakage-resilience and computational signer-privacy for ABS schemes whose predicate is represented as a general circuit. After that, we generically constructed an ABS scheme for a circuit and proved that it is unforgeable and signer-private under our definitions. Then, we instantiated it under the DLIN and SXDH assumptions. The instantiation is the first one whose leakage-resilience is guaranteed under standard assumptions.

Chapter 6

Conclusion

Cryptosystems' secret-information leakage made by various causes such as side-channel attacks is extremely hard for us to perfectly prevent. Thus, it is one of the most serious threats. As a result, leakage-resilience which guarantees that if some related-information about secret-information is leaked, the security is maintained, is practically demanded. Various security models considering leakage-resilience have been proposed. Especially, hard-to-invert leakage (HL) model is considered to be theoretically/practically the most meaningful. In this thesis, we accomplished the following three research tasks concerning HL-resilience.

The first one is regarding IBE scheme with HL-resilience. The IBE scheme proposed by Yuen et al. at EUROCRYPT'12 [YCZY12] is known as the only one claimed to be correctly proven to be secure in a security model with HL-resilience. In this work, we showed that their security proof is defective by presenting some concrete counterexamples of PPT adversaries which indicate the deficiency. Moreover, we proposed an original IBE construction and proved that it is secure in an adaptive IND-CPA security model considering HL-resilience under the DLIN assumption. Thus, our IBE scheme is currently only one whose HL-resilience was correctly proven.

The second one is regarding digital signature with HL-resilience. We proposed a generic construction of digital signature, and proved that it is secure in sEUF-CMA security model considering resilience to polynomially hard-to-invert leakage. After that, we proved that it can be instantiated under the DLIN and SXDH assumptions. Currently, as far as we know, three digital signature schemes with HL-resilience have been proposed by Faust et al. at ASIACRYPT'12 [FHN$^+$12] and the others [YYH12, WMHT16]. Among them, our instantiation of digital signature is not only the only one resilient to polynomially hard-to-invert leakage under standard assumptions, but also the only one secure in an sEUF-CMA security model with HL-resilience.

The third one is regarding ABS/IBS schemes with HL-resilience. A lot of ABS/IBS schemes secure in non-leakage setting have been proposed, e.g., [PS06, MPR11, OT11, SAH16]. However, no schemes secure in a security model with some leakage-resilience under standard assumptions have been proposed, as far as we know. We proposed generic constructions of ABS/IBS schemes, and proved that they are secure in an adaptive wEUF-CMA security models with HL-resilience. For the ABS scheme, we also proved that it is computationally signer-private. Moreover, we showed that they can be instantiated under the DLIN and SXDH assumptions. It should be noted that our ABS/IBS instantiations are the first ones proven to be secure in HL model under standard assumptions, and more generally, the first ones proven to be leakage-resilient under standard assumptions.

A lot of open problems related to leakage-resilient cryptography are left. In the area of leakage-resilient cryptography, BL model has developed the fastest because of the simplest definition of the model. Thus, there are many problems which have been solved in BL model, but have not been solved yet in HL model. The followings are some examples of such problems, and we can objectively say that our results in this thesis more or less contribute to solving the open problems.

We know that there are many cryptographic primitives such that concrete schemes secure in BL model have been proposed, but ones secure in HL model have not been proposed, e.g., attribute-based encryption, functional encryption, inner-product encryption, fully homomorphic encryption, and etc. Proposing the first scheme of such primitives is meaningful. Secondly, presenting the first scheme secure in a combination model of CL model and HL model is also meaningful. CL model [BKKV10, DHLAW10a] generally means a model where the leakage function during each period is restricted in its output bit-length like BL model, whereas a combination of CL and HL models means a model where the leakage function is required to be hard-to-invert like HL model. Thirdly, proposing the first scheme resilient to both of the implementation attacks, (hard-to-invert) leakage and *tampering*, must be meaningful. Tampering attack means an attack which makes the attacker to modify the secret-key of cryptographic devices and observe the behavior of the devices. PKE and digital signature schemes secure against bounded leakage and tampering have been proposed [FX16]. It must be natural to think how can we realize schemes secure against hard-to-invert leakage and tampering.

Bibliography

[AARR02] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. The EM side-channel (s). In *CHES 2002*, volume 2523 of LNCS, pages 29–45. Springer, 2002.

[ABB10] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT 2010*, volume 6110 of LNCS, pages 553–572. Springer, 2010.

[ADN$^+$10] J. Alwen, Y. Dodis, M. Naor, G. Segev, S. Walfish, and D. Wichs. Public-key encryption in the bounded-retrieval model. In *EUROCRYPT 2010*, volume 6110 of LNCS, pages 553–572. Springer, 2010.

[ADVW13] S. Agrawal, Y. Dodis, V. Vaikuntanathan, and D. Wichs. On continual leakage of discrete log representations. In *ASIACRYPT 2013*, volume 8270 of LNCS, pages 401–420. Springer, 2013.

[ADW09] J. Alwen, Y. Dodis, and D. Wichs. Leakage-resilient public-key cryptography in the bounded-retrieval model. In *CRYPTO 2009*, volume 5677 of LNCS, pages 36–54. Springer, 2009.

[AGV09] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *TCC 2009*, volume 5444 of LNCS, pages 474–495. Springer, 2009.

[And97] R. Anderson. Two remarks on public key cryptology. `http://www.cl.cam.ac.uk/users/rja14`, 1997.

[BB04a] D. Boneh and X. Boyen. Efficient selective-ID secure identity-based encryption without random oracles. In *EUROCRYPT 2004*, volume 3027 of LNCS, pages 223–238. Springer, 2004.

[BB04b] D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In *CRYPTO 2004*, volume 3152 of LNCS, pages 443–459. Springer, 2004.

[BBS04] D. Boneh, X. Boyen, and H. Shacham. Short group signatures. In *CRYPTO 2004*, volume 3152 of LNCS, pages 41–55. Springer, 2004.

[BDIR18] F. Benhamouda, A. Degwekar, Y. Ishai, and T. Rabin. On the local leakage resilience of linear secret sharing schemes. In *CRYPTO 2018*, volume 10991 of LNCS, pages 531–561. Springer, 2018.

[BF01] D. Boneh and M. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO 2001*, volume 2139 of LNCS, pages 213–229. Springer, 2001.

[BG10] Z. Brakerski and S. Goldwasser. Circular and leakage resilient public-key encryption under subgroup indistinguishability (or: quadratic residuosity strikes back). In *CRYPTO 2010*, volume 6223 of LNCS, pages 1–20. Springer, 2010.

[BGG94] M. Bellare, O. Goldreich, and S. Goldwasser. Incremental cryptography: the case of hashing and signing. In *CRYPTO 1994*, volume 839 of LNCS, pages 216–233. Springer, 1994.

[BGJK12] E. Boyle, S. Goldwasser, A. Jain, and Y.T. Kalai. Multiparty computation secure against continual memory leakage. In *STOC 2012*, pages 1235–1254. ACM, 2012.

[BHHO08] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *CRYPTO 2008*, volume 5157 of LNCS, pages 108–125. Springer, 2008.

[BKKV10] Z. Brakerski, Y.T. Kalai, J. Katz, and V. Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS 2010*, pages 501–510. IEEE, 2010.

[BL14] A. Berkoff and F.H. Liu. Leakage resilient fully homomorphic encryption. In *TCC 2014*, volume 8349 of LNCS, pages 515–539. Springer, 2014.

[Ble98] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the rsa encryption standard pkcs# 1. In *CRYPTO 1998*, volume 1462 of LNCS, pages 1–12. Springer, 1998.

[BM99] M. Bellare and S.K. Miner. A forward-secure digital signature scheme. In *CRYPTO 1999*, volume 1666 of LNCS, pages 431–448. Springer, 1999.

[BOS17] M. Bellare, A. O'Neill, and I. Stepanovs. Forward-security under continual leakage. In *CANS 2017*, volume 11261 of LNCS, pages 3–26. Springer, 2017.

[Bra93] S.A. Brands. An efficient off-line electronic cash system based on the representation problem. Technical report, CWI, 1993.

[BSW06] D. Boneh, E. Shen, and B. Waters. Strongly unforgeable signatures based on computational diffie-hellman. In *PKC 2006*, volume 3958 of LNCS, pages 229–240. Springer, 2006.

[BSW11] E. Boyle, G. Segev, and D. Wichs. Fully leakage-resilient signatures. In *EUROCRYPT 2011*, volume 6632 of LNCS, pages 89–108. Springer, 2011.

[CCS09] J. Camenisch, N. Chandran, and V. Shoup. A public key encryption scheme secure against key dependent chosen plaintext and adaptive chosen ciphertext attacks. In *EUROCRYPT 2009*, volume 5479 of LNCS, pages 351–368. Springer, 2009.

[CDRW10] S.S.M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters. Practical leakage-resilient identity-based encryption from simple assumptions. In *ACMCCS 2010*, pages 152–161. ACM, 2010.

[CHK03] R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In *EUROCRYPT 2003*, volume 2656 of LNCS, pages 255–271. Springer, 2003.

[CHK04] R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT 2004*, volume 3027 of LNCS, pages 207–222. Springer, 2004.

[CNS99] J.S. Coron, D. Naccache, and J.P. Stern. On the security of rsa padding. In *CRYPTO 1999*, volume 1666 of LNCS, pages 1–18. Springer, 1999.

[Cor99] J.S. Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In *CHES 1999*, pages 292–302. Springer, 1999.

[CQX18] Y. Chen, B. Qin, and H. Xue. Regular lossy functions and their applications in leakage-resilient cryptography. In *CT-RSA 2018*, volume 10808 of LNCS, pages 491–511. Springer, 2018.

[CZLC16] Y. Chen, Z. Zhang, D. Lin, and Z. Cao. Generalized (identity-based) hash proof system and its applications. *Security and Communication Networks*, 9(12):1698–1716, 2016.

[DCLW06] G. Di Crescenzo, R.J. Lopton, and S. Walfish. Perfectly secure password protocols in the bounded retrieval model. In *TCC 2006*, volume 3876 of LNCS, pages 225–244. Springer, 2006.

[Des87] Y. Desmedt. Society and group oriented cryptography: A new concept. In *CRYPTO 1987*, volume 293 of LNCS, pages 120–127. Springer, 1987.

[DF89] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *CRYPTO 1989*, volume 435 of LNCS, pages 307–315. Springer, 1989.

[DGK$^+$10] Y. Dodis, S. Goldwasser, Y.T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption with auxiliary inputs. In *TCC 2010*, volume 5978 of LNCS, pages 361–381. Springer, 2010.

[DHLAW10a] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Cryptography against continuous memory attacks. In *FOCS 2010*, pages 511–520. IEEE, 2010.

[DHLAW10b] Y. Dodis, K. Haralambiev, A. López-Alt, and D. Wichs. Efficient public-key cryptography in the presence of key leakage. In *ASIACRYPT 2010*, volume 6477 of LNCS, pages 613–631. Springer, 2010.

[DKL09] Y. Dodis, Y.T. Kalai, and S. Lovett. On cryptography with auxiliary input. In *STOC 2009*, pages 621–630. ACM, 2009.

[DKXY02] Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-insulated public key cryptosystems. In *EUROCRYPT 2002*, volume 2332 of LNCS, pages 65–82. Springer, 2002.

[DP08] S. Dziembowski and K. Pietrzak. Leakage-resilient cryptography. In *FOCS 2008*, pages 293–302. IEEE, 2008.

[Dzi06] S. Dziembowski. Intrusion-resilience via the bounded-storage model. In *TCC 2006*, volume 3876 of LNCS, pages 207–224. Springer, 2006.

[FHN$^+$12] S. Faust, C. Haway, J.B. Nielsen, P.S. Nordholt, and A. Zottarel. Signature schemes secure against hard-to-invert leakage. In *ASIACRYPT 2012*, volume 7658 of LNCS, pages 98–115. Springer, 2012.

[FNV15] A. Faonio, J.B. Nielsen, and D. Venturi. Mind your coins: fully leakage-resilient signatures with graceful degradation. In *ICALP 2015*, volume 9134 of LNCS, pages 456–468. Springer, 2015.

[FX16] E. Fujisaki and K. Xagawa. Public-key cryptosystems resilient to continuous tampering and leakage of arbitrary functions. In *ASIACRYPT 2016*, volume 10031 of LNCS, pages 908–938. Springer, 2016.

[GJS11] S. Garg, A. Jain, and A. Sahai. Leakage-resilient zero knowledge. In *CRYPTO 2011*, volume 6841 of LNCS, pages 297–315. Springer, 2011.

[GM84] S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of computer and system sciences*, 28(2):270–299, 1984.

[GMO01] K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic analysis: concrete results. In *CHES 2001*, volume 2162 of LNCS, pages 251–361. Springer, 2001.

[GPT14] D. Genkin, I. Pipman, and E. Tromer. Get your hands off my laptop: Physical side-channel key-extraction attacks on pcs. In *CHES 2014*, volume 8731 of LNCS, pages 242–260. Springer, 2014.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC 2008*, pages 197–206. ACM, 2008.

[GS08] J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *EUROCRYPT 2008*, volume 4965 of LNCS, pages 415–432. Springer, 2008.

[GST14] D. Genkin, A. Shamir, and E. Tromer. RSA key extraction via low-bandwidth acoustic cryptanalysis. In *CRYPTO 2014*, volume 8616 of LNCS, pages 444–461. Springer, 2014.

[GV12] D. Galindo and S. Vivek. A practical leakage-resilient signature scheme in the generic group model. In *SAC 2012*, volume 7707 of LNCS, pages 50–65. Springer, 2012.

[HHP16] J. Huang, Q. Huang, and C. Pan. A black-box construction of strongly unforgeable signature schemes in the bounded leakage model. In *ProvSec 2016*, volume 10005 of LNCS, pages 320–339. Springer, 2016.

[HLAWW16] C. Hazay, A. Lopez-Alt, H. Wee, and D. Wichs. Leakage-resilient cryptography from minimal assumptions. *Journal of Cryptology*, 29(3):514–551, 2016.

[HS13] M. Hutter and J.M. Schmidt. The temperature side channel and heating fault attacks. In *CARDIS 2013*, volume 8419 of LNCS, pages 219–235. Springer, 2013.

[HSH$^+$08] J.A. Halderman, S.D. Schoen, N. Heninger, W. Clarkson, W. Paul, J.A. Calandrino, A.J. Feldman, J. Appelbaum, and E.W. Felten. Lest we remember: cold boot attacks on encryption keys. In *USENIX Security Symposium 2008*, pages 45–60. USENIX Association, 2008.

[HWZ07] Q. Huang, D.S. Wong, and Y. Zhao. Generic transformation to strongly unforgeable signatures. In *ACNS 2007*, volume 4521 of LNCS, pages 1–17. Springer, 2007.

[IM18] M. Ishizaka and K. Matsuura. Strongly unforgeable signature resilient to polynomially hard-to-invert leakage under standard assumptions. In *ISC 2018*, volume 11060 of LNCS, pages 422–441. Springer, 2018.

[Kiy15] S. Kiyoshima. An alternative approach to non-black-box simulation in fully concurrent setting. In *TCC 2015*, volume 9014 of LNCS, pages 290–318. Springer, 2015.

[KJJ99] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *CRYPTO 1999*, volume 1666 of LNCS, pages 388–397. Springer, 1999.

[KKS11] Y.T. Kalai, B. Kanukurthi, and A. Sahai. Cryptography with tamperable and leaky memory. In *CRYPTO 2011*, volume 6841 of LNCS, pages 373–390. Springer, 2011.

[Koc96] P. Kocher. Timing attacks on implementations on Diffie-Hellman, RSA, DSS, and other systems. In *CRYPTO 1996*, volume 4521 of LNCS, pages 1–17. Springer, 1996.

[KP10] E. Kiltz and K. Pietrzak. Leakage resilient elgamal encryption. In *ASIACRYPT 2010*, volume 6477 of LNCS, pages 595–612. Springer, 2010.

[KP13] K. Kurosawa and L.T. Phong. Leakage resilient IBE and IPE under the DLIN assumption. In *ACNS 2013*, volume 7954 of LNCS, pages 487–501. Springer, 2013.

[KPW15] E. Kiltz, J. Pan, and H. Wee. Structure-preserving signatures from standard assumptions, revisited. In *CRYPTO 2015*, volume 9216 of LNCS, pages 275–295. Springer, 2015.

[KV09] J. Katz and V. Vaikuntanathan. Signature schemes with bounded leakage resilience. In *ASIACRYPT 2009*, volume 5912 of LNCS, pages 703–720. Springer, 2009.

[LLW11] A. Lewko, M. Lewko, and B. Waters. How to leak on key updates. In *STOC 2011*, pages 725–734. ACM, 2011.

[LRW11] A. Lewko, Y. Rouselakis, and B. Waters. Achieving leakage resilience through dual system encryption. In *TCC 2011*, volume 6597 of LNCS, pages 70–88. Springer, 2011.

[LTZY16] J. Li, M. Teng, Y. Zhang, and Q. Yu. A leakage-resilient cca-secure identity-based encryption scheme. *The Computer Journal*, 59(7):1066–1075, 2016.

[LW10] A. Lewko and B. Waters. New techniques for dual system encryption and fully secure HIBE with short ciphertexts. In *TCC 2010*, volume 5978 of LNCS, pages 455–479. Springer, 2010.

[MDS99] T.S. Messerges, E.A. Dabbish, and R.H. Sloan. Power analysis attacks of modular exponentiation in smartcards. In *CHES 1999*, volume LNCS of 1717, pages 144–157. Springer, 1999.

[MPR11] H.K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In *CT-RSA 2011*, volume 6558 of LNCS, pages 376–392. Springer, 2011.

[MR04] S. Micali and L. Reyzin. Physically observable cryptography. In *TCC 2004*, volume 2951 of LNCS, pages 278–296. Springer, 2004.

[MTVY11] T. Malkin, I. Teranishi, Y. Vahlis, and M. Yung. Signatures resilient to continual leakage on memory and computation. In *TCC 2011*, volume 6597 of LNCS, pages 89–108. Springer, 2011.

[NS09] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. In *CRYPTO 2009*, volume 5677 of LNCS, pages 18–35. Springer, 2009.

[OT11] T. Okamoto and K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *PKC 2011*, volume 4058 of LNCS, pages 207–222. Springer, 2011.

[Pan14] O. Pandey. Achieving constant round leakage-resilient zero-knowledge. In *TCC 2014*, volume 6841 of LNCS, pages 146–166. Springer, 2014.

[PS06] K.G. Paterson and J.C.N. Schuldt. Efficient identity-based signatures secure in the standard model. In *ACISP 2006*, volume 4058 of LNCS, pages 207–222. Springer, 2006.

[QL13] B. Qin and S. Liu. Leakage-resilient chosen-ciphertext secure public-key encryption from hash proof system and one-time lossy filter. In *ASIACRYPT 2013*, volume 8270 of LNCS, pages 381–400. Springer, 2013.

[QL14] B. Qin and S. Liu. Leakage-flexible cca-secure public-key encryption: simple construction and free of pairing. In *PKC 2014*, volume 8383 of LNCS, pages 19–36. Springer, 2014.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005*, pages 84–93. ACM, 2005.

[RS91] C. Rackoff and D.R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO 1991*, volume 576 of LNCS, pages 433–444. Springer, 1991.

[SAH16] Y. Sakai, N. Attrapadung, and G. Hanaoka. Attribute-based signatures for circuits from bilinear map. In *PKC 2016*, volume 9612 of LNCS, pages 283–300. Springer, 2016.

[Sch00] W. Schindler. A timing attack against RSA with the Chinese remainder theorem. In *CHES 2000*, volume 1965 of LNCS, pages 109–124. Springer, 2000.

[Sha84] A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO 1984*, volume 196 of LNCS, pages 47–53. Springer, 1984.

[SHGL16] S.F. Sun, S. Han, D. Gu, and S. Liu. Public key cryptosystems secure against memory leakage attacks. *IET Information Security*, 10(6):403–412, 2016.

[SPW07] R. Steinfeld, J. Pieprzyk, and H. Wang. How to strengthen any weakly unforgeable signature into a strongly unforgeable signature. In *CT-RSA 2007*, volume 4377 of LNCS, pages 357–371. Springer, 2007.

[ST04] A. Shamir and E. Tromer. Acoustic cryptanalysis: on noisy people and noisy machines. EUROCRYPT Rump Session, 2004.

[Sta11] F.X. Standaert. Leakage resilient cryptography: a practical overview. Invited talk at SKEW 2011 (ECRYPT Workshop on Symmetric Encryption 2011), 2011.

[TLNL14] F. Tang, H. Li, Q. Niu, and B. Liang. Efficient leakage-resilient signature schemes in the generic bilinear group model. In *ISPEC 2014*, volume 8434 of LNCS, pages 418–432. Springer, 2014.

[Wat05] B. Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT 2005*, volume 3494 of LNCS, pages 114–127. Springer, 2005.

[Wat09] B. Waters. Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO 2009*, volume 5677 of LNCS, pages 619–639. Springer, 2009.

[WCLH18] H. Wang, K. Chen, J. K. Liu, and Z. Hu. Leakage-resilient chosen-ciphertext secure functional encryption from garbled circuits. In *ISPEC 2018*, volume 11125 of LNCS, pages 119–140. Springer, 2018.

[WMHT16] Y. Wang, T. Matsuda, G. Hanaoka, and K. Tanaka. Signatures resilient to uninvertible leakage. In *SCN 2016*, volume 9841 of LNCS, pages 372–390. Springer, 2016.

[WT14] Y. Wang and K. Tanaka. Generic transformation to strongly existentially unforgeable signature schemes with leakage resiliency. In *ProvSec2014*, volume 8782 of LNCS, pages 117–129. Springer, 2014.

[WT15] Y. Wang and K. Tanaka. Generic transformation to strongly existentially unforgeable signature schemes with continuous leakage resiliency. In *ACISP 2015*, volume 9144 of LNCS, pages 213–229. Springer, 2015.

[WTH16] J.D. Wu, Y.M. Tseng, and S.S. Huang. Leakage-resilient id-based signature scheme in the generic bilinear group model. *Security and Communication Networks*, 9(17):3987–4001, 2016.

[WY15] Z. Wang and S.M. Yiu. Attribute-based encryption resilient to auxiliary input. In *ProvSec 2015*, volume 9451 of LNCS, pages 371–390. Springer, 2015.

[YAX$^+$16] Z. Yu, M.H. Au, Q. Xu, R. Yang, and J. Han. Leakage-resilient functional encryption via pair encodings. In *ACISP 2016*, volume 9722 of LNCS, pages 443–460. Springer, 2016.

[YCZY12] T.H. Yuen, S.S.M. Chow, Y. Zhang, and S.M. Yiu. Identity-based encryption resilient to continual auxiliary leakage. In *EUROCRYPT 2012*, volume 7232 of LNCS, pages 117–134. Springer, 2012.

[YYH12] T.H. Yuen, S.M. Yiu, and L.C.K. Hui. Fully leakage-resilient signatures with auxiliary inputs. In *ACISP 2012*, volume 7372 of LNCS, pages 294–307. Springer, 2012.

[ZCG$^+$18] J. Zhang, J. Chen, J. Gong, A. Ge, and C. Ma. Leakage-resilient attribute based encryption in prime-order groups via predicate encodings. *Designs, Codes and Cryptography*, 86(6):1339–1366, 2018.

[ZCQ12] Z. Zhang, Z. Cao, and H. Qian. Chosen-ciphertext attack secure public key encryption with auxiliary inputs. *Security and Communication Networks*, 5(12):1404–1411, 2012.

[Zha14] M. Zhang. New model and construction of ABE: achieving key resilient-leakage and attribute direct-revocation. In *ACISP 2014*, volume 8544 of LNCS, pages 192–208. Springer, 2014.

[ZSW$^+$13] M. Zhang, W. Shi, C. Wang, Z. Chen, and Y. Mu. Leakage-resilient attribute-based encryption with fast decryption: models, analysis and constructions. In *ISPEC 2013*, volume 7863 of LNCS, pages 75–90. Springer, 2013.

[ZWTM13] M. Zhang, C. Wang, T. Takagi, and Y. Mu. Functional encryption resilient to hard-to-invert leakage. *The Computer Journal*, 58(4):735–749, 2013.

Appendix A

Publication List

International Journal Papers (Refereed):

[A1] M. Ishizaka, S. Ohata and K. Matsuura. Generic construction of ciphertext-policy attribute-based signcryption secure in the adaptive predicate model. In *IPSI Transactions on Advanced Research*, volume 12, number 2, pages 16–26, 2016. IPSI BgD. (†*Neither directly nor indirectly included in this thesis.*)

International Conference Papers (Refereed):

[B1] M. Ishizaka and K. Matsuura. Strongly unforgeable signature resilient to polynomially hard-to-invert leakage under standard assumptions. In *21st Information Security Conference (ISC 2018)*, volume 11060 of LNCS, pages 422-441, Guildford, United Kingdom, September 2018. Springer.

[B2] M. Ishizaka and K. Matsuura. Identity-based encryption resilient to auxiliary leakage under the decisional linear assumption. In *17th International Conference on Cryptology And Network Security (CANS 2018)*, volume 11124 of LNCS, pages 417–439, Naples, Italy, September 2018. Springer.

[B3] J. Hayata, M. Ishizaka, Y. Sakai, G. Hanaoka, K. Matsuura. Generic construction of adaptively secure anonymous key-policy attribute-based encryption from public-key searchable encryption. In *2018 International Symposium on Information Theory and Its Applications (ISITA 2018)*, Singapore, October 2018. IEEE. (†*Neither directly nor indirectly included in this thesis.*)

Domestic Conference Papers (Non-Refereed):

[C1] 石坂理人, 大畑幸矢, 松浦幹太. 適応的述語安全な暗号文ポリシー型属性ベース Signcryption の一般的構成法. In *2016 年 暗号と情報セキュリティシンポジウム (SCIS 2016)*, 鹿児島, 2016 年 1 月. 電子情報通信学会. (**Written in Japanese.*) (†*Neither directly nor indirectly included in this thesis.*)

[C2] 石坂理人, 松浦幹太. ID ベース暗号方式 (黒澤・Phong, ACNS'13) の補助漏洩耐性の証明. In *2017 年 暗号と情報セキュリティシンポジウム (SCIS 2017)*, 沖縄, 2017 年 1 月. 電子情報通信学会. (**Written in Japanese.*)

[C3] 石坂理人, 松浦幹太. Continual Auxiliary Leakage に耐性を持つ適応的安全な述語署名. In *2017 年 コンピュータセキュリティシンポジウム (CSS 2017)*, 山形, 2017 年 10 月. 情報処理学会. (**Written in Japanese.*)

[C4] 林田淳一郎, 石坂理人, 坂井祐介, 花岡悟一郎, 松浦幹太. 公開鍵型検索可能暗号を用いた適応的安全な匿名鍵ポリシー型属性ベース暗号の一般的構成. In *2018 年 暗号と情報セキュリティシンポジウム (SCIS 2018)*, 新潟, 2018 年 1 月. 電子情報通信学会. (**Written in Japanese.*) (†*Neither directly nor indirectly included in this thesis.*)

[C5] M. Ishizaka, K. Matsuura. Identity/attribute-based signatures resilient to hard-to-invert leakage under standard assumptions. In *2018 年 コンピュータセキュリティシンポジウム (CSS 2018)*, 長野, 2018 年 10 月. 情報処理学会.

[C6] 石坂理人, 松浦幹太. DLIN 仮定下で強偽造困難性及び多項式的逆変換困難漏洩耐性を持つ電子署名. In *2019 年 暗号と情報セキュリティシンポジウム (SCIS 2019)*, 滋賀, 2019 年 1 月. 電子情報通信学会. (**Written in Japanese.*)

Preprint Papers:

[D1] M. Ishizaka, K. Matsuura. Strongly unforgeable signatures resilient to polynomially hard-to-invert leakage under standard assumptions. Cryptology ePrint Archive: Report 2018/1044, 2018.

Awards:

[E1] *Best Student Papers Award* of CSS2017 (awarded for [C3])

[E2] *Best Paper Award* of ISC2018 (awarded for [B1])