Strong Authenticated Key Exchange with Auxiliary Inputs*

Rongmao Chen[†], Yi Mu[‡], Guomin Yang[‡], Willy Susilo[‡], and Fuchun Guo[‡]

[†]College of Computer
National University of Defense Technology, China
chromao@nudt.edu.cn
[‡]Institute of Cybersecurity and Cryptology
School of Computing and Information Technology
University of Wollongong, Australia
{ymu,gyang,wsusilo,fuchun}@uow.edu.au

Abstract. Leakage attacks, including various kinds of side-channel attacks, allow an attacker to learn partial information about the internal secrets such as the secret key and the randomness of a cryptographic system. Designing a strong, meaningful, yet achievable security notion to capture practical leakage attacks is one of the primary goals of leakage-resilient cryptography. In this work, we revisit the modelling and design of authenticated key exchange (AKE) protocols with leakage resilience. We show that the prior works on this topic are inadequate in capturing realistic leakage attacks. To close this research gap, we propose a new security notion named *leakage-resilient eCK model w.r.t. auxiliary inputs* (AI-LR-eCK) for AKE protocols, which addresses the limitations of the previous models. Our model allows computationally hard-to-invert leakage of *both the long-term secret key and the randomness*, and also addresses a limitation existing in most of the previous models where the adversary is disallowed to make leakage queries during the challenge session. As another major contribution of this work, we present a generic framework for the construction of AKE protocols that are secure under the proposed AI-LR-eCK model. An instantiation based on the *Decision Diffie-Hellman* (DDH) assumption in the standard model is also given to demonstrate the feasibility of our proposed framework.

1 Introduction

Authenticated Key Exchange (AKE) protocols, which are among the most widely used cryptographic primitives, form a central component in many network standards, such as IPSec, SSL/TLS, SSH. An AKE protocol enables a secure channel to be established among a set of communicating parties by allowing them to agree on a common secret key. Practical AKE protocols such as the ISO protocol (a.k.a. SIG-DH) [**?**,**?**], the Internet Key Exchange protocol (a.k.a. SIGMA) [**?**] and their variants have been proposed and deployed in the aforementioned network standards.

In order to formally define the security of an AKE protocol, Bellare and Rogaway (BR) [**?**] proposed the first formal complexity theoretic security notion for AKE. The BR model and its variants are nowadays the *de facto* standard for AKE security analysis. In particular, the Canetti-Krawczyk (CK) model [**?**], which can be considered as the extension and combination of the BR model and the Bellare-Canetti-Krawczyk (BCK) model [**?**], has been used to prove the security of many widely used AKE protocols such as SIG-DH and SIGMA. Noting that the CK model does not capture several attacks such as the Key Compromise Impersonation (KCI) and Ephemeral Key Compromise attacks, LaMacchia et al. [**?**] introduced an extension of the CK model, named eCK model, to consider stronger adversaries (in some aspects). We should note that the CK model and the eCK model are incomparable, and refer the readers to Choo et al. [**?**] for a detailed summary of the differences among the aforementioned AKE models.

* A previous version of this paper is published in the Designs, Codes and Cryptography (available at http://link.springer.com/article/10.1007/s10623-016-0295-3). This is the full version by fixing a subtle flaw in the security proof.

Leakage Attacks In Reality. Although the security notion of AKE has been extensively studied by the research community in the last two decades, all the aforementioned AKE security models belong to the traditional "black-box" setting, where honest parties are viewed as interactive Turing machines, and each party has its own private memory and randomness source. In the real world, however, attackers do not always obey such a somewhat idealized setting. Specifically, the physical implementation of a cryptographic system can leak sensitive information through physical channels, such as the computation-time, power-consumption, radiation/noise/heat emission etc. Through these "side channels", an attacker is able to learn some imperfect information of the user secret [**?**,**?**,**?**]. Moreover, potential weakness of the randomness can be caused due to different reasons, such as the side-channel leakages mentioned above and the poor implementation of pseudo-random number generators (PRNGs). Security vulnerabilities involving either flawed or weakened PRNGs have already been reported in the literature [**?**,**?**,**?**]. In consequence, an AKE protocol proven secure in the traditional model could be completely insecure in the presence of leakage attacks.

Modelling Leakage Attacks. The seminal research of defending against leakage attacks by treating them in an abstract way was first proposed by Micali and Reyzin [**?**]. Since then significant progress has been done within the cryptographic community to incorporate leakage attacks into the traditional black-box security definitions. Informally, in the leakage setting the adversary is allowed to learn the partial information of a user secret via an abstract leakage function f in addition to the normal black-box interaction with an honest party. More precisely, the adversary is provided with access to a leakage oracle: the adversary can query the oracle with a polynomial-time computable function f, and then receive $f(\mathsf{sk})$, where sk is the user secret key. It is obvious that if we allow the adversary to choose any leakage function f, then it will lead to the leakage of the key in its entirety and the security of the system is surely compromised. Hence certain restrictions on the class of *admissible* leakage functions are necessary. For example, in the work by Micali and Reyzin [**?**], they put the restrictions on the leakage function so that only computation leaks information and the amount of leakage per occurrence is bounded by a leakage parameter. *The challenge is to model the necessary restrictions in a meaningful and reasonable manner so that we can capture real leakage attacks and also make the security notion achievable*.

To address the above problem, several leakage models have been proposed in the literature. Inspired by the cold boot attack presented by Halderman et al. [**?**], Akavia et al. [**?**] formalized a general framework, named bounded leakage model, which explicitly assumes that a leakage attack only reveals a fraction of the secret key to the adversary. Specifically, the output-length of the (adversarial chosen) leakage function is bounded to be strictly less than the length of the secret key. In a subsequent work, Naor and Segev [**?**] relaxed the restriction on the leakage function and only required that the secret key still has a "sufficient" amount of min-entropy left after the leakage. Such a leakage model, named *noisy leakage* model, can capture a wider class of real-world leakage attacks since now the leakage size can be arbitrarily long.

The Auxiliary Input Model. Instead of imposing a min-entropy requirement on the secret key under leakage, a more general model was put forward by Dodis et al. [**?**]. Their notion, named *auxiliary input model*, requires that it is *computationally* hard for the adversary to recover the secret key given the leakage. In other words, this model allows any one-way leakage function f. One can note that such *hard-to-invert* leakage is a generalization of the bounded/noisy leakage model, and hence can capture a larger class of leakage functions. For example, a one-way per-

mutation is allowed to be a leakage function in the auxiliary input model, while disallowed in the bounded/noisy leakage model since it information-theoretically reveals the entire secret key. Moreover, as shown by Dodis et al. [**?**], the auxiliary input model offers additional advantages in terms of capturing leakage attacks in the real-world. For some cryptographic systems where the secret key (e.g., biometric keys) is used for multiple tasks, the attacker may eventually obtain leakage larger than the upper bound permitted in the bounded/noisy leakage model after a sufficiently long time. However, if the system is secure w.r.t. auxiliary inputs, one can safely use the secret key as long as the total leakage does not (computationally) reveal the entire secret key. Readers are please referred to the work by by Dodis et al. [**?**] for more detailed discussions.

1.1 Gaps Between Existing Works and The Reality for Leakage-Resilient AKE

Leakage-resilient cryptography, particularly leakage-resilient cryptographic primitives such as pseudorandom generators [**?**], signature schemes [**?**], and encryption schemes [**?**,**?**,**?**,**?**], have been extensively studied under different leakage models outlined above. However, only very few works have been done on the modelling and construction of leakage-resilient AKE protocols. This seems surprising given the importance of AKE, but on the other hand is also "understandable" since it is believed that leakage-resilient PKE or signature can be straightforwardly adopted for the construction of AKE protocols resilient to leakage attacks, which has been demonstrated in some existing leakage-resilient AKE constructions. In particular, Alwen et al. [**?**] showed that a leakage-resilient AKE protocol, namely eSIG-DH, can be constructed from an entropically-unforgeable digital signature scheme secure under chosen-message attacks. Dodis et al. [**?**] proposed two new constructions of AKE protocols that are leakage-resilient in the CK security model. Their first construction follows the result of work by Alwen et al. [**?**], i.e., authenticating Diffie-Hellman (DH) key exchange using a leakage-resilient signature scheme. The second construction, named Enc-DH, is based on a leakage-resilient CCA-secure PKE scheme and follows the idea that both parties authenticate each other by requiring the peer entity to correctly decrypt the DH ephemeral public key encrypted under the long-term public key.

However, we notice that the above straightforward approach for constructing leakage-resilient AKE protocols has some limitations and fails to fully capture general leakage attacks in reality due to the following reasons.

– *Limitation on The Assumption.* Although the AKE models in the work [**?**,**?**] consider leakage attacks, they have *an assumption that the leakage would not happen during the challenge AKE session*. More precisely, to guarantee the security of the session key for the challenge (or target) session, the adversary is disallowed to access the leakage oracle during the session. The reason behind such an assumption is that for PKE- and signature-based Diffie-Hellman AKE protocols, the adversary can use the leakage function to break the authentication mechanism based on the PKE or signature in order to launch an active impersonation attack and obtain the session key. The assumption can bypass such a trivial attack, but on the other hand greatly limits the strength of the adversary since in reality leakage attack may happen at any time. Such a definitional difficulty was also recognized by Naor et al. [**?**] and later resolved in the work on leakage-resilient encryption schemes [**?**,**?**]. However, few formal treatments have been proposed for AKE to address this problem. It is worth noting that Alawatugoda et al. [**?**,**?**] addressed this problem in the sense by constructing AKE protocols that are resilient to the after-the-fact leakage which unfortunately suffers from the split-state assumption [**?**]. Moreover, Alwen et al. [**?**]

suggested a solution to remove this somewhat unnatural assumption by instantiating eSIG-DH with a leakage-resilient and existentially-unforgeable (instead of entropically-unforgeable) signature scheme. Nevertheless, their option introduces a new assumption that the amount of leakage must be smaller than the length of a signature.

– *Limitation on The Modelling of Randomness Leakage.* Another limitation of the previous work on leakage-resilient AKE is that they didn't fully capture general leakage attacks, which may involve the randomness (a.k.a. ephemeral secret) used by an AKE protocol. As mentioned before, in reality randomness leakage may occur, e.g., due to the poor implementation of PRNGs [**?**,**?**,**?**]. Also, leakage attacks in practice (e.g., timing or power consumption analysis) can be closely related to the randomness. Since randomness determines the value of a session key and hence plays a crucial role in an AKE protocol, it is necessary to capture randomness leakage in the design and analysis of leakage-resilient AKE. It is worth noting that although the eCK model [**?**] captures the compromising of randomness, it is different from the leakage attack. Specifically, in the eCK model, either long-term secret or ephemeral secret (but not both) of a party can be completely revealed while partial leakage of both secrets is not allowed. In particular, the attacker may reveal the long-term secret and obtain partial leakage of the ephemeral secret. This adversarial capability of the attacker has not ever been captured by any existing model.

– *Limitation on The Leakage Setting.* The leakage model for AKE introduced in the work [**?**,**?**] is an extension of the CK model [**?**] in the bounded leakage setting [**?**]. To be specific, the adversary is given access to a leakage oracle $\mathcal{O}^{l}_{\mathsf{sk}}(\cdot)$, which can be invoked adaptively subjecting to the constraint that the adversary can only learn at most l bits of sk. As outlined above, such a leakage setting is relatively weaker than the auxiliary input setting [**?**]. To obtain AKE protocols secure under the auxiliary input model, it is likely that we can build such a protocol either from a digital signature scheme that is *random message unforgeable with auxiliary input attacks*, e.g., in the work by Faust et al. [**?**], or from an auxiliary input secure encryption scheme, e.g., in the work by Dodis et al. [**?**]. In fact, it has been shown by Yang et al. [**?**] that an AKE protocol secure under CK model with auxiliary input attacks can be built based on a digital signature scheme that is random message unforgeable under random message and auxiliary input attacks. However, as illustrated above, such a straightforward solution cannot overcome the limitations of *unnatural assumption* and *deficient leakage-capturing* (i.e., randomness leakage). Moreover, it is still unknown how to design an eCK-based model under the auxiliary input setting and actually no existing work has considered this issue.

Other Related Works. To remove the *unnatural assumption*, the notion of *after-the-fact leakage* proposed by Halevi and Lin [**?**] appears to be a potential solution, which, actually has been adopted in the work [**?**,**?**] by Alawatugoda et al. Their proposed model allows the adversary to access the leakage oracle during the challenge session execution but implicitly requires the long-term secret has split-state since otherwise their definition is unachievable in the eCK-model. Moreover, the central idea of their AKE construction is to utilize a split-state encryption scheme with a special property (i.e., PG-IND), which is a strong assumption. We also note that it seems not natural to use the split-state approach for dealing with randomness leakage, which is not captured by their model but our goal in this work.

Another potential solution is the second approach proposed by Faust et al. [**?**], which defined an auxiliary input model that only allows *exponentially hard-to-invert* functions instead of

computationally hard-to-invert functions, of the signing key. Since the signing algorithm can be viewed as a polynomially hard-to-invert function, it is excluded from the allowed set of leakage function and hence the adversary can issue the leakage query during the session. However, the exponentially hard-to-invert restriction would greatly reduce the allowed leakage functions and cannot fully capture general leakage attacks.

Noting that some previous work on encryption and signature schemes [**?**,**?**] has considered leakage from the randomness, one may try to incorporate these techniques for AKE constructions to capture leakage of the randomness. Unfortunately, randomness (ephemeral secret) used in AKE protocols is usually separated from that used in the authentication mechanism, i.e., encryption or signature schemes, and the ephemeral key plays a crucial role in determining the session key. As mentioned before, there has been no work addressing the leakage of ephemeral secret key in AKE protocols. In particular, we must ensure that the session key is pseudo-random even if the randomness (i.e., the secret keying material) is leaked, which is a difficult task under the strong leakage setting (e.g., auxiliary input).

We should also note that there are some other research works on leakage-resilient AKE. Whereas these works didn't follow the approach of [**?**,**?**], they still suffer from the same limitations mentioned above. Moriyama and Okamoto [**?**] introduced an eCK-based leakage-resilient model for AKE protocols. However, it only considers the bounded leakage resilience [**?**] of the long-term secret key but not the ephemeral secret key. Moreover, their construction requires the notion of πPRFs (Pseudo-Random Functions), of which the construction under standard model still remains unknown.

Goal of This Work. Based on the aforementioned observations, we can conclude that designing a *strong, meaningful,* yet *achievable* leakage model for AKE is of practical and theoretical importance. In this paper, we give a formal study on AKE protocols in the auxiliary input model under the goal of closing the research gaps outlined above.

Table 1. Comparison with Existing Leakage-Resilient AKE Security Models

AKE Models	**Partial Leakage Setting**[1]			**Basic Models**
	Reasonable Assumption[2]	Randomness Leakage	Leakage Setting	
[**?**]	No	×	**Bounded**	CK
[**?**]	No	×	**Bounded**	CK
[**?**]	No	×	**Auxiliary Input**	CK
[**?**]	No	×	**Bounded**	eCK
[**?**]	No	×	**Bounded**	eCK
This Work	Yes	$\surd$	**Auxiliary Input**	eCK

[1] The partial leakage setting studied in this work is independent from reveal queries in eCK-based AKE models. Here "$\surd$" means partial information of the randomness (a.k.a ephemeral secret) can be leaked through leakage queries.

[2] The works [**?**,**?**,**?**,**?**] suffer from the unnatural assumption that leakage would not happen during the challenge session. The work [**?**] suffer from the split-state assumption.

1.2 Our Contributions

In this work, we investigate both the modelling and construction of AKE in the auxiliary input model, under the goal of developing a strong, meaningful, yet achievable framework that can overcome the limitations of the previous solutions. Particularly, we:

- formulate a new security model named *leakage-resilient eCK model w.r.t. auxiliary inputs* (AI-LR-eCK) for AKE protocols, which is the *first* eCK-based AKE security notion that captures computationally hard-to-invert leakage attacks on both the long-term secret and the randomness (i.e., the ephemeral secret) under some reasonable and necessary restrictions;
- propose a general framework for the construction of 2-round AI-LR-eCK-secure AKE protocols to illustrate the feasibility of the AI-LR-eCK model; We also provide a formal security analysis of the generic construction through a rigorous reduction proof;
- show an instantiation of the proposed general framework by instantiating all the building blocks under the standard model.

1.3 Overview of Our Techniques

In this section we present an overview of our techniques for modelling and constructing AKE with auxiliary input. The overview consists of three parts, i.e., the AI-LR-eCK model, the generic construction and the instantiations. We start with describing more clearly the notion of AI-LR-eCK.

Leakage-Resilient eCK model w.r.t. Auxiliary Inputs. As shown in Table **??**, our proposed AI-LR-eCK model resolves the limitations in the previous leakage models for AKE.

– *Modelling General Leakage Attacks.* We incorporate the auxiliary input model into the eCK security notion in order to capture the *computationally hard-to-invert leakage* of both the *long-term secret* and *ephemeral secret*.

We define the set of admissible leakage functions $\mathcal{H}$ by following the work of Dodis et al. [**?**], i.e., we require a secret key is hard to compute when given the public key in additional to the leakage. The reason for considering such kind of auxiliary input function class is that AKE protocols (in some sense) belong to the public key setting (especially the long-term key) in terms of the authentication part. Since the public key (e.g., lpk) itself leaks some information about the secret key (lsk), as pointed out by Dodis et al. [**?**], it would be impossible to prove that the scheme remains secure w.r.t the class of admissible function that is hard-to-invert without given the public key. More precisely, for the modelling of long-term secret key (denoted by lsk) leakage, we define $\mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ as the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{lsk}|+|\mathsf{lpk}|} \rightarrow \{0,1\}^*$, such that given $(\mathsf{lpk}, h(\mathsf{lsk}, \mathsf{lpk}))$ (lpk is the long-term public key), no probabilistic polynomial-time (PPT) adversary can find lsk with probability greater than ϵ_{lsk}. Similarly, to model the *leakage of randomness* (the ephemeral secret, denoted by esk), we define $\mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$ as the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{esk}|+|\mathsf{epk}|} \rightarrow \{0,1\}^*$, such that given $(\mathsf{epk}, h(\mathsf{esk}, \mathsf{epk}))$ (epk denotes the ephemeral public key), no PPT adversary can find esk with probability greater than ϵ_{esk}.

Our proposed AI-LR-eCK model deserves more interpretation. To model the leakage query from the adversary, we define the leakage query $\mathsf{LongTermKeyLeakage}(f_{\mathsf{lsk}}, \mathsf{pid})$ for the adversary to query the leakage oracle and learn $f_{\mathsf{lsk}}(lsk, lpk)$ where $f_{\mathsf{lsk}} \in \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ and (lsk, lpk) denotes the long-term key pair of party pid. As for the ephemeral secret leakage, we also define the query $\mathsf{EphemeralKeyLeakage}(f_{\mathsf{esk}}, \mathsf{sid})$ for the adversary to access the leakage oracle and learn $f_{\mathsf{esk}}(esk, epk)$ where $f_{\mathsf{esk}} \in \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$ and (esk, epk) denotes the ephemeral key pair used by the owner of the session sid. One should note that separating the leakage queries for the long-term key and the ephemeral key could be reasonable in practice, as these keys are usually not stored in the same place (e.g., the long-term key can be stored in ROM, while ephemeral keys are stored in RAM). To be more precise, in AI-LR-eCK model, the adversary is allowed to

reveal the long-term secret of both parties and obtain partial information of all the ephemeral secrets generated in the AKE protocol through the leakage query. More details are referred to the Table 1. Therefore, compared to existing models, our proposed AI-LR-eCK model is the strongest security model for AKE with auxiliary inputs.

– *A Reasonable Assumption.* A somewhat *unnatural assumption*, namely disallowing leakage queries to be made during the challenge session, has been introduced in the previous models. We should note that it is *impossible* to remove the assumption without any additional treatment(s) (e.g., *split-state* assumption by Alawatugoda et al. [**?**]). The reason is that different from PKE or signature, in AKE the adversary can actively participate in the challenge session. As a result, it can utilize the leakage function to break the authentication mechanism or directly learn the session key.

In this paper, we address this definitional difficulty using a different but more reasonable approach. We observe that in reality, most of the leakage attacks are constrained by the hardware device and measurement methods in their physical implementations. Therefore, as pointed out in the work [**?**,**?**], *a fully adaptive choice of the leakage function may be an overly powerful model* to capture leakage attacks. Inspired by this observation, we modify the fully adaptive definition used in the previous models by asking the adversary to submit two leakage function sets $\mathcal{F}_{\mathsf{lsk}} \subseteq \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$, and $\mathcal{F}_{esk} \subseteq \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$ *prior to* the game setup. During the security game, the adversary is only allowed to *adaptively* query any leakage functions belonging to the committed sets. As the submitted leakage function sets are chosen by the adversary *independently of the system parameters*, the adversary can no longer trivially break the challenge session and hence the aforementioned unnatural assumption can be abolished.

One may object that artificially requiring the adversary to fix the leakage function sets is somewhat counter-intuitive, as it seems that we bypass the definitional difficulty by just reducing the leakage capturing. This is actually no the case. As illustrated above, most of the practical leakage attacks in reality depend on the physical implementation rather than the realistic cryptosystem. In fact, one can note that a similar treatment has also been adopted for other cryptographic schemes with leakage resilience [**?**,**?**,**?**,**?**]. Therefore, compared to the unnatural assumption which disallows the adversary to access the leakage query during the challenge session, our assumption is meaningful in capturing most of the leakages that occur in practice and hence is more reasonable. Moreover, unlike the notion of *after-the-fact leakage* which implicitly requires the split-state assumption, our approach is more general and also more natural to tackle with the randomness leakage.

Our Generic Construction. We present a generic framework for constructing AI-LR-eCK-secure AKE protocols to illustrate the feasibility of our proposed security model. Our generic construction utilizes several building blocks, including a *strong extractor with hard-to-invert auxiliary inputs* (denoted as AI-SE in the rest of paper), a *twisted pseudo-random function* (tPRF), a *smooth projective hash function* (SPHF) and a signature scheme that is existentially unforgeable under chosen message and auxiliary input attacks (EU-CMAA).

– *Underlying Primitives.* The notion of AI-SE was introduced by Yuen et al. [**?**]. A randomness extractor Ext is said to be a *strong extractor with hard-to-invert auxiliary inputs* if no PPT adversary can distinguish $(r, f(x), \mathsf{Ext}(x, r))$ from $(r, f(x), u)$, where r, u are chosen uniformly at random and f is any computationally hard-to-invert function. It has been shown in Yuen et

al. [**?**] that such a strong extractor can be constructed based on the modified Goldreich-Levin theorem of Dodis et al. [**?**].

The notion of *twisted pseudo-random function* (tPRF) was introduced and used in the work [**?**,**?**] for constructing AKE protocols in the traditional setting. A tPRF $\widetilde{F}$ is pseudo-random in terms of two different forms. Firstly, for any polynomial q, no PPT adversary can distinguish $(x_1, \cdots, x_q, \widetilde{F}(K, x_1), \cdots, \widetilde{F}(K, x_q))$ from $(x_1, \cdots, x_q, R_1, \cdots, R_q)$ where K, $\{x_i, R_i\}_{i=1}^{q}$, are randomly chosen. Secondly, no PPT adversary can distinguish $(K, \widetilde{F}(K, x))$ from (K, R) for any randomly chosen K, x, R. An instantiation of tPRF from normal pseudo-random function was given in [**?**], which shows that pseudo-random functions *do imply* tPRFs.

The definition of a *smooth projective hash function* (SPHF) [**?**] requires the existence of a domain $\mathcal{X}$ and an underlying $\mathcal{NP}$ language $\mathcal{L}$, where elements of $\mathcal{L}$ form a subset of $\mathcal{X}$, i.e., $\mathcal{L} \subset \mathcal{X}$. The key property of SPHF is that the hash value of any word $W \in \mathcal{L}$ can be computed by using either a secret hashing key, or a public projection key with the witness to the fact that $W \in \mathcal{L}$. However, the projection key gives almost no information about the hash value of any point in $\mathcal{X} \setminus \mathcal{L}$. In our paper, we adopt an extension of the SPHF by introducing a new algorithm WordG for the word generation and require the SPHF to be based on a hard-on-the-average $\mathcal{NP}$-language and hence is *pseudo-random* [**?**]. That is, given a word $W \in \mathcal{L}$, without the corresponding witness or the secret hashing key, the distribution of its hash value is computationally indistinguishable from an uniform distribution.

– *Roadmap of Our Framework.* Our framework is based on the eCK-secure AKE protocols proposed in the work [**?**,**?**], which are in the traditional (i.e., non-leakage) setting. Our 2-round AI-LR-eCK-secure AKE protocol works as follows. First, for both the initiator and the responder, we apply the AI-SE on the long-term secret key lsk and ephemeral secret key esk to extract two new secrets, i.e., $\widetilde{lsk}, \widetilde{esk}$. According to the definition of AI-SE, we know that $\widetilde{lsk}, \widetilde{esk}$ are indistinguishable from random values from the view of the adversary even in the presence of computationally hard-to-invert leakages on lsk and esk.

We then use $\widetilde{lsk}$ and $\widetilde{esk}$ to generate a session-unique secret hashing key and a witness for the WordG algorithm for the SPHF. However, since in the AI-LR-eCK model the adversary can reveal either the long-term secret key or the ephemeral secret key, if we directly apply the SPHF, e.g., by setting one of $\widetilde{lsk}$ and $\widetilde{esk}$ as the secret hashing key, then the adversary can reveal lsk or esk to obtain the secret hashing key (as AI-SE guarantees nothing in this situation!). Due to a similar reason, we cannot directly use $\widetilde{lsk}$ or $\widetilde{esk}$ as the witness in word generation. To tackle this problem, our construction generates a session-specific secret hashing key and a witness for the WordG algorithm by applying a tPRF $\widetilde{F}(\widetilde{lsk}, \widetilde{esk})$. More precisely, the initiator derives the two random values (r_1, r_2) from the tPRF. It first applies r_1 to generate the hashing key and the projection key hp. It then signs hp using r_2 and sends hp with the corresponding signature to the responder. Upon receiving the message from the initiator, the responder verifies the signature and then executes the same procedure to generate two random values (w, r_3) by applying the tPRF. It then use w as the witness to generate the word W and signs W with other communicated messages using r_3. Finally, it returns W and the corresponding signature to the initiator. The initiator and the responder can then compute the hash value of W, which is the shared session key, since the initiator knows the secret hashing key while the responder has the projection key and the witness.

It is worth noting that in our framework, we require the underlying signature scheme to be *existentially unforgeability under chosen message and auxiliary input attacks* (EU-CMAA) [**?**]. However, as illustrated by Faust et al. [**?**], no EU-CMAA-seucre signature scheme w.r.t. polynomially hard-to-invert leakage exists. The reason is that the signing algorithm can be regarded as a polynomially hard-to-invert leakage function. To address this problem, the work by Faust et al. [**?**] considered a restricted class of leakage functions that are exponentially hard-to-invert. *Fortunately, we do not need to enforce such a restriction on the leakage functions in this work*. It is due to the fact that in our general framework of AKE construction, for authentication, the initiator ($\mathcal{A}$) generates the signature on the transcript $(\mathsf{hp}, \widehat{A}, lpk_{\mathcal{B}}, \widehat{B})$ where $lpk_{\mathcal{B}}$ is the long-term public key of the responder ($\mathcal{B}$) and likewise the responder signs the transcript $(W_{\mathcal{B}}, \widehat{B}, \mathsf{hp}, \widehat{A})$ that contains the ephemeral public key hp of the initiator. Therefore, the adversary cannot forge a signature on the challenge session through leakage query, as it is asked to specify the allowed leakage function sets prior to the game setup in the $\mathsf{AI\text{-}LR\text{-}eCK}$ model and hence neither $lpk_{\mathcal{B}}$ nor hp can be embedded into the leakage function by the adversary.

An Instantiation of Our Framework. We show that the building blocks in our framework can be instantiated in the standard model. To be precise, we first describe a construction of $\mathsf{AI\text{-}SE}$ introduced in [**?**] using inner product from the Goldreich-Levin theorem [**?**] over any field $GF(q)$ for a prime q. A simple instantiation of tPRF from PRFs is then presented. We introduce the Diffie-Hellman language $\mathcal{L}_{\mathsf{DH}} = \{(u_1, u_2) | \exists r \in \mathbb{Z}_p, s.t., u_1 = g_1^r, u_2 = g_2^r\}$ where $\mathbb{G}$ is a group of primer order p and $g_1, g_2 \in \mathbb{G}$ are generators. An SPHF, denoted by $\mathcal{SPHF}_{\mathsf{DH}}$, is then constructed based on $\mathcal{L}_{\mathsf{DH}}$. We show that the EU-CMAA-secure signature scheme of Faust et al. [**?**] (henceforth called the FHNNZ signature scheme) can be used as the underlying signature scheme for instantiating our framework.

2 Preliminaries

In this section, we introduce some notations and definitions used in our construction.

2.1 Notations

In this paper, for a finite set Ω, $\omega \xleftarrow{\$} \Omega$ denotes that ω is selected uniformly at random from Ω.

Computational Indistinguishability. Let $\mathcal{V}_1$ and $\mathcal{V}_2$ be two probability distributions over a finite set Ω where $|\Omega| \geq 2^k$ and k is a security parameter. We then define a distinguisher $\widetilde{\mathcal{D}}$ as follows. In the game, $\widetilde{\mathcal{D}}$ takes as input $\mathcal{V}_1$ and $\mathcal{V}_2$, the challenger flips a coin $\gamma \xleftarrow{\$} \{0, 1\}$. $\widetilde{\mathcal{D}}$ is then given an element $v_1 \xleftarrow{\$} \mathcal{V}_1$ if $\gamma = 1$, otherwise an element $v_2 \xleftarrow{\$} \mathcal{V}_2$. Finally, $\widetilde{\mathcal{D}}$ outputs a bit $\gamma' \in \{0, 1\}$ as its guess on γ. We define the advantage of $\widetilde{\mathcal{D}}$ in this game as $\mathsf{Adv}_{\widetilde{\mathcal{D}}}^{\mathcal{V}_1, \mathcal{V}_2}(k) = \Pr[\gamma' = \gamma] - 1/2$. We say that $\mathcal{V}_1$ and $\mathcal{V}_2$ are *computationally indistinguishable* if for any probabilistic polynomial-time (PPT) distinguisher $\widetilde{\mathcal{D}}$, $\mathsf{Adv}_{\widetilde{\mathcal{D}}}^{\mathcal{V}_1, \mathcal{V}_2}(k)$ is negligible, and we denote it by $\mathcal{V}_1 \stackrel{c}{\equiv} \mathcal{V}_2$.

2.2 Strong Extractor with Hard-to-Invert Auxiliary Inputs

A central component of our construction is a new variant of strong randomness extractor which is proposed in [**?**]. The new notion, named strong extractor with ϵ-hard-to-invert auxiliary inputs, is defined as follows.

Definition 1 (Strong Extractor with ϵ-Hard-to-Invert Auxiliary Inputs [?]). *Let* $\mathsf{Ext} : \{0,1\}^{l_1} \times \{0,1\}^{l_2} \to \{0,1\}^{m'}$*, where* l_1, l_2 *and* m' *are polynomial in the security parameter* k*.* Ext *is said to be a strong extractor with* ϵ*-hard-to-invert auxiliary inputs, if for all pairs* (x, f) *such that* $x \in \{0,1\}^{l_2}$ *and* $f \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$*, we have:*

$$\{(r, f(x), \mathsf{Ext}(x, r))\} \stackrel{c}{\equiv} \{(r, f(x), u)\}$$

where $r \in \{0,1\}^{l_1}, u \in \{0,1\}^{m'}$ *are chosen uniformly at random.*

It is worth noting that the above leakage function $f \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$ can be viewed as a composition of q functions $f_1, f_2, \cdots, f_q$, where $q \in \mathbb{N}^+$ and $f_i \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$. Details are provided in Section 3.1. The following Lemma is obtained in [**?**].

Lemma 1 [?]. *Let* $r \in \{0,1\}^{l_1}$ *be chosen uniformly at random. For any pair* (x, f) *where* $x \in \{0,1\}^{l_2}$ *and* $f \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$*, no PPT adversary can recover* x *with probability* $\geq \epsilon$ *given* $(r, f, \mathsf{Ext}(x, r))$*, provided that* $\mathsf{Ext}(x, r)$ *is a strong extractor with* ϵ*-hard-to-invert auxiliary inputs.*

2.3 Pseudo-Random Function

In this paper, we adopt two specific types of pseudo-random function called $\pi\mathsf{PRF}$ and *twisted* pseudo-random function (tPRF) in our construction. $\pi\mathsf{PRF}$ was firstly defined in [**?**] .Twisted PRF was introduced in [**?**] and later improved in [**?**]. Here we briefly recall the notion of $\pi\mathsf{PRF}$ and introduce an enhanced version of the improved twisted PRF for our work.

Let $k \in \mathbb{N}$ be a security parameter. A function family F is associated with $\{\mathsf{Seed}_k\}_{k\in\mathbb{N}}$, $\{\mathsf{Dom}_k\}_{k\in\mathbb{N}}$ and $\{\mathsf{Rng}_k\}_{k\in\mathbb{N}}$. Formally, for any $\sum \stackrel{\$}{\leftarrow} \mathsf{Seed}_k$, $\sigma \stackrel{\$}{\leftarrow} \sum$, $\mathcal{D} \stackrel{\$}{\leftarrow} \mathsf{Dom}_k$ and $\mathcal{R} \stackrel{\$}{\leftarrow} \mathsf{Rng}_k$, $\mathsf{F}_\sigma^{k,\sum,\mathcal{D},\mathcal{R}}$ defines a function which maps an element of $\mathcal{D}$ to an element of $\mathcal{R}$. That is, $\mathsf{F}_\sigma^{k,\sum,\mathcal{D},\mathcal{R}}(\rho) \in \mathcal{R}$ for any $\rho \in \mathcal{D}$.

Definition 2 (PRF). *We say that* F *is a pseudo-random function* (PRF) *family if*

$$\{\mathsf{F}_\sigma^{k,\sum,\mathcal{D},\mathcal{R}}(\rho_i)\} \stackrel{c}{\equiv} \{\mathsf{RF}(\rho_i)\}$$

for any $\rho_i \in \mathcal{D}$ *adaptively chosen by any polynomial time distinguisher, where* RF *is a truly random function. That is, for any* $\rho \in \mathcal{D}, \mathsf{RF}(\rho) \stackrel{\$}{\leftarrow} \mathcal{R}$*.*

$\pi\mathsf{PRF}$. Roughly speaking, $\pi\mathsf{PRF}$ refers to a pseudo-random function family that if a specific key σ is pairwise-independent from other keys, then the output of function with key σ is computationally indistinguishable from a random element.

Formally, let $Z_\sum$ be a set of random variables over $\sum$, and $I_\sum$ be a set of indices regarding $\sum$ such that there exits a deterministic polynomial-time algorithm, $f_\sum : I_\sum \to Z_\sum$, which on input the index $i \in I_\sum$, output $\sigma_i \in Z_\sum$. Consider the random variables $\{\sigma_{i_j}\}_{j=0,\ldots,q(k)} = \{f_\sum(i_j)\}_{j=0,\ldots,q(k)}$ where $i_j \in I_\sum$ and $q(k)$ a polynomial function of k. We say that σ_{i_0} is *pairwisely independent* from other variables $\sigma_{i_1}, ..., \sigma_{i_{q(k)}}$ if for any pair of $(\sigma_{i_0}, \sigma_{i_j})(j = 1, ..., q(k))$, for any $(x, y) \in \sum^2$, we have $\Pr[\sigma_{i_0} \to x \wedge \sigma_{i_j} \to y] = 1/|\sum|^2$.

Definition 3 ($\pi\mathsf{PRF}$). Define $\widetilde{\mathsf{F}}(\rho_j) = \mathsf{F}_{\sigma_{i_j}}^{k,\sum,\mathcal{D},\mathcal{R}}(\rho_j)$ for $i_j \in I_\sum$, $\rho_j \in \mathcal{D}$. We say that F is a $\pi\mathsf{PRF}$ family if

$$\{\widetilde{\mathsf{F}}(\rho_j)\} \stackrel{c}{\equiv} \{\widetilde{\mathsf{RF}}(\rho_j)\}$$

for any $\{i_j \in I_{\sum}, \rho_j \in \mathcal{D}\}$ $(j = 0, 1, ..., q(k))$ adaptively chosen by any polynomial time distinguisher such that σ_{i_0} is pairwisely independent from $\sigma_{i_j} (j > 0)$, where $\widetilde{\mathsf{RF}}$ is the same as $\widetilde{\mathsf{F}}$ except that $\widetilde{\mathsf{RF}}(\rho_0)$ is replace by a truly random value in $\mathcal{R}$.

Definition 4 (Twisted PRF). *We say that a function $\widetilde{F} : \{0,1\}^{t_1} \times \{0,1\}^{t_2} \to \{0,1\}^{t_2}$ is a twisted pseudo-random function* (tPRF) *if*

- *For any polynomial $q(k)$,*

$$\{(x_1, \cdots, x_{q(k)}, \widetilde{F}(K, x_1), \cdots, \widetilde{F}(K, x_{q(k)}))\} \stackrel{c}{\equiv} \{(x_1, \cdots, x_{q(k)}, R_1, \cdots, R_{q(k)})\}$$

  where $K \in \{0,1\}^{t_1}, x_i \in \{0,1\}^{t_2}, R_i \in \{0,1\}^{t_2}$ are randomly chosen for any $i = 1, ..., q(k)$, and
- *For any randomly chosen K, x, R,*

$$\{(K, \widetilde{F}(K, x))\} \stackrel{c}{\equiv} \{(K, R)\}$$

An Enhanced Variant of Twisted PRF. In this work, we consider an enhanced variant of tPRF. That is, similar to the distinguisher defined in the security game of PRF, we allow the distinguisher to adaptively choose the input in the security game of tPRF. Precisely, the two properties of tPRF defined above still hold when $(x_1, \cdots, x_{q(k)})$ and K are chosen by any polynomial time distinguisher respectively.

An Instantiation of (Enhanced) tPRF from PRF. In [**?**], the authors showed that a twisted pseudo-random function $\widetilde{F} : \{0,1\}^{t_1} \times \{0,1\}^{t_2} \to \{0,1\}^{t_2}$ can be instantiated from a pseudo-random function $F : \{0,1\}^{t'_1} \times \{0,1\}^{t'_1} \to \{0,1\}^{t_2}$ as follows,

$$\widetilde{F}((k, k'), (a, a')) = F_k(a) \oplus F_{a'}(k')$$

where $t_1 = t_2 = 2t'_1$ and k, k', a, a' are randomly chosen from $\{0,1\}^{t'_1}$. A formal proof was given in [**?**] to show that $\widetilde{F}$ is a tPRF if F is a PRF. Therefore, we can see that PRFs *do imply* tPRFs. One can easily note that such an instantiation is also an enhanced tPRF.

2.4 Smooth Projective Hash Function

Smooth projective hash function(SPHF) is originally introduced by Cramer and Shoup [**?**] and extended for constructions of many cryptographic primitives [**?**,**?**,**?**,**?**,**?**]. We start with the original definition.

Syntax. Roughly speaking, the definition of an SPHF requires the existence of a domain $\mathcal{X}$ and an underlying $\mathcal{NP}$ language $\mathcal{L}$, where elements of $\mathcal{L}$ form a subset $\mathcal{X}$, i.e., $\mathcal{L} \subset \mathcal{X}$. A key property of SPHF is that, for any point W in the language $\mathcal{L}$ ($W \in \mathcal{L}$), the hash value of W can be computed by using either a secret hashing key which also works for the computation of any point in the set $\mathcal{X} \setminus \mathcal{L}$, or a public projection key which only works for any point $W \in \mathcal{L}$ and requires the knowledge of the witness w for the fact that $W \in \mathcal{L}$. Formally, an SPHF over a language $\mathcal{L} \subset \mathcal{X}$, onto a set $\mathcal{Y}$, is defined by the following algorithms

- $\mathsf{SPHFSetup}(1^k)$: generates the global parameters param and the description of an $\mathcal{NP}$ language $\mathcal{L}$ from the security parameter k;

- $\mathsf{HashKG}(\mathcal{L}, \mathsf{param})$: generates a hashing key hk for the language $\mathcal{L}$;
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}))$: derives the projection key hp from the hashing key hk;
- $\mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W)$: outputs the hash value $\mathsf{hv} \in \mathcal{Y}$ on the word W from the hashing key hk;
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathcal{L}, \mathsf{param}), W, w)$: outputs the hash value $\mathsf{hv}' \in \mathcal{Y}$, on the word W from the projection key hp, and the witness w for the fact that $W \in \mathcal{L}$.

Extension. In order to make the SPHF notion well applied for our work, similar to [**?**], we also need an extension of the SPHF in this paper. Precisely, we introduce the WordG algorithm and slightly modify the $\mathsf{Hash}, \mathsf{ProjHash}$ algorithms for SPHF as follows.[1]

- $\mathsf{WordG}(\mathcal{L}, \mathsf{param}, w)$: generates a word $W \in \mathcal{L}$ with w the witness ;
- $\mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W, aux)$: outputs the hash value $\mathsf{hv} \in \mathcal{Y}$ on the word W from the hashing key hk and the auxiliary input aux;
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathcal{L}, \mathsf{param}), W, w, aux)$: outputs the hash value $\mathsf{hv}' \in \mathcal{Y}$, on the word W from the projection key hp, the witness w for the fact that $W \in \mathcal{L}$ and the auxiliary input aux.

Property. A smooth projective hash function $\mathcal{SPHF}$=($\mathsf{SPHFSetup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{WordG}, \mathsf{Hash}, \mathsf{ProjHash}$) should satisfy the following properties,

- *Correctness*. Let $W = \mathsf{WordG}(\mathcal{L}, \mathsf{param}, w)$, then for all hashing key hk and projection key hp , we have

$$\mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W, aux) = \mathsf{ProjHash}(\mathsf{hp}, (\mathcal{L}, \mathsf{param}), W, w, aux)$$

- *Smoothness*. For any $W \in \mathcal{X} \backslash \mathcal{L}$. Then the following two distributions are perfectly indistinguishable:

$$\mathcal{V}_1 = \{(\mathcal{L}, \mathsf{param}, W, \mathsf{hp}, aux, \mathsf{hv}) | \mathsf{hv} = \mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W, aux)\},$$

$$\mathcal{V}_2 = \{(\mathcal{L}, \mathsf{param}, W, \mathsf{hp}, aux, \mathsf{hv}) | \mathsf{hv} \xleftarrow{\$} \mathcal{Y}\}.$$

To summary, a smooth projective hash function has the property that the projection key uniquely determines the hash value of any word in the language $\mathcal{L}$ but gives almost no information about the hash value of any point in $\mathcal{X} \setminus \mathcal{L}$.

Definition 5 (2-smooth **SPHF).** *For any $W_1, W_2 \in \mathcal{X} \backslash \mathcal{L}$, let aux_1, aux_2 be the auxiliary inputs such that $(W_1, aux_1) \neq (W_2, aux_2)$, we say an SPHF is 2-*$\mathsf{smooth}$ *if the following two distributions are statistically indistinguishable :*

$$\mathcal{V}_1 = \{(\mathcal{L}, \mathsf{param}, W_1, W_2, \mathsf{hp}, aux_1, aux_2, \mathsf{hv}_1, \mathsf{hv}_2) | \mathsf{hv}_2 = \mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W_2, aux_2)\},$$

$$\mathcal{V}_2 = \{(\mathcal{L}, \mathsf{param}, W_1, W_2, \mathsf{hp}, aux_1, aux_2, \mathsf{hv}_1, \mathsf{hv}_2) | \mathsf{hv}_2 \xleftarrow{\$} \mathcal{Y}\}.$$

where $\mathsf{hv}_1 = \mathsf{Hash}(\mathsf{hk}, (\mathcal{L}, \mathsf{param}), W_1, aux_1)$.

[1] In the rest of paper, all the SPHFs are referred to as the extended SPHF and defined by algorithms ($\mathsf{SPHFSetup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{WordG}, \mathsf{Hash}, \mathsf{ProjHash}$).

Definition 6 (Hard Subset Membership Problem). *For a finite set $\mathcal{X}$ and an NP language $\mathcal{L} \subset \mathcal{X}$, we say the subset membership problem is hard if for any word $W \xleftarrow{\$} \mathcal{L}$, W is computationally indistinguishable from any random element chosen from $\mathcal{X}\backslash\mathcal{L}$. Formally, we have that*

$$\mathsf{Adv}^{\mathsf{SMP}}_{\mathcal{A}}(k) = \Pr\left[b' = b : \begin{array}{l}(\mathcal{L}, \mathsf{param}) \xleftarrow{\$} \mathsf{SPHFSetup}(1^k);\\ b \xleftarrow{\$} \{0,1\}; (W_0, w) \xleftarrow{\$} \mathsf{SampYes}(\mathcal{L}, \mathsf{param});\\ W_1 \xleftarrow{\$} \mathsf{SampNo}(\mathcal{L}, \mathsf{param});\\ b' \leftarrow \mathcal{A}(\mathcal{L}, \mathsf{param}, W_b)\end{array}\right] - \frac{1}{2} \leq \mathsf{negl(k)}.$$

Here $\mathsf{SampYes}(\mathcal{L}, \mathsf{param})$ *is a polynomial time algorithm for sampling an element* (W, w) *from* $\mathcal{L}$ *where* w *is the witness to the membership* $W \in \mathcal{L}$ *and* $\mathsf{SampNo}(\mathcal{L}, \mathsf{param})$ *is for sampling an element* W *from* $\mathcal{X}\backslash\mathcal{L}$.

2.5 Signature Schemes Secure Against Hard-To-Invert Leakage

A signature scheme consists of a tuple of PPT algorithm $(\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ defined as follows.

- $\mathsf{KeyGen}(1^k)$: generates the signing and verification key $(\mathsf{sk}, \mathsf{vk})$;
- $\mathsf{Sign}(\mathsf{sk}, m)$: outputs the signature σ of a message m;
- $\mathsf{Verify}(\mathsf{vk}, m, \sigma)$: outputs 1 if the signature is accepted and 0 otherwise.

A signature scheme should satisfy the following correctness property: for any message m and keys $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^k)$, $\Pr[\mathsf{Verify}(\mathsf{vk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1$. The standard security notion for a signature scheme is existentially unforgeability under chosen message attacks (EU-CMA). In this paper, we are interested in the following notion for signature schemes proposed in [**?**], namely *existentially unforgeability under chosen message and auxiliary input attacks* (EU-CMAA).

Definition 7 (EU-CMAA) [?]. *A signature scheme* $\mathcal{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ *is secure against EU-CMAA w.r.t.* $\mathcal{H}$ *which is a class of admissible leakage functions, if for any PPT adversary* $\mathcal{M}$ *and any function* $h \in \mathcal{H}$,

$$\mathsf{Adv}^{\mathsf{EU\text{-}CMAA}}_{\mathcal{SIG},\mathcal{H},\mathcal{M}}(k) = \Pr\left[\begin{array}{c}(sk, vk) \leftarrow \mathsf{KeyGen}(1^k);\\ (m^*, \sigma^*) \leftarrow \mathcal{M}^{\mathcal{O}(sk,\cdot)}(vk, h(vk, sk)) :\\ \mathsf{Verify}(vk, m^*, \sigma^*) = 1.\end{array}\right]$$

is negligible in k, *where oracle* $\mathcal{O}(sk, \cdot)$ *outputs* $\mathsf{Sign}(sk, m)$ *for any input message* m.

2.6 The Extended Canetti-Krawczyk (eCK) Model for AKE

In this subsection, we review the eCK security model for AKE protocols.

AKE Protocols. An AKE protocol is run among parties $(\mathcal{A}, \mathcal{B}, \mathcal{C}, ...)$ which are modelled as probabilistic polynomial-time Turing Machines. Each party has a *long-term secret key* (lsk) together with a certificate that binds the corresponding *long-term public key* (lpk) to the party. Here we denote $\widehat{A}(\widehat{B})$ as the long-term public key of party $\mathcal{A}$ ($\mathcal{B}$) with the certificate issued by the certificate authority $\mathcal{CA}$.

Any two parties, say $\mathcal{A}, \mathcal{B}$, can be activated to run an instance of the AKE protocol, namely *session* to obtain a shared session key. During the session execution, the party that first sends a message is called an *initiator* and the party that first receives a message is called a *responder*. Suppose that $\mathcal{A}$ is the session initiator. When the session is activated, $\mathcal{A}$ generates the *ephemeral public/secret key* pair $(epk_{\mathcal{A}}, esk_{\mathcal{A}})$ and sends $(\widehat{B}, \widehat{A}, epk_{\mathcal{A}})$ to session responder $\mathcal{B}$. $\mathcal{B}$ then creates the ephemeral public/secret key pair $(epk_{\mathcal{B}}, esk_{\mathcal{B}})$ and sends $(\widehat{A}, \widehat{B}, epk_{\mathcal{A}}, epk_{\mathcal{B}})$ to $\mathcal{A}$. At the end of the session execution, each party derives the shared session key by taking as input his/her own long-term secret key and ephemeral secret key, along with the long-term public key and ephemeral public key received from the other party.

The session of initiator $\mathcal{A}$ with responder $\mathcal{B}$ is identified by the session identifier $(\widehat{A}, \widehat{B}, epk_{\mathcal{A}}, epk_{\mathcal{B}})$, where $\mathcal{A}$ is the owner of the session and $\mathcal{B}$ the peer of the session. The session of responder $\mathcal{B}$ with initiator $\mathcal{A}$ is identified as $(\widehat{B}, \widehat{A}, epk_{\mathcal{B}}, epk_{\mathcal{A}})$, where $\mathcal{B}$ is the owner of the session and $\mathcal{A}$ the peer of the session. Session $(\widehat{B}, \widehat{A}, epk_{\mathcal{B}}, epk_{\mathcal{A}})$ is said to be the *matching session* of $(\widehat{A}, \widehat{B}, epk_{\mathcal{A}}, epk_{\mathcal{B}})$. If the party outputs a session key at the end of the session, we called the session is completed.

eCK Security Model. The extended Canetti-Krawczyk (eCK) model was proposed by LaMacchia, Lauter and Mityagin [**?**] based on the CK model which was formulated by Canetti and Krawczyk [**?**] for the AKE protocols. Roughly speaking, in the eCK definition, the adversary $\mathcal{M}$ is modelled as a probabilistic polynomial time Turing machine that controls all communications between the honest parties. In the eCK model, adversary $\mathcal{M}$ is allowed to issue the following oracle queries.

- $\mathsf{Send}(\mathcal{A}, \mathcal{B}, message)$. Sends $message$ to party $\mathcal{A}$ on behalf of party $\mathcal{B}$. Returns $\mathcal{A}$'s response to this message. This query allows $\mathcal{M}$ to activate $\mathcal{A}$ to start a session and present party $\mathcal{B}$ to communicate with $\mathcal{A}$.
- $\mathsf{EstablishParty}(\mathsf{pid})$. This query allows the adversary to register a long-term public key on behalf of party pid, which is said to be *dishonest*. Note that if a party is not dishonest, we call the party *honest*.
- $\mathsf{LongTermKeyReveal}(\mathsf{pid})$. This query allows the adversary to learn the long-term secret key of party pid.
- $\mathsf{SessionKeyReveal}(\mathsf{sid})$. This query allows the adversary to obtain the session key of the completed session sid.
- $\mathsf{EphemeralKeyReveal}(\mathsf{sid})$. This query allows the adversary to obtain the ephe-meral secret key of session sid.

Eventually, adversary $\mathcal{M}$ selects a completed session sid^* as the *test session* and makes a query $\mathsf{Test}(\mathsf{sid}^*)$ as follows.

- $\mathsf{Test}(\mathsf{sid}^*)$. To answer this query, the challenger pick $b \xleftarrow{\$} \{0,1\}$. If $b = 1$, the challenger returns $SK^* \leftarrow \mathsf{SessionKeyReveal}(\mathsf{sid}^*)$. Otherwise, the challenger sends $\mathcal{M}$ a random key $R^* \xleftarrow{\$} \{0,1\}^{|SK^*|}$.

Note that the Test query can be issued only once and the game terminates as soon as $\mathcal{M}$ outputs its guess b' on b. Here, we require the *test session* to be a *fresh session* which is defined as follows.

Definition 6 (**Fresh Session in eCK Model**). *Let* sid *be the completed session owned by an honest party $\mathcal{A}$ with peer $\mathcal{B}$, who is also honest. If there exists the matching session to session*

sid*, we denote the matching session as* $\overline{\mathsf{sid}}$*. Session* sid *is said to be fresh if none of the following conditions hold:*

- $\mathcal{M}$ *issues a* $\mathsf{SessionKeyReveal}(\mathsf{sid})$ *query or a* $\mathsf{SessionKeyReveal}(\overline{\mathsf{sid}})$ *query (If* $\overline{\mathsf{sid}}$ *exists).*
- $\overline{\mathsf{sid}}$ *exists and* $\mathcal{M}$ *issues either*
  - $\mathsf{LongTermKeyReveal}(\mathcal{A}) \wedge \mathsf{EphemeralKeyReveal}(\mathsf{sid})$*, or*
  - $\mathsf{LongTermKeyReveal}(\mathcal{B}) \wedge \mathsf{EphemeralKeyReveal}(\overline{\mathsf{sid}})$.
- $\overline{\mathsf{sid}}$ *does not exist and* $\mathcal{M}$ *issues either*
  - $\mathsf{LongTermKeyReveal}(\mathcal{A}) \wedge \mathsf{EphemeralKeyReveal}(\mathsf{sid})$*, or*
  - $\mathsf{LongTermKeyReveal}(\mathcal{B})$.

We remark that the freshness of the test session can be identified only after the game is completed as $\mathcal{M}$ can continue the other queries after the Test query. That is, $\mathcal{M}$ wins the game if he correctly guesses the challenge for the test session which remains fresh until the end of the game. Formally, we have the following notion for eCK security.

Definition 8 (**eCK Security**). *Let the test session* sid^* *be fresh where adversary* $\mathcal{M}$ *issues* $\mathsf{Test}(sid^*)$ *query. We define the advantage of* $\mathcal{M}$ *in the eCK game by*

$$\mathsf{Adv}^{\mathsf{eCK}}_{\mathcal{M}}(k) = \Pr[b' = b] - 1/2,$$

where k *is the security parameter of the AKE protocol. We say the AKE protocol is eCK-secure if the matching session computes the same session key and for any probabilistic polynomial-time adversary* $\mathcal{M}$*,* $\mathsf{Adv}^{\mathsf{eCK}}_{\mathcal{M}}$ *is negligible.*

3 Formal Security Against Auxiliary Inputs for AKE

In this section, we introduce a new security model, namely leakage-resilient eCK model w.r.t. auxiliary input (AI-LR-eCK), for AKE protocols.

3.1 Admissible Auxiliary Input Functions

To model both the long-term secret key and ephemeral secret key leakage with respect to the auxiliary input, we firstly specify the set of admissible functions $\mathcal{H}$. Following the work of Dodis et al. [**?**], we define two classes of auxiliary input leakage functions.

- Let $\mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{lsk}})$ be the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{lsk}|+|\mathsf{lpk}|} \rightarrow \{0,1\}^*$, such that given $h(\mathsf{lsk}, \mathsf{lpk})$ (for a randomly generated long-term key pair $(\mathsf{lsk}, \mathsf{lpk})$), no PPT adversary can find lsk with probability greater than ϵ_{lsk}. The function $h(\mathsf{lsk})$ can be viewed as a composition of $q_{\mathsf{lsk}} \in \mathbb{N}^+$ functions, i.e., $h(\mathsf{lsk}, \mathsf{lpk}) = (h_1(\mathsf{lsk}, \mathsf{lpk}), \cdots, h_{q_{\mathsf{lsk}}}(\mathsf{lsk}, \mathsf{lpk}))$ where for all $i \in \{1, \cdots, q_{\mathsf{lsk}}\}$, $h_i \in \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{lsk}})$.
- Let $\mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ be the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{lsk}|+|\mathsf{lpk}|} \rightarrow \{0,1\}^*$, such that given $(\mathsf{lpk}, h(\mathsf{lsk}, \mathsf{lpk}))$ (for a randomly generated long-term key pair $(\mathsf{lsk}, \mathsf{lpk})$), no PPT adversary can find lsk with probability greater than ϵ_{lsk}. The function $h(\mathsf{lsk}, \mathsf{lpk})$ can be viewed as a composition of $q_{\mathsf{lsk}} \in \mathbb{N}^+$ functions, i.e., $h(\mathsf{lsk}, \mathsf{lpk}) = (h_1(\mathsf{lsk}, \mathsf{lpk}), \cdots, h_{q_{\mathsf{lsk}}}(\mathsf{lsk}, \mathsf{lpk}))$ where for all $i \in \{1, \cdots, q_{\mathsf{lsk}}\}, h_i \in \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$.

For the auxiliary input functions with respect to ephemeral secret key leakage, we also define two classes.

- Let $\mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{esk}})$ be the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{esk}|+|\mathsf{epk}|} \rightarrow \{0,1\}^*$, such that given $h(\mathsf{esk}, \mathsf{epk})$ (for a randomly generated ephemeral key pair $(\mathsf{esk}, \mathsf{epk})$), no PPT adversary can find esk with probability greater than ϵ_{esk}. The function $h(\mathsf{esk}, \mathsf{epk})$ can be viewed as a composition of $q_{\mathsf{esk}} \in \mathbb{N}^+$ functions, i.e., $h(\mathsf{esk}, \mathsf{epk}) = (h_1(\mathsf{esk}, \mathsf{epk}), \cdots, h_{q_{\mathsf{esk}}}(\mathsf{esk}, \mathsf{epk}))$ where for all $i \in \{1, \cdots, q_{\mathsf{esk}}\}$, $h_i \in \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{esk}})$.
- Let $\mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$ be the class of all the polynomial-time computable functions $h : \{0,1\}^{|\mathsf{esk}|+|\mathsf{epk}|} \rightarrow \{0,1\}^*$, such that given $(\mathsf{epk}, h(\mathsf{esk}, \mathsf{epk}))$ (for a randomly generated ephemeral key pair $(\mathsf{esk}, \mathsf{epk})$), no PPT adversary can find esk with probability greater than ϵ_{esk}. The function $h(\mathsf{esk}, \mathsf{epk})$ can be viewed as a composition of $q_{\mathsf{esk}} \in \mathbb{N}^+$ functions, i.e., $h(\mathsf{esk}) = (h_1(\mathsf{esk}, \mathsf{epk}), \cdots, h_{q_{\mathsf{esk}}}(\mathsf{esk}, \mathsf{epk}))$ where for all $i \in \{1, \cdots, q_{\mathsf{esk}}\}$, $h_i \in \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$.

One can note that if either $\epsilon_{\mathsf{lsk}} \leq 2^{-|\mathsf{lsk}|}$ or $\epsilon_{\mathsf{esk}} \leq 2^{-|\mathsf{esk}|}$, our definitions would be trivialized since then no leakage if admissible. In the definition of $\mathcal{H}_{\mathsf{ow}}$, we require that it is hard to compute the secret key given only the leakage. In contrast, when defining $\mathcal{H}_{\mathsf{lpk\text{-}ow}}$ and $\mathcal{H}_{\mathsf{epk\text{-}ow}}$, we ask that the secret key is hard to compute when given the public key in additional to the leakage, which means the allowed leakage functions *depend on the information of the public key* while the leakage class $\mathcal{H}_{\mathsf{ow}}$ is defined *independently of the concrete AKE scheme*.

In this work, we typically define the auxiliary input model with respect to the class $\mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ and $\mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$. This is because that AKE protocols normally belong to the public key setting in terms of the authentication mechanism. As stated in [**?**], it would be impossible to prove that the scheme remains secure w.r.t the class of admissible auxiliary input function $\mathcal{H}_{\mathsf{ow}}$ since the public key (e.g., lpk) itself leaks some information about the secret key (lsk). The stronger security notion for the class of admissible auxiliary input function $\mathcal{H}_{\mathsf{ow}}$, can be then achieved according to the following lemma which is firstly proven by Dodis et al.[**?**] and later formalized in [**?**].

Lemma 2. ([?,?]) *For any ϵ_{lsk} and ϵ_{esk}, we have that*

1. $\mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}}) \subseteq \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{lsk}})$, $\mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}}) \subseteq \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{esk}})$.
2. $\mathcal{H}_{\mathsf{ow}}(2^{-|\mathsf{lpk}|} \cdot \epsilon_{\mathsf{lsk}}) \subseteq \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$, $\mathcal{H}_{\mathsf{ow}}(2^{-|\mathsf{epk}|} \cdot \epsilon_{\mathsf{esk}}) \subseteq \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{esk}})$.

It is worth noting that the public-key setting is not always the case for the secrets that are not involved in the authentication mechanism (e.g., the ephemeral secret). A counterexample is our generic construction that will be introduced below. Nevertheless, we consider such class of auxiliary input function to make our model general. Actually, for the ephemeral secret that has no public key (i.e., $|\mathsf{epk}| = 0$), we have that $\mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}}) = \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{esk}})$ according to **Lemma 2**.

3.2 Auxiliary Input Model for AKE

We are now ready to present the new security model, i.e., leakage-resilient eCK model w.r.t. auxiliary input (AI-LR-eCK) for AKE protocols. Roughly speaking, we allow the adversary to access the leakage oracle in addition to the queries specified by the original eCK model. Formally, we define the following two leakage queries for the adversary in the AI-LR-eCK model.

- LongTermKeyLeakage$(f_{\mathsf{lsk}}, \mathsf{pid})$. This allows the adversary to query the leakage oracle and learn $f_{\mathsf{lsk}}(lsk, lpk)$ where $f_{\mathsf{lsk}} \in \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ and (lsk, lpk) denotes the long-term key pair of party pid.

– $\mathsf{EphemeralKeyLeakage}(f_{\mathsf{esk}}, \mathsf{sid})$. This allows the adversary to query the leakage oracle and learn $f_{\mathsf{esk}}(esk, epk)$ where $f_{\mathsf{esk}} \in \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$ and (esk, epk) denotes the ephemeral key pair used by the owner of the session sid.

One should note that separating the leakage queries for the long-term key and the ephemeral key is reasonable in practice, as these keys are usually not stored in the same place (e.g., the long-term key can be stored in ROM, while ephemeral keys are stored in RAM).

Note that in our new security model, we aim to remove the restriction in the previous models (excluding the after-the-fact-leakage AKE model proposed in [**?**,**?**] by Alawatugoda et al.). That is, the adversary can access to the leakage oracle before, during or after the test session. However, as shown below, if there is no other restriction with respect to the leakage query, a trivial attack can be launched by the adversary as follows.

A Trivial Attack. Consider a test session sid^* which is owned by party $\mathcal{A}$ with peer $\mathcal{B}$. Note that for a 2-pass AKE protocol, the session key of sid^* is determined by $(\widehat{A}, \widehat{B}, lsk_{\mathcal{A}}, esk^*_{\mathcal{A}}, lpk_{\mathcal{B}}, epk^*_{\mathcal{B}})$ which contains only two secret keys (i.e., $lsk_{\mathcal{A}}, esk^*_{\mathcal{A}}$). Since $\mathcal{M}$ is allowed to reveal $esk^*_{\mathcal{A}}$ ($lsk_{\mathcal{A}}$) in the eCK model, $\mathcal{M}$ can launch a trivial attack by encoding the session key derivation function into the leakage function of $lsk_{\mathcal{A}}$ ($esk^*_{\mathcal{A}}$) and hence wins the security game. Therefore, some special treatments should be adopted otherwise the security cannot be achieved at all.

Our Treatment. As we have discussed before, in reality, leakage attacks are *often determined by the hardware devices and measurement methods in physical implementations*. On the other hand, the trivial attack above assumes that the adversary can adaptively choose any abstract leakage function, which is *overly strong for capturing leakage attacks in reality* [**?**,**?**]. Therefore, it is reasonable to change the full adaptive definition which would give a more meaningful way to address the trivial attack. Inspired by this observation, in our proposed model, we ask the adversary to submit two leakage function sets $\mathcal{F}_{\mathsf{lsk}} \subseteq \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ and $\mathcal{F}_{\mathsf{esk}} \subseteq \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}})$, where both $|\mathcal{F}_{\mathsf{lsk}}|$ and $|\mathcal{F}_{\mathsf{esk}}|$ are polynomial in the security parameter, prior to the game setup. During the security game, the adversary is only allowed to *adaptively* ask for any leakage functions belonging to the committed sets. More precisely, for all the queries $\mathsf{LongTermKeyLeakage}(f_{\mathsf{lsk}}, \mathsf{pid})$ and $\mathsf{EphemeralKeyLeakage}(f_{\mathsf{esk}}, \mathsf{sid})$ that are issued by the adversary, we require that $f_{\mathsf{lsk}} \in \mathcal{F}_{\mathsf{lsk}}, f_{\mathsf{esk}} \in \mathcal{F}_{\mathsf{esk}}$. It is worth noting that the above natural relaxation of leakage resilience has also been considered in e.g., [**?**,**?**,**?**,**?**] and is generally known as *non-adaptive leakage resilience* where the adversary has to fix the leakage functions in advance before seeing the system parameters.

Readers may consider a slightly stronger model where the adversary is only required to submit $\mathcal{F}_{\mathsf{lsk}}, \mathcal{F}_{\mathsf{esk}}$ before the challenge phase (i.e., the test session). However, if we adopt this approach, then we cannot avoid the following trivial attack. Consider a test session sid^* where the session key is determined by $(\widehat{A}, \widehat{B}, lsk_{\mathcal{A}}, esk^*_{\mathcal{A}}, lpk_{\mathcal{B}}, epk^*_{\mathcal{B}})$. Suppose that the matching session $\overline{\mathsf{sid}^*}$ does not exist, i.e., adversary $\mathcal{M}$ controls the generation of $epk^*_{\mathcal{B}}$. To specify the function set $\mathcal{F}_{\mathsf{esk}}$, $\mathcal{M}$ can first obtain $lsk_{\mathcal{A}}$ via $\mathsf{LongTermKeyReveal}(\mathcal{A})$ and then encode the session key computation function and $(\widehat{A}, \widehat{B}, lsk_{\mathcal{A}}, lpk_{\mathcal{B}}, epk^*_{\mathcal{B}})$ into a leakage function f_{esk} that outputs the session key. Note that this can be done before the activation of test session since adversary $\mathcal{M}$ can choose $epk^*_{\mathcal{B}}$ by itself. The reason behind this trival attack is that the adversary can be actively involved in the challenge session in AKE.

We define the notion of a *fresh session* in the AI-LR-eCK model as follows.

Definition 9 (**(ϵ_1, ϵ_2)-Leakage Fresh Session in** AI-LR-eCK **Model**). *Let* sid *be a completed session owned by an honest party $\mathcal{A}$ with peer $\mathcal{B}$, who is also honest. Let* $\overline{\mathsf{sid}}$ *denote the matching session of* sid*, if it exists. Session* sid *is said to be fresh in the* AI-LR-eCK *model if the following conditions hold:*

- sid *is a fresh session in the sense of eCK model.*
- $\mathcal{M}$ *only issues the following leakage queries*
  - $\mathsf{LongTermKeyLeakage}(f_{\mathsf{lsk}}, \mathcal{A})$,
  - $\mathsf{LongTermKeyLeakage}(f_{\mathsf{lsk}}, \mathcal{B})$,
  - $\mathsf{EphemeralKeyLeakage}(f_{\mathsf{esk}}, \mathsf{sid})$,
  - $\mathsf{EphemeralKeyLeakage}(f_{\mathsf{esk}}, \overline{\mathsf{sid}})$ *(if* $\overline{\mathsf{sid}}$ *exists),*

  such that $f_{\mathsf{lsk}} \in \mathcal{F}_{\mathsf{lsk}} \subseteq \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_1)$, $f_{\mathsf{esk}} \in \mathcal{F}_{\mathsf{esk}} \subseteq \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_2)$ where both $\mathcal{F}_{\mathsf{lsk}}$ and $\mathcal{F}_{\mathsf{esk}}$ are submitted by $\mathcal{M}$ at the very beginning of the security game.

It remains to formally define the notion of AI-LR-eCK security.

Definition 10 (AI-LR-eCK **Security**). *Let the test session* sid^* *be (ϵ_1, ϵ_2)-leakage fresh where adversary $\mathcal{M}$ issues* $\mathsf{Test}(sid^*)$ *query. We define the advantage of $\mathcal{M}$ in the* AI-LR-eCK *game by*

$$\mathsf{Adv}^{\mathsf{AI\text{-}LR\text{-}eCK}}_{\mathcal{M}}(k) = \Pr[b' = b] - 1/2,$$

*where k is the security parameter of the AKE protocol. We say the AKE protocol is (ϵ_1, ϵ_2)-leakage-resilient eCK w.r.t. auxiliary inputs-secure ((ϵ_1, ϵ_2)-*AI-LR-eCK*-secure) if the matching session computes the same session key and for any probabilistic polynomial-time adversary $\mathcal{M}$,* $\mathsf{Adv}^{\mathsf{AI\text{-}LR\text{-}eCK}}_{\mathcal{M}}(k)$ *is negligible.*

Table 2. Classification of Valid Attacks by Adversary $\mathcal{M}$

Events	Owner of sid^*	Existence of $\overline{\mathsf{sid}}^*$	$lsk_{\mathcal{A}}$	$esk^*_{\mathcal{A}}$	$lsk_{\mathcal{B}}$	$esk^*_{\mathcal{B}}$
E_1	$\mathcal{A}/\mathcal{B}$	Yes	R	L	R	L
E_2	$\mathcal{A}/\mathcal{B}$	Yes	L	R	L	R
E_3	$\mathcal{A}/\mathcal{B}$	Yes	R	L	L	R
E_4	$\mathcal{A}/\mathcal{B}$	Yes	L	R	R	L
E_5	$\mathcal{A}$	No	R	L	L	$\otimes$
E_6	$\mathcal{A}$	No	L	R	L	$\otimes$
E_7	$\mathcal{B}$	No	L	$\otimes$	R	L
E_8	$\mathcal{B}$	No	L	$\otimes$	L	R

1. "R" means the long-term or ephemeral secret key is revealed and "L" means the long-term or ephemeral secret key is accessed through leakage query by the adversary. "$\otimes$" means that the ephemeral secret key is under the control of the adversary.

Classification of Valid Attacks in AI-LR-eCK**.** Here we discuss the relationship between the reveal oracle, e.g., LongTermKeyReveal and the leakage oracle, e.g., LongTermKeyLeakage. We can see that it is meaningless for $\mathcal{M}$ to issue the leakage query on the long-term secret key (ephemeral secret key) if it has already obtained the whole key through querying the reveal oracle. Indeed, adversary $\mathcal{M}$ can compute itself the leakage function $f_{\mathsf{lsk}}(lsk_{\mathcal{A}})$ if $lsk_{\mathcal{A}}$ is known to him. Therefore, the valid attack on the test session can be classified according to Table **??**.

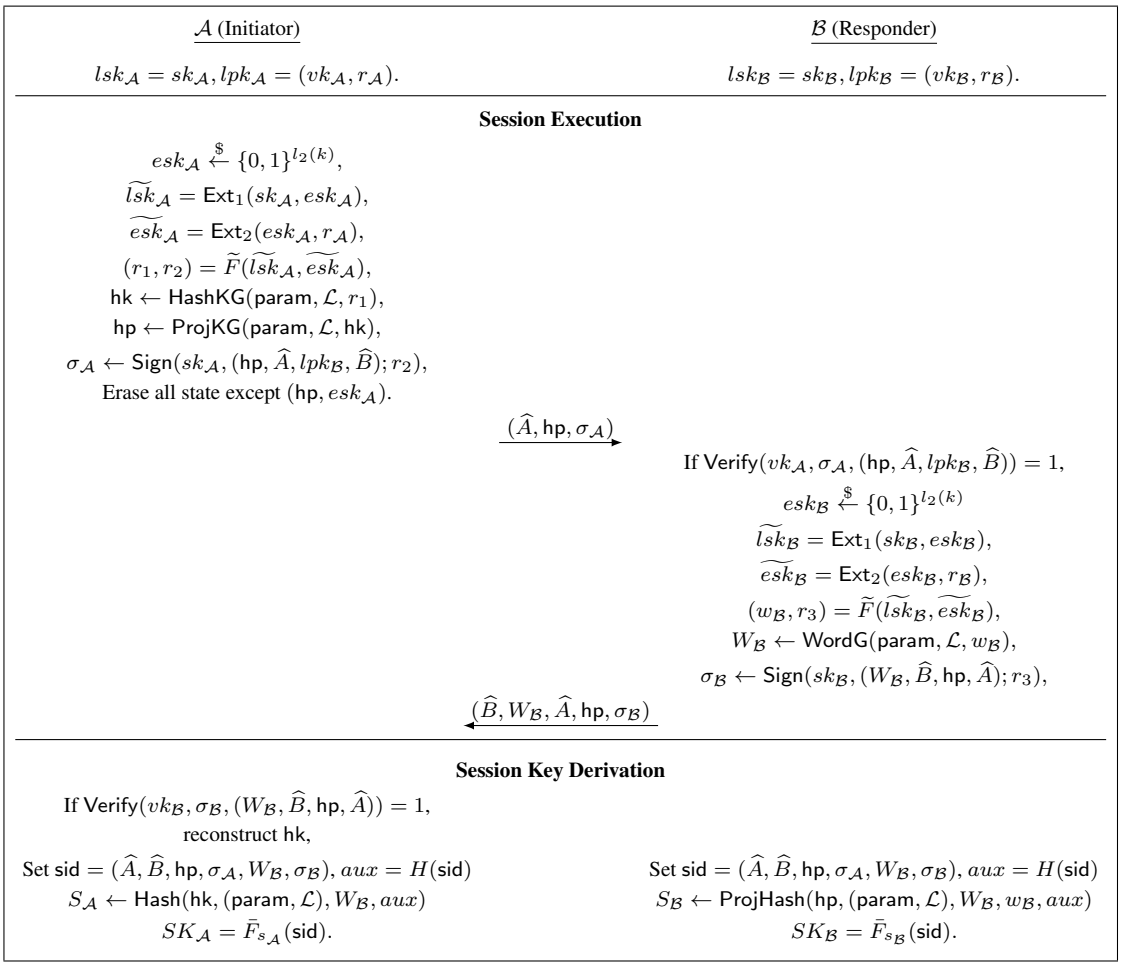

Fig. 1. Framework for AI-LR-eCK-Secure AKE.

Without loss of generality, we assume that the test session sid^* is a completed session owned by an honest party $\mathcal{A}$ with peer $\mathcal{B}$. Let $\overline{\mathsf{sid}}^*$ denote the matching session of sid^*, if it exists. We denote $esk^*_{\mathcal{A}}, esk^*_{\mathcal{B}}$ as the ephemeral secret keys used by $\mathcal{A}$ and $\mathcal{B}$ in sid^* and $\overline{\mathsf{sid}}^*$ (if it exists), respectively.

4 A General Framework for AI-LR-eCK-Secure AKE

In this section, we propose a general framework for constructing AI-LR-eCK-secure AKE protocols.

4.1 Our Generic Construction

Fig. **??** describes the generic construction for the secure AKE protocol with auxiliary inputs. We describe the framework as follows.

System Setup. Suppose that k is the system security parameter. Let $\mathcal{SPHF} = (\mathsf{SPHFSetup}, \mathsf{HashKG}, \mathsf{ProjKG}, \mathsf{WordG}, \mathsf{Hash}, \mathsf{ProjHash})$ denote an SPHF over $\mathcal{L} \subset \mathcal{X}$ and onto the set $\mathcal{Y}$. The assoicated auxiliary input space is denoted by $\mathcal{AUX}$. Let $\mathcal{SIG} = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$ be

an EU-CMAA-secure signature scheme with the signing key space denoted as $\mathcal{SK}$. Let $\mathsf{Ext}_1 : \mathcal{SK} \times \{0,1\}^{l_2(k)} \rightarrow \{0,1\}^{t_1(k)}, \mathsf{Ext}_2 : \{0,1\}^{l_2(k)} \times \{0,1\}^{l'_2(k)} \rightarrow \{0,1\}^{t_2(k)}$ be two strong extractors with hard-to-invert auxiliary inputs. Let $\widetilde{F} : \{0,1\}^{t_1(k)} \times \{0,1\}^{t_2(k)} \rightarrow \{0,1\}^{t_3(k)}$ be an enhanced tPRF and $\bar{F} : \mathcal{Y} \times \{0,1\}^* \rightarrow \{0,1\}^{l_3(k)}$ be a πPRF. Pick the collision-resistant hash function $H : \{0,1\}^* \rightarrow \mathcal{AUX}$. The system parameter is $(\mathsf{param}, \mathsf{Ext}_1, \mathsf{Ext}_2, \widetilde{F}, H)$ where $\mathsf{param} \leftarrow \mathsf{SPHFSetup}(1^k)$.

Key Generation. At the long-term key generation stage, $\mathcal{A}$ picks $r_{\mathcal{A}} \xleftarrow{\$} \{0,1\}^{l'_2(k)}$ and runs the algorithm KeyGen to obtain a signing/verification key pair $(sk_{\mathcal{A}}, vk_{\mathcal{A}})$, $\mathcal{A}$ then sets its long-term key pair as $lsk_{\mathcal{A}} = sk_{\mathcal{A}}, lpk_{\mathcal{A}} = (vk_{\mathcal{A}}, r_{\mathcal{A}})$. Similarly, $\mathcal{B}$ picks $r_{\mathcal{B}} \xleftarrow{\$} \{0,1\}^{l'_2(k)}$ and runs the algorithm KeyGen to obtain signing/verification key pair $(sk_{\mathcal{B}}, vk_{\mathcal{B}})$, $\mathcal{B}$ then sets its long-term key pair as $lsk_{\mathcal{B}} = sk_{\mathcal{B}}, lpk_{\mathcal{B}} = (vk_{\mathcal{B}}, r_{\mathcal{B}})$.

Session Execution $(\mathcal{A} \rightleftharpoons \mathcal{B})$**.** The key exchange protocol between $\mathcal{A}$ and $\mathcal{B}$ executes as follows.

- $(\mathcal{A} \rightharpoonup \mathcal{B})$. $\mathcal{A}$ performs the following steps.
  1. Selects the ephemeral secret key $esk_{\mathcal{A}} \xleftarrow{\$} \{0,1\}^{l_2(k)}$.
  2. Sets $\widetilde{lsk}_{\mathcal{A}} = \mathsf{Ext}_1(sk_{\mathcal{A}}, esk_{\mathcal{A}}), \widetilde{esk}_{\mathcal{A}} = \mathsf{Ext}_2(esk_{\mathcal{A}}, r_{\mathcal{A}})$.
  3. Computes $(r_1, r_2) = \widetilde{F}(\widetilde{lsk}_{\mathcal{A}}, \widetilde{esk}_{\mathcal{A}})$.
  4. Runs $\mathsf{HashKG}(\mathsf{param}, \mathcal{L}, r_1)$ to obtain the hashing key hk.
  5. Runs $\mathsf{ProjKG}(\mathsf{param}, \mathcal{L}, \mathsf{hk})$ to obtain the projection key hp.
  6. Signs hp by running $\mathsf{Sign}(sk_{\mathcal{A}}, \mathsf{hp}, \widehat{A}, lpk_{\mathcal{B}}, \widehat{B}); r_2)$ to obtain $\sigma_{\mathcal{A}}$.
  7. Erase all state except $(\mathsf{hp}, esk_{\mathcal{A}})$ and sends $(\widehat{A}, \mathsf{hp}, \sigma_{\mathcal{A}})$ to $\mathcal{B}$.
- $(\mathcal{B} \rightharpoonup \mathcal{A})$. Upon receiving the message from $\mathcal{A}$, $\mathcal{B}$ executes the following steps.
  1. Verifies $\sigma_{\mathcal{A}}$ and aborts if the verification fails, otherwise,
  2. Selects the ephemeral secret key $esk_{\mathcal{B}} \xleftarrow{\$} \{0,1\}^{l_2(k)}$.
  3. Sets $\widetilde{lsk}_{\mathcal{B}} = \mathsf{Ext}_1(sk_{\mathcal{B}}, esk_{\mathcal{B}}), \widetilde{esk}_{\mathcal{B}} = \mathsf{Ext}_2(esk_{\mathcal{B}}, r_{\mathcal{B}})$.
  4. Computes $(w_{\mathcal{B}}, r_3) = \widetilde{F}(\widetilde{lsk}_{\mathcal{B}}, \widetilde{esk}_{\mathcal{B}})$.
  5. Runs the algorithm $\mathsf{WordG}(\mathsf{param}, \mathcal{L}, w_{\mathcal{B}})$ to obtain a word $W_{\mathcal{B}}$.
  6. Signs $W_{\mathcal{B}}$ by running $\mathsf{Sign}(sk_{\mathcal{B}}, (W_{\mathcal{B}}, \widehat{B}, \mathsf{hp}, \widehat{A}), r_3)$ to obtain $\sigma_{\mathcal{B}}$.
  7. Sends $(\widehat{B}, W_{\mathcal{B}}, \widehat{A}, \mathsf{hp}, \sigma_{\mathcal{B}})$ to $\mathcal{A}$.

Session Key Derivation. When $\mathcal{A}$ receives $(\widehat{B}, W_{\mathcal{B}}, \widehat{A}, \mathsf{hp}, \sigma_{\mathcal{B}})$, $\mathcal{A}$ firstly verifies the signature $\sigma_{\mathcal{B}}$ by running $\mathsf{Verify}(vk_{\mathcal{B}}, \sigma_{\mathcal{B}}, (\widehat{B}, W_{\mathcal{B}}, \mathsf{hp}, \widehat{A}))$. If the verification fails, $\mathcal{A}$ aborts the protocol, otherwise, $\mathcal{A}$ reconstructs hk using $(sk_{\mathcal{A}}, esk_{\mathcal{A}}, r_{\mathcal{A}})$. Finally, $\mathcal{A}$ sets $\mathsf{sid} = (\widehat{A}, \widehat{B}, \mathsf{hp}, \sigma_{\mathcal{A}}, W_{\mathcal{B}}, \sigma_{\mathcal{B}})$, computes $aux = H(\mathsf{sid})$ and computes the session key as $K_{\mathcal{A}} \leftarrow \mathsf{Hash}(\mathsf{hk}, (\mathsf{param}, \mathcal{L}), W_{\mathcal{B}}, aux)$ and $SK_{\mathcal{A}} = \bar{F}_{s_{\mathcal{A}}}(\mathsf{sid})$. Similarly, party $\mathcal{B}$ computes the session key as $K_{\mathcal{B}} \leftarrow \mathsf{ProjHash}(\mathsf{hp}, (\mathsf{param}, \mathcal{L}), W_{\mathcal{B}}, w_{\mathcal{B}}, aux)$ and $SK_{\mathcal{B}} = \bar{F}_{s_{\mathcal{B}}}(\mathsf{sid})$.

Correctness. Due to the projective hash function, we can easily obtain that,

$$\mathsf{Hash}(\mathsf{hk}, (\mathsf{param}, \mathcal{L}), W_{\mathcal{B}}, aux) = \mathsf{ProjHash}(\mathsf{hp}, (\mathsf{param}, \mathcal{L}), W_{\mathcal{B}}, w_{\mathcal{B}}, aux)$$

which guarantees that $S_{\mathcal{A}} = S_{\mathcal{B}}$ and thus $SK_{\mathcal{A}} = SK_{\mathcal{B}}$.

4.2 Security Analysis

Theorem 1. *The above construction is* (ϵ_1, ϵ_2)-$\mathsf{AI\text{-}LR\text{-}eCK}$-*secure if the underlying* $\mathcal{SPHF}$ *is 2-smooth and the associated subset membership problem is hard,* $\mathcal{SIG}$ *is secure against EU-CMAA w.r.t.* $\mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_1)$, Ext_1 *is a strong extractor with* ϵ_1*-hard-to-invert auxiliary inputs, and* Ext_2 *is a strong extractor with* ϵ_2*-hard-to-invert auxiliary inputs, where both* ϵ_1 *and* ϵ_2 *are negligible.*

Proof. Denote the advantage of adversary $\mathcal{M}$ against our construction in the security model as $\mathsf{Adv}_{\mathcal{M}}(k)$.

Let the test session sid^* be as follows.

$$\mathsf{sid}^* = ((\widehat{A}, \mathsf{hp}^*, \sigma^*_{\mathcal{A}}), (\widehat{B}, W^*_{\mathcal{B}}, \widehat{A}, \mathsf{hp}^*, \sigma^*_{\mathcal{B}})) \text{ or } ((\widehat{B}, W^*_{\mathcal{B}}, \widehat{A}, \mathsf{hp}^*, \sigma^*_{\mathcal{B}}), (\widehat{A}, \mathsf{hp}^*, \sigma^*_{\mathcal{A}})).$$

We adopt the game-hopping technique for the security proof of our construction. We define a sequence of modified attack games, $\mathsf{Game}_0, \mathsf{Game}_1, \cdots$ between the simulator $\mathcal{S}$ and adversary $\mathcal{M}$. In each game, a random bit b is chosen by the simulator $\mathcal{S}$ for the test session and $\mathcal{M}$ outputs a bit b' at the end of game. We denote Succ_i as the event that $b = b'$ in Game_i. Therefore, the advantage of $\mathcal{M}$ in Game_i is $\mathsf{Adv}_{\mathcal{M},i} = \Pr[\mathsf{Succ}_i] - 1/2$.

It remains to show the details of each game.

Game 0. Game_0 is the original attack game. Suppose that adversary $\mathcal{M}$ activates at most $n(k)$ honest parties $\{P_1, \cdots, P_{n(k)}\}$ and activates party $\mathcal{A} \in \{P_1, \cdots, P_{n(k)}\}$ as an initiator at most $N(k)$ times. In the i-th activation, $\mathcal{S}$ chooses $esk_{\mathcal{A},i} \xleftarrow{\$} \{0,1\}^{l_2(k)}$, computes $\widetilde{lsk}_{\mathcal{A},i} = \mathsf{Ext}_1(sk_{\mathcal{A}}, esk_{\mathcal{A},i}), \widetilde{esk}_{\mathcal{A}_i} = \mathsf{Ext}_2\ (esk_{\mathcal{A},i}, r_{\mathcal{A}})$, and $(r_{1,i}, r_{2,i}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i}, \widetilde{esk}_{\mathcal{A},i})$. Suppose $\mathcal{M}$ activates party $\mathcal{B} \in \{P_1, \cdots, P_{n(k)}\}$ as a responder at most $N(k)$ times. In the j-th activation, $\mathcal{S}$ chooses $esk_{\mathcal{B},j} \xleftarrow{\$} \{0,1\}^{l_2(k)}$, computes $\widetilde{lsk}_{\mathcal{B},j} = \mathsf{Ext}_1(s_{\mathcal{B}}, esk_{\mathcal{B},j})$, $\widetilde{esk}_{\mathcal{B},j} = \mathsf{Ext}_2(esk_{\mathcal{B},j}, r_{\mathcal{B}})$, and $(w_{\mathcal{B},j}, r_{3,j}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{B},j}, \widetilde{esk}_{\mathcal{B},j})$. Similarly, for any activation of the other $n(k) - 2$ parties, $\mathcal{S}$ simulates the session correctly. One can easily that,

$$\mathsf{Adv}^{\mathsf{AI\text{-}LR\text{-}eCK}}_{\mathcal{M}}(k) = \mathsf{Adv}_{\mathcal{M},0} \tag{1}$$

Game 1. Game_1 is the same game as Game_0, except that at the beginning, $\mathcal{S}$ chooses two parties randomly from $[P_1, \cdots, P_{n(k)}]$ and aborts the game if they are not $\mathcal{A}$ and $\mathcal{B}$ respectively. We then have that,

$$\mathsf{Adv}_{\mathcal{M},1} = (1/n(k)^2) \cdot \mathsf{Adv}_{\mathcal{M},0} \tag{2}$$

Game 2. Game_2 is the same as Game_1 except that $\mathcal{S}$ aborts the game if $\mathcal{M}$ generates $\sigma^*_{\mathcal{A}}$ or $\sigma^*_{\mathcal{B}}$. Now we analyze the probability that $\mathcal{M}$ generates $\sigma^*_{\mathcal{A}}$ or $\sigma^*_{\mathcal{B}}$. On one hand, if adversary $\mathcal{M}$ obtains $lsk_{\mathcal{A}}$ ($lsk_{\mathcal{B}}$) through the reveal query, the case that $\mathcal{M}$ generates $\sigma^*_{\mathcal{A}}$ ($\sigma^*_{\mathcal{B}}$) would not happen as this implies that $\mathcal{M}$ has corrupted $\mathcal{A}$ ($\mathcal{B}$). This is not allowed due to the freshness requirement of sid^*. On the other hand, the probability that $\mathcal{M}$ forges either of the above signatures without obtaining the corresponding long-term secret key is bounded by the unforgeability of the underlying signature scheme $\mathcal{SIG}$. One may note that in our construciton, the random r_2 of signing in the test and other sessions is derived from the revealed key and leaked key while the one used by the singing oracle in the EU-CMAA model is uniformaly random. We insist that they are indistinguishable from the view of the adversary. This is due to the security property of the underlying randomness extractor and tPRF. Therefore, during the simulation, when the

random r_2 is replaced by a uniformly random number by the signing oracle defined in the EU-CMAA model, the returned signature is indistinguishable from the one in the real AI-LR-eCK game from the view of a computational adversary. Suppose that the advantage of any adversary against $\mathcal{SIG}$ in the EU-CMAA model is at most $\mathsf{Adv}^{\mathsf{EU\text{-}CMAA}}_{\mathcal{SIG},\mathcal{H}_{\mathsf{lpk\text{-}ow}}}(k)$, then we have that,

$$Adv_{\mathcal{M},2} = (1 - \mathsf{Adv}^{\mathsf{EU\text{-}CMAA}}_{\mathcal{SIG},\mathcal{H}_{\mathsf{lpk\text{-}ow}}}(k)) \cdot \mathsf{Adv}_{\mathcal{M},1} \tag{3}$$

Therefore, we say that $(\widehat{A}, \mathsf{hp}^*, \sigma^*_{\mathcal{A}})$ must be correctly computed by $\mathcal{S}$ in the i^*-th activation for some $i^* \in \{1, \cdots, N(k)\}$ and $(\widehat{B}, W^*_{\mathcal{B}}, \sigma^*_{\mathcal{B}})$ must be correctly computed by $\mathcal{S}$ in the j^*-th activation for some $j^* \in \{1, \cdots, N(k)\}$.

Game 3. Game_3 is the same as Game_2 except for the following. $\mathcal{S}$ chooses $i, j \in \{1, \cdots, N(k)\}$ randomly and aborts the game it does not hold that $i = i^*, j = j^*$. It is easily to obtain that,

$$Adv_{\mathcal{M},3} = (1/N(k)^2) \cdot \mathsf{Adv}_{\mathcal{M},2} \tag{4}$$

Game 4. Game_4 is the same as Game_3 except for the following. Suppose that the behaviour of $\mathcal{M}$ on the test session (and the matching session, if exits) belongs to the event $\mathsf{E}^* \in \{\mathsf{E}_1, \cdots, \mathsf{E}_8\}$ (Table 1). $\mathcal{S}$ then chooses an event $\mathsf{E}' \xleftarrow{\$} \{\mathsf{E}_1, \cdots, \mathsf{E}_8\}$ and aborts the game if $\mathsf{E}' \neq \mathsf{E}^*$. Therefore, we can see that,

$$Adv_{\mathcal{M},4} = 1/8 \cdot \mathsf{Adv}_{\mathcal{M},3} \tag{5}$$

Game 5. Game_5 is the same as Game_4 except for the following. For any activation of $\mathcal{A}$ as an initiator, $\mathcal{S}$ simulates it as follows.

CASE A.5.1. $\mathsf{E}^* \in \{\mathsf{E}_1, \mathsf{E}_3, \mathsf{E}_5\}$.

Instead of computing $\widetilde{esk}_{\mathcal{A},i^*} = \mathsf{Ext}_2(esk_{\mathcal{A},i^*}, r_{\mathcal{A}})$, $\mathcal{S}$ chooses $\widetilde{esk}_{\mathcal{A},i^*} \xleftarrow{\$} \{0,1\}^{t_2(k)}$. In this sub-case, we have the following claim.

CLAIM 1. *For any adversary $\mathcal{M}$, we have that*

$$|\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}| \leq 2\mathsf{AdvExt}_2(k),$$

where $\mathsf{AdvExt}_2(k)$ *is the most advantage of any adversary against* $\mathsf{Ext}_2 : \{0,1\}^{l_2(k)} \times \{0,1\}^{l'_2(k)} \rightarrow \{0,1\}^{t_2(k)}$ *which is a strong extractor with ϵ_2-hard-to-invert auxiliary inputs.*

Proof. We use simulator $\mathcal{S}$ as the adversary against the strong extractor with ϵ-hard-to-invert auxiliary inputs. More precisely, we assume a security test environment for Ext_2, where $\mathcal{S}$ is given $(r_{\mathcal{A}}, f_1(esk_{\mathcal{A},i^*}), \cdots, f_{q_e}(esk_{\mathcal{A},i^*}), T^*)$, of which T^* is either $T_0 = \mathsf{Ext}_2(esk_{\mathcal{A},i^*}, r_{\mathcal{A}})$ or $T_1 = r \xleftarrow{\$} \{0,1\}^{t_2(k)}$. During the simulation, $\mathcal{S}$ returns $(f_1(esk_{\mathcal{A},i^*}), \cdots, f_{q_e}(\widetilde{esk}_{\mathcal{A},i^*}))$ [2] as the leakage query outputs for $\mathcal{M}$. As for the test session, i.e., i^*-th session, $\mathcal{S}$ sets $\widetilde{esk}_{\mathcal{A},i^*}$ as T^*. One can note that when $T^* = T_0$, the simulation is equivalent to Game_4. Otherwise the simulation is equivalent to Game_5.

Finally, when $\mathcal{M}$ outputs its guess b', $\mathcal{S}$ outputs 1 if $b' = b$, otherwise outputs 0. Let $\mathsf{AdvExt}_2(k)$ be the advantage of $\mathcal{S}$ against Ext_2, then we have that,

$$\begin{aligned}\mathsf{AdvExt}_2(k) &= \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_0] - \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_1] \\ &= \Pr[b' = b | T^* = T_0] - \Pr[b' = b | T^* = T_1] \\ &= \frac{1}{2}(\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}).\end{aligned}$$

[2] Since the ephemeral secret key $esk_{\mathcal{A},i^*}$ has no corresponding public key, we have that $f_j \in \mathcal{H}_{\mathsf{epk\text{-}ow}}(\epsilon_{\mathsf{esk}}) = \mathcal{H}_{\mathsf{ow}}(\epsilon_{\mathsf{esk}})$ for all $1 < j < q_e$ according to **Lemma 2**.

and $|\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}| \leq 2\mathsf{AdvExt}_2(k)$. This completes the proof of CLAIM 1.

CASE A.5.2. $\mathsf{E}^* \in \{\mathsf{E}_2, \mathsf{E}_4, \mathsf{E}_6\}$[3].

Instead of computing $\widetilde{lsk}_{\mathcal{A},i} = \mathsf{Ext}_1(sk_{\mathcal{A}}, esk_{\mathcal{A},i})$, $\mathcal{S}$ chooses $\widetilde{lsk}_{\mathcal{A},i} \xleftarrow{\$} \{0,1\}^{t_1(k)}$ for all $i \in \{1, \cdots, N(k)\}$. In this sub-case, we have the following claim.

CLAIM 2. *For any adversary $\mathcal{M}$, we have that*

$$|\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}| \leq 2 \cdot N(k) \cdot \mathsf{AdvExt}_1(k),$$

where $\mathsf{AdvExt}_1(k)$ is the most advantage of any adversary against $\mathsf{Ext}_1 : \mathcal{SK} \times \{0,1\}^{l_2(k)} \to \{0,1\}^{t_1(k)}$ which is a strong extractor with ϵ_1-hard-to-invert auxiliary inputs.

Proof. We use simulator $\mathcal{S}$ as the adversary against the strong extractor with ϵ-hard-to-invert auxiliary inputs. More precisely, we assume $N(k)$ security games for Ext_1, where for each $i \in \{1, \cdots, N(k)\}$, $\mathcal{S}$ is given $(esk_{\mathcal{A},i}, f_1(sk_{\mathcal{A}}), \cdots, f_{q_l}(sk_{\mathcal{A}}), T_i^*)$, of which T_i^* is either $T_{0,i} = \mathsf{Ext}_1(sk_{\mathcal{A}}, esk_{\mathcal{A},i})$ or $T_{1,i} = r_i \xleftarrow{\$} \{0,1\}^{t_1(k)}$. During the simulation, $\mathcal{S}$ returns $(f_1(sk_{\mathcal{A}}), \cdots, f_{q_l}(sk_{\mathcal{A}}))$[4] as the leakage query outputs for $\mathcal{M}$. As for the i-th session, $\mathcal{S}$ sets $\widetilde{lsk}_{\mathcal{A},i}$ as T_i^*. One can note that if $T_i^* = T_{0,i}$ for all $i \in \{1, N(k)\}$, the simulation is equivalent to Game_4. Otherwise the simulation is equivalent to Game_5. Therefore, applying the hybrid argument and according to the analysis in CLAIM 1 we then have that,

$$|\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}| \leq 2 \cdot N(k) \cdot \mathsf{AdvExt}_1(k).$$

This completes the proof of CLAIM 2.

For any activation of $\mathcal{B}$ as a responder, similarly, $\mathcal{S}$ simulates as follows.

CASE B.5.1. $\mathsf{E}^* \in \{\mathsf{E}_1, \mathsf{E}_4, \mathsf{E}_7\}$.

$\mathcal{S}$ chooses $\widetilde{esk}_{\mathcal{B},j^*} \xleftarrow{\$} \{0,1\}^{t_2(k)}$, instead of from $\mathsf{Ext}_2(esk_{\mathcal{B},j^*}, r_{\mathcal{B}})$.

CASE B.5.2. $\mathsf{E}^* \in \{\mathsf{E}_2, \mathsf{E}_3, \mathsf{E}_8\}$.

Instead of computing $\widetilde{lsk}_{\mathcal{B},j} = \mathsf{Ext}_1(s_{\mathcal{B}}, esk_{\mathcal{B},j})$, $\mathcal{S}$ chooses $\widetilde{lsk}_{\mathcal{B},j} \xleftarrow{\$} \{0,1\}^{t_1(k)}$ for all $j \in \{1, \cdots, N(k)\}$.

One can note that CASE B.5.1 and CASE B.5.2 are similar to CASE A.5.1 and CASE A.5.2 respectively.

Therefore, for Game_5, we always have that,

$$|\mathsf{Adv}_{\mathcal{M},5} - \mathsf{Adv}_{\mathcal{M},4}| \leq 2 \cdot \max\{N(k) \cdot \mathsf{AdvExt}_1(k), \mathsf{AdvExt}_2(k)\} \tag{6}$$

Game 6. Game_6 is the same as Game_5 except for the following.

For any activation of $\mathcal{A}$ as an initiator, $\mathcal{S}$ simulates as follows.

CASE A.6.1. $\mathsf{E}^* \in \{\mathsf{E}_1, \mathsf{E}_3, \mathsf{E}_5\}$.

[3] One may notice that here $\mathcal{S}$ does not simulate the session of $\mathcal{A}$ when $\mathsf{E}^* \in \{\mathsf{E}_7, \mathsf{E}_8\}$. This is because that when E_7 or E_8 happens, the session of $\mathcal{A}$ is under the control of the adversary and thus it does not exist. It is also the case for the events E_5 and E_6 where $\mathcal{S}$ does not need to simulate the session of $\mathcal{B}$.

[4] Noting that $sk_{\mathcal{A}}$ here has the verification key $vk_{\mathcal{A}}$, one may wonder if the leakage query made by $\mathcal{M}$ can be answered by $\mathcal{S}$. It is actually the case, as for each leakage function $h_j \in \mathcal{H}_{\mathsf{lpk\text{-}ow}}(\epsilon_{\mathsf{lsk}})$ $(1 < j < q_l)$ by $\mathcal{M}$, we can set $f_j(sk_{\mathcal{A}}) = (h_j(sk_{\mathcal{A}}, vk_{\mathcal{A}}), vk_{\mathcal{A}}) \in \mathcal{H}_{ow}(\epsilon_{\mathsf{lsk}})$.

Instead of computing $(r_{1,i^*}, r_{2,i^*}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i^*}, \widetilde{esk}_{\mathcal{A},i^*})$, $\mathcal{S}$ chooses $(r_{1,i^*}, r_{2,i^*}) \xleftarrow{\$} \{0,1\}^{t_3(k)}$. In this sub-case, we have the following claim.

CLAIM 3. *For any adversary $\mathcal{M}$, we have that*

$$|\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}| \leq 2\mathsf{AdvtPRF}(k),$$

where $\mathsf{AdvtPRF}(k)$ *is the most advantage of any adversary against the enhanced twisted* PRF $\widetilde{F}$.

Proof. We use simulator $\mathcal{S}$ as the adversary against the second property of the enhanced twisted pseudo-random function. Precisely, we assume a security test environment for $\widetilde{F}$, where $\mathcal{S}$ specifies the input $\widetilde{lsk}_{\mathcal{A},i^*}$ and is given T^*. Here T^* is either $T_0 = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i^*}, \widetilde{esk}_{\mathcal{A},i^*})$ or $T_1 = r \xleftarrow{\$} \{0,1\}^{t_2(k)}$. During the simulation, as for the test session, i.e., i^*-th session, $\mathcal{S}$ sets (r_{1,i^*}, r_{2,i^*}) as T^*. One can note that when $T^* = T_0 = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i^*}, \widetilde{esk}_{\mathcal{A},i^*})$, although $\widetilde{esk}_{\mathcal{A},i^*}$ here is unknown to $\mathcal{S}$ and may be different from the one picked randomly by $\mathcal{S}$ in Game_5, they are statistically indistinguishable from the view of adversary $\mathcal{M}$. Therefore, when $T^* = T_0$, the simulation is equivalent to Game_5. Otherwise the simulation is equivalent to Game_6.

Finally, when $\mathcal{M}$ outputs its guess b', $\mathcal{S}$ outputs 1 if $b' = b$, otherwise outputs 0. Let $\mathsf{AdvtPRF}(k)$ be the advantage of $\mathcal{S}$ against $\widetilde{F}$, then we have that,

$$\begin{aligned}\mathsf{AdvtPRF}(k) &= \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_0] - \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_1] \\ &= \Pr[b' = b | T^* = T_0] - \Pr[b' = b | T^* = T_1] \\ &= \frac{1}{2}(\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}).\end{aligned}$$

and $|\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}| \leq 2\mathsf{AdvtPRF}(k)$. This completes the proof of CLAIM 3.

CASE A.6.2. $\mathsf{E}^* \in \{\mathsf{E}_2, \mathsf{E}_4, \mathsf{E}_6\}$.

Instead of computing $(r_{1,i}, r_{2,i}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i}, \widetilde{esk}_{\mathcal{A},i})$, $\mathcal{S}$ chooses $(r_{1,i}, r_{2,i}) \xleftarrow{\$} \{0,1\}^{t_3(k)}$ for all $i \in \{1, \cdots, N(k)\}$. In this sub-case, we have the following claim.

CLAIM 4. *For any adversary $\mathcal{M}$, we have that*

$$|\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}| \leq 2\mathsf{AdvtPRF}(k),$$

where $\mathsf{AdvtPRF}(k)$ *is the most advantage of any adversary against the enhanced twisted* PRF $\widetilde{F}$.

Proof. We use simulator $\mathcal{S}$ as the adversary against the first property of the enhanced twisted pseudo-random function. Precisely, we assume a security test environment for $\widetilde{F}$, where $\mathcal{S}$ specifies the input $(\widetilde{esk}_{\mathcal{A},1}, \cdots, \widetilde{esk}_{\mathcal{A},N(k)})$ and is given $(T_1^*, \cdots, T_{N(k)}^*)$. For all $i \in \{1, \cdots, N(k)\}$, T_i^* is either $T_{0,i} = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i}, \widetilde{esk}_{\mathcal{A},i})$ or $T_{1,i} = r \xleftarrow{\$} \{0,1\}^{t_2(k)}$. During the simulation, as for the i-th session, $\mathcal{S}$ sets $(r_{1,i}, r_{2,i})$ as T_i^*. One can note that when $T_i^* = T_{0,i} = \widetilde{F}(\widetilde{lsk}_{\mathcal{A},i}, \widetilde{esk}_{\mathcal{A},i})$ for all $i \in \{1, \cdots, N(k)\}$, although $\widetilde{lsk}_{\mathcal{A},i}$ here is unknown to $\mathcal{S}$ and may be different from the one picked randomly by $\mathcal{S}$ in Game_5, they are statistically indistinguishable from the view of adversary $\mathcal{M}$. Therefore, when $T_i^* = T_{0,i}$, the simulation is equivalent to Game_5. Otherwise the simulation is equivalent to Game_6.

Finally, when $\mathcal{M}$ outputs its guess b', $\mathcal{S}$ outputs 1 if $b' = b$, otherwise outputs 0. Let $\mathsf{AdvtPRF}(k)$ be the advantage of $\mathcal{S}$ against $\widetilde{F}$, then we have that,

$$\begin{aligned}\mathsf{AdvtPRF}(k) &= \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_0] - \Pr[\mathcal{S} \text{ outputs } 1 | T^* = T_1] \\ &= \Pr[b' = b | T^* = T_0] - \Pr[b' = b | T^* = T_1] \\ &= \frac{1}{2}(\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}).\end{aligned}$$

and $|\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}| \leq 2\mathsf{AdvtPRF}(k)$. This completes the proof of CLAIM 4.

For any activation of $\mathcal{B}$ as a responder, similarly, $\mathcal{S}$ simulates as follows.

CASE B.6.1. $\mathsf{E}^* \in \{\mathsf{E}_1, \mathsf{E}_4, \mathsf{E}_7\}$.

Instead of computing $(w_{\mathcal{B},j^*}, r_{3,j^*}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{B},j^*}, \widetilde{esk}_{\mathcal{B},j^*})$, $\mathcal{S}$ chooses $(w_{\mathcal{B},j^*}, r_{3,j^*}) \xleftarrow{\$} \{0,1\}^{t_3(k)}$.

CASE B.6.2. $\mathsf{E}^* \in \{\mathsf{E}_2, \mathsf{E}_3, \mathsf{E}_8\}$.

Instead of computing $(w_{\mathcal{B},j}, r_{3,j}) = \widetilde{F}(\widetilde{lsk}_{\mathcal{B},j}, \widetilde{esk}_{\mathcal{B},j})$, $\mathcal{S}$ chooses $(w_{\mathcal{B},j^*}, r_{3,j^*}) \xleftarrow{\$} \{0,1\}^{t_3(k)}$ for all $j \in \{1, \cdots, N(k)\}$.

One can note that CASE B.6.1 and CASE B.6.2 are similar to CASE A.6.1 and CASE A.6.2 respectively.

Therefore, for Game_6, we always have that,

$$|\mathsf{Adv}_{\mathcal{M},6} - \mathsf{Adv}_{\mathcal{M},5}| \leq 2 \cdot \mathsf{AdvtPRF}(k) \tag{7}$$

Game 7. Game_7 is the same as Game_6 except for the test session sid^*, $\mathcal{S}$ computes $SK^*_{\mathcal{B}}$ as $SK^*_{\mathcal{B}} \leftarrow \mathsf{Hash}(\mathsf{hk}^*, (\mathsf{param}, \mathcal{L}), W^*_{\mathcal{B}}, aux^*)$ instead of $SK^*_{\mathcal{B}} \leftarrow \mathsf{ProjHash}(\mathsf{hp}^*, (\mathsf{param}, \mathcal{L}), W^*_{\mathcal{B}}, w^*_{\mathcal{B}}, aux^*)$. We can see that Game 7 is identical to the Game 6 from the view of adversary $\mathcal{M}$ due to the fact that $SK^*_{\mathcal{B}} \leftarrow \mathsf{Hash}(\mathsf{hk}^*, \mathsf{param}, \mathcal{L}, W^*_{\mathcal{B}}, aux^*) = \mathsf{ProjHash}(\mathsf{hp}^*, \mathsf{param}, \mathcal{L}, W^*_{\mathcal{B}}, w^*_{\mathcal{B}}, aux^*)$. Therefore, we have

$$\mathsf{Adv}_{\mathcal{M},7} = \mathsf{Adv}_{\mathcal{M},6}. \tag{8}$$

Game 8. Game_8 is the same as Game_7 except that $\mathcal{S}$ choose $W^*_{\mathcal{B}} \in \mathcal{X} \setminus \mathcal{L}$ instead of deriving it from $\mathcal{L}$ through the algorithm WordG. We then have the following result.

CLAIM 5. *For any adversary $\mathcal{M}$, we have that*

$$|\mathsf{Adv}_{\mathcal{M},8} - \mathsf{Adv}_{\mathcal{M},7}| \leq 2\mathsf{AdvSPHF}_{\mathsf{SMP}}(k),$$

where $\mathsf{AdvSPHF}_{\mathsf{SMP}}(k)$ is the most advantage of any adversary against the hard subset membership probelm of $\mathcal{SPHF}$.

Proof. We use simulator $\mathcal{S}$ as the adversary against the hard subset membership problem of the underlying SPHF, i.e., $\mathcal{SPHF}$. More precisely, we assume a security test environment for $\mathcal{SPHF}$, where $\mathcal{S}$ is given $(\mathsf{param}, \mathcal{L}, W^*)$, of which W^* either belongs to $\mathcal{L}$ or belongs to $\mathcal{X} \setminus \mathcal{L}$. During the simulation, as for the test session, i.e., i^*-th session, $\mathcal{S}$ sets $W^*_{\mathcal{B}}$ in the test session as W^*. Therefore, when W^* belongs to $\mathcal{L}$, the simulation is equivalent to Game_7 as we can implicitly assume that $W^* = \mathsf{WordG}(\mathsf{param}, \mathcal{L}, w_{\mathcal{B},j^*})$. Although $w_{\mathcal{B},j^*}$ here may be different from that one picked by $\mathcal{S}$ in Game_7, they are statistically indistinguishable from the view of adversary $\mathcal{M}$. Otherwise if W^* belongs to $\mathcal{X} \setminus \mathcal{L}$, the simulation is equivalent to Game_8. Finally, when $\mathcal{M}$ outputs its guess b', $\mathcal{S}$ outputs 0 if $b' = b$, otherwise outputs 1. Let $\mathsf{AdvSPHF}_{\mathsf{SMP}}(k)$

be the advantage of $\mathcal{S}$ against the hard subset membership problem of $\mathcal{SPHF}$, then we have that,

$$|\mathsf{Adv}_{\mathcal{M},8} - \mathsf{Adv}_{\mathcal{M},7}| \leq 2\mathsf{AdvSPHF}_{\mathsf{SMP}}(k) \tag{9}$$

This completes the proof of CLAIM 5.

It is worth noting that adversary $\mathcal{M}$ may actives a session sid, which is not matching to session sid^*, with $\mathcal{A}$. Precisely, $\mathcal{M}$ can choose $W \in \mathcal{X} \setminus \mathcal{L}$ (e.g., by replaying $W^*_{\mathcal{B}}$ or its variant), sends W to $\mathcal{A}$ and issues $\mathsf{SessionKeyReveal}(\mathsf{sid})$ query to learn the shared key. According to the property of 2-smoothness of $\mathcal{SPHF}$, we have that $S^*_{\mathcal{A}}$ is pairwisely independent from any other such key and all public information. Therefore, the derived session key $SK^*_{\mathcal{A}}$ is computationally indistinguishable from a truly random element from $\mathcal{M}$'s view due to the application of πPRF. That is, we have that $\Pr[\mathsf{Succ}_8] = \frac{1}{2}$ and thus $\mathsf{Adv}_{\mathcal{M},8} = 0$.

According to (1),(2),(3),(4),(5),(6),(7),(8), (9), we have the conclusion that $\mathsf{Adv}^{\mathsf{AI\text{-}LR\text{-}eCK}}_{\mathcal{M}}(k)$ is negligible, which proves **Theorem 1**.

5 Instantiating the Building Blocks

In this section, we show an instantiation of our framework based on the strong extractor with hard-to-invert auxiliary inputs [**?**], a tPRF from PRFs, the *Decision Diffie-Hellman* (DDH) assumption and the FHNNZ signature scheme [**?**].

5.1 Strong Extractor with Hard-To-Invert Auxiliary Inputs

The work in [**?**] showed that a strong extractor with ϵ-hard-to-invert auxiliary inputs can be constructed from the modified Goldreich-Levin theorem [**?**]. We first describe the following Goldreich-Levin theorem [**?**] over any field $GF(q)$ for a prime q. Denote $< x, r >= \sum_{i=1}^{l} r_i x_i$ as the inner product of $x = (x_1, \cdots, x_l)$ and $r = (r_1, \cdots, r_l)$.

Theorem 2. *Let q be a prime, and let H be an arbitrary subset of $GF(q)$. Let $f : H^n \to \{0,1\}^*$ be any (possibly randomized) function. If there is a distinguisher $\mathcal{D}$ that runs in time t such that $|\Pr[\mathcal{D}(r, y, < s, r >) = 1] - \Pr[\mathcal{D}(r, y, u) = 1]| = \delta$ where $s \xleftarrow{\$} H^n, y \leftarrow f(s), r \xleftarrow{\$} GF(q)^n, u \xleftarrow{\$} GF(q)$, then there is an inverter $\mathcal{I}$ that runs in time $t' = t \cdot \mathsf{poly}(n, |H|, 1/\delta)$ such that $\Pr[\mathcal{I}(y) = s] \geq \frac{\delta^3}{512 \cdot n \cdot q^2}$.*

Below we show the instantiation described in [**?**]. Let $\mathsf{Ext}(x, r) =< x, r >$, we have the following theorem.

Theorem 3. *Let k be the security parameter. Let x be chosen uniformly random from $\{0,1\}^{l(k)}$ where $l(k) = \mathsf{poly}(k)$. Then given $f \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$, no PPT distinguisher $\mathcal{D}$ can distinguish $(r, f(x), \mathsf{Ext}(x, r))$ from $(r, f(x), u)$ with probability $\epsilon' \geq (512 \cdot l(k) q^2 \epsilon)^{1/3}$, where $r \xleftarrow{\$} GF(q)^{l(k)}$, $u \xleftarrow{\$} GF(q)$.*

The proof of above theorem is referred to [**?**].

5.2 A Simple Instantiation of tPRF

Following the work in [**?**], we show that the enhanced tPRF, i.e., $\widetilde{F} : \{0,1\}^{t_1(k)} \times \{0,1\}^{t_2(k)} \rightarrow \{0,1\}^{t_3(k)}$ can be simply instantiated as follows. Without loss of generality, we take the construction of $\widetilde{F}(\widetilde{lsk}_{\mathcal{A}}, \widetilde{esk}_{\mathcal{A}})$ as an example.

Let $\widetilde{lsk}_{\mathcal{A}} = \widetilde{lsk}'_{\mathcal{A}} || \widetilde{lsk}''_{\mathcal{A}}, \widetilde{esk}_{\mathcal{A}} = \widetilde{esk}'_{\mathcal{A}} || \widetilde{esk}''_{\mathcal{A}}$, where $|\widetilde{lsk}'_{\mathcal{A}}| = k_1, |\widetilde{lsk}''_{\mathcal{A}}| = k_2, |\widetilde{esk}'_{\mathcal{A}}| = k_3, |\widetilde{esk}''_{\mathcal{A}}| = k_4$ such that $k_1 + k_2 = t_1(k), k_3 + k_4 = t_2(k)$. Pick two PRFs $F' : \{0,1\}^{k_1} \times \{0,1\}^{k_3} \rightarrow \{0,1\}^{t_3(k)}, F'' : \{0,1\}^{k_4} \times \{0,1\}^{k_2} \rightarrow \{0,1\}^{t_3(k)}$. Then we construct $\widetilde{F}$ as follows.

$$\widetilde{F}(\widetilde{lsk}_{\mathcal{A}}, \widetilde{esk}_{\mathcal{A}}) = F'_{\widetilde{lsk}'_{\mathcal{A}}}(\widetilde{esk}'_{\mathcal{A}}) \oplus F''_{\widetilde{esk}''_{\mathcal{A}}}(\widetilde{lsk}''_{\mathcal{A}}).$$

According to the **Theorem 1** in [**?**], we then have the following theorem.

Theorem 4. *The above $\widetilde{F}$ is an enhanced* tPRF *as long as both F' and F'' are* PRF*s.*

5.3 DDH-based SPHF

We introduce the Diffie Hellman language $\mathcal{L}_{\mathsf{DH}}$ and show how to construct a 2-smooth SPHF on $\mathcal{L}_{\mathsf{DH}}$.

Diffie-Hellman Language. Let $\mathbb{G}$ be a group of primer order p and $g_1, g_2 \in \mathbb{G}$. The Diffie-Hellman Language is as follows.

$$\mathcal{L}_{\mathsf{DH}} = \{(u_1, u_2) | \exists r \in \mathbb{Z}_p, s.t., u_1 = g_1^r, u_2 = g_2^r\}$$

One can see that the witness space of $\mathcal{L}_{\mathsf{DH}}$ is $\mathcal{W} = \mathbb{Z}_p$ and $\mathcal{L}_{\mathsf{DH}} \subset \mathcal{X} = \mathbb{G}^2$. We have the following theorems.

Theorem 5. *The subset membership problem over $\mathcal{X} = \mathbb{G}^2$ and $\mathcal{L}_{\mathsf{DH}}$ is hard.*

Proof. One can easily obtain the theorem above from the DDH assumption and hence we omit the proof here. Actually, if an adversary can distinguish a word randomly picked from $\mathcal{L}_{\mathsf{DH}}$ from a random element chosen from $\mathcal{X} \backslash \mathcal{L}_{\mathsf{DH}}$, we can build a distinguisher for the DDH problem by using the adversary as a subroutine.

SPHF on $\mathcal{L}_{\mathsf{DH}}$. Here we show how to construct a 2-smooth SPHF (denoted by $\mathcal{SPHF}_{\mathsf{DH}}$) over the language $\mathcal{L}_{\mathsf{DH}} \subset \mathcal{X} = \mathbb{G}^2$ onto the group $\mathcal{Y} = \mathbb{G}$. Let $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$ denote a collision-resistant hash function. The concrete construction is as follows.

- $\mathsf{SPHFSetup}(1^\lambda)$: $\mathsf{param} = (\mathbb{G}, p, g_1, g_2)$;
- $\mathsf{HashKG}(\mathcal{L}_{\mathsf{DH}}, \mathsf{param})$: $\mathsf{hk} = (\alpha_1, \alpha_2, \beta_1, \beta_2) \xleftarrow{\$} \mathbb{Z}_p^4$;
- $\mathsf{ProjKG}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}))$: $\mathsf{hp} = (\mathsf{hp}_1, \mathsf{hp}_2) = (g_1^{\alpha_1} g_2^{\alpha_2}, g_1^{\beta_1} g_2^{\beta_2}) \in \mathbb{G}_p^2$;
- $\mathsf{WordG}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), w = r)$: $W = (g_1^r, g_2^r)$;
- $\mathsf{Hash}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W = (u_1, u_2) = (g_1^r, g_2^r), aux = d = H_1(W, aux'))$: $\mathsf{hv} = u_1^{\alpha_1 + d\beta_1} \cdot u_2^{\alpha_2 + d\beta_2}$;
- $\mathsf{ProjHash}(\mathsf{hp}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W = (u_1, u_2) = (g_1^r, g_2^r), w = r, aux = d = H_1(W, aux'))$: $\mathsf{hv}' = \mathsf{hp}_1^r \mathsf{hp}_2^{dr}$.

Note that $\mathcal{Y} = \mathbb{G}, \mathcal{HK} = \mathbb{Z}_p^4, \mathcal{HP} = \mathbb{G}_p^2, \mathcal{AUX} = \mathbb{Z}_p, \mathcal{W} = \mathbb{Z}_p$. Then we have the following theorem.

Theorem 6. *$\mathcal{SPHF}_{\mathsf{DH}}$ is a 2-*smooth *SPHF.*

Proof. We show that $\mathcal{SPHF}_{\mathsf{DH}}$ is projective and smooth (2-smooth).

– *Correctness*. With the above notations, for a word $W = (u_1, u_2) = (g_1^r, g_2^r)$ we have

$$\begin{aligned}\mathsf{Hash}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W, d) &= u_1^{\alpha_1 + d\beta_1} u_2^{\alpha_2 + d\beta_2} \\ &= \mathsf{hp}_1^r \mathsf{hp}_2^{dr} \\ &= \mathsf{ProjHash}(\mathsf{hp}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W, r, d).\end{aligned}$$

– *Smoothness* (2-smooth). Suppose $g_2 = g_1^\theta$. Note that $\mathsf{hp}_1 = g_1^{\alpha_1} g_2^{\alpha}, \mathsf{hp}_2 = g_1^{\beta_1} g_2^{\beta_2}$ which constraints $(\alpha_1, \alpha_2, \beta_1, \beta_2)$ to satisfy

$$\log_{g_1} \mathsf{hp}_1 = \alpha_1 + \theta\alpha_2. \tag{10}$$

$$\log_{g_1} \mathsf{hp}_2 = \beta_1 + \theta\beta_2. \tag{11}$$

Let $W_1 = (g_1^{r_1}, g_2^{r_2}), W_2 = (g_1^{r_1'}, g_2^{r_2'}) \in \mathcal{X} \backslash \mathcal{L}_{\mathsf{DH}}$ where $r_1 \neq r_2, r_1' \neq r_2'$, suppose $aux_1 = d_1 = H_1(W_1, aux_1'), aux_2 = d_2 = H_1(W_2, aux_2')$, then the hash value hv_1 of W_1, hv_2 of W_2 are as follows,

$$\mathsf{hv}_1 = \mathsf{Hash}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W_1, aux_1) = g_1^{r_1(\alpha_1 + d_1\beta_1)} g_2^{r_2(\alpha_2 + d_1\beta_2)},$$

$$\mathsf{hv}_2 = \mathsf{Hash}(\mathsf{hk}, (\mathcal{L}_{\mathsf{DH}}, \mathsf{param}), W_2, aux_2) = g_1^{r_1'(\alpha_1 + d_2\beta_1)} g_2^{r_2'(\alpha_2 + d_2\beta_2)},$$

which also constraint $(\alpha_1, \alpha_2, \beta_1, \beta_2)$ to satisfy

$$\log_{g_1} \mathsf{hv}_1 = r_1\alpha_1 + r_2\theta\alpha_2 + r_1 d_1\beta_1 + r_2 d_1\theta\beta_2. \tag{12}$$

$$\log_{g_1} \mathsf{hv}_2 = r_1'\alpha_1 + r_2'\theta\alpha_2 + r_1' d_2\beta_1 + r_2' d_2\theta\beta_2. \tag{13}$$

From the above equations, we have

$$(\alpha_1, \alpha_2, \beta_1, \beta_2) \cdot \mathbf{A} = (\log_{g_1} \mathsf{hp}_1, \log_{g_1} \mathsf{hp}_2, \log_{g_1} \mathsf{hv}_1, \log_{g_1} \mathsf{hv}_2),$$

where $\mathbf{A}$ is a matrix defined as

$$\mathbf{A} = \begin{bmatrix} 1 & \theta & 0 & 0 \\ 0 & 0 & 1 & \theta \\ r_1 & \theta r_2 & r_1 d_1 & \theta r_2 d_1 \\ r_1' & \theta r_2' & r_1' d_2 & \theta r_2' d_2 \end{bmatrix}.$$

Since $(W_1, aux_1) \neq (W_2, aux_2)$ where $aux_1 = d_1 = H_1(W_1, aux_1'), aux_2 = d_2 = H_1(W_2, aux_2')$, we have that $d_1 \neq d_2$. Furthermore, as $\theta \neq 0$, $r_1 \neq r_2$ and $r_1' \neq r_2'$, we can obtain that the determinant of $\mathbf{A}$ is $\theta^2 \cdot (r_2 - r_1) \cdot (r_2' - r_1') \cdot (d_2 - d_1) \neq 0$ and hence the equation (13) is independent of the equation (12). Therefore, we have that hv_2 is perfectly indistinguishable from any element randomly chosen from $\mathbb{G}$.

5.4 The FHNNZ Signature Scheme [?]

The FHNNZ signature scheme is introduced by Faust et al. and shown to be existential unforgeability under chosen message and auxiliary input attacks (EU-CMAA)-secure with respect to exponentially hard-to-invert leakage. Below we give a quick review of the scheme.

The Scheme Construction. Let H denote a family of second preimage resistant hash functions with key sampling algorithm Gen_H and input domain $\{0,1\}^{l_{\mathsf{in}}}$ where $l_{\mathsf{in}} = \mathrm{poly}(k)$. Let $\Gamma = (\mathsf{KeyGen}_E, \mathsf{Enc}, \mathsf{Dec})$ be an IND-WLCCA secure labelled public-key encryption scheme, and $\Pi = (\mathsf{CRSGen}, \mathsf{P}, \mathsf{V})$ a reusable-CRS NIZK system. Readers are referred to [**?**] for the details of these underlying tools.

- $\mathsf{KeyGen}(1^k)$: Sample $s \leftarrow \mathsf{Gen}_H(1^k)$ and $(\mathsf{ek}, \mathsf{dk}) \leftarrow \mathsf{KeyGen}_E(1^k)$. Furthermore, sample $(\mathsf{crs}, \mathsf{td_s}) \leftarrow S_1(1^k)$ and $x \leftarrow \{0,1\}^{l_{\mathsf{in}}}$, where $S = (S_1, S_2)$ is the simulator for Π. Compute $y = H_s(x)$. Set $(\mathsf{sk}, \mathsf{vk}) = (x, (y, s, \mathsf{ek}, \mathsf{crs}))$.
- $\mathsf{Sign}(m, \mathsf{sk})$: Compute $C = \mathsf{Enc}^m(\mathsf{ek}, x)$. Using crs and Π, generate a NIZK proof π proving that $\exists x$ such that $(C = \mathsf{Enc}^m(\mathsf{ek}, x) \wedge y = H_s(x))$. Output $\sigma = (C, \pi)$
- $\mathsf{Verify}(\mathsf{vk}, m, \sigma)$: Parse σ as C, π. Use crs and V to verify the NIZK proof π. Output 1 if the proof verifies correctly and 0 otherwise.

On the Leakage Functions. As shown in [**?**], the above construction is EU-CMAA-secure with respect to only exponentially hard-to-invert leakage. The central idea of their construction is to add an encryption $C = \mathsf{Enc}^m(\mathsf{ek}, x)$ of the signing key x to each signature. The encryption key ek is part of the verification key of the signature scheme but the corresponding decryption key dk is not part of the signing key. They then set up the signature scheme that dk can be guessed with probability p such that $\mathsf{negl}(k) \geq p \gg 2^{-k^c}$ for some negligible function $\mathsf{negl}(\cdot)$ and a constant $1 > c > 0$. Since any PPT algorithm taking as input a signature and the public key (excluding dk) can output the signing key x with probability p, the signing algorithm is then excluded from the admissible leakage function set $\mathcal{H}$ which is with hardness at least 2^{-k^c} (i.e., exponentially hard-to-invert).

Our Adoption of the FHNNZ Signature. At first glance, it seems that our construction cannot achieve the security under the AI-LR-eCK model if we just directly adopt the FHNNZ signature, as the admissible leakage function sets defined in AI-LR-eCK are computationally hard-to-invert and hence do not exclude the signing algorithm. However, this is actually not the case. That is, our construction is AI-LR-eCK-secure even we do not put such a restriction on the leakage functions when applying the FHNNZ signature in our AKE protocol. It is due to the fact that in our general framework of AKE construction, for authentication, the initiator ($\mathcal{A}$) generates the signature on the transcript including the long-term public key (i.e., $lpk_{\mathcal{B}}$) of the responder and likewise the responder ($\mathcal{B}$) signs on the transcript containing the ephemeral public key (i.e., hp) of the initiator. Therefore, the adversary cannot forge a signature on the challenge session through the leakage query, as it is asked to specify the allowed leakage function sets prior to the game setup in the AI-LR-eCK model and hence neither the aforementioned long-term public key nor the ephemeral public key can be embedded into the leakage function by the adversary.

6 Conclusion

In this work, we filled the research gap between the existing works and the reality for AKE with leakage resilience. Our proposed model, AI-LR-eCK, is a strong yet meaningful AKE security

notion that captures computationally hard-to-invert leakage attacks on both the long-term secret and the randomness (i.e., the ephemeral secret) under some reasonable restrictions. We also presented a general framework of AI-LR-eCK-secure AKE protocols with an instantiation to show that our new model is indeed achievable.

Acknowledgements. We would like to thank Dr. Zheng Yang for pointing out the subtle flaw in the security proof of the previous version and also his valuable comments on our fixing solution. We also want to thank Professor Colin Boyd for helpful discussions which improve this full version.

References

1. Entity authentication mechanisms-part3: Entity authentication using asymmetric techniques. ISO/IEC IS 9789-3, 1993.
2. Abdalla, M., Benhamouda, F., Blazy, O., Chevalier, C., Pointcheval, D.: Sphf-friendly non-interactive commitments. In: ASIACRYPT. pp. 214–234 (2013)
3. Akavia, A., Goldwasser, S., Vaikuntanathan, V.: Simultaneous hardcore bits and cryptography against memory attacks. In: TCC. pp. 474–495 (2009)
4. Alawatugoda, J., Boyd, C., Stebila, D.: Continuous after-the-fact leakage-resilient key exchange. In: ACISP. pp. 258–273 (2014)
5. Alawatugoda, J., Stebila, D., Boyd, C.: Modelling after-the-fact leakage for key exchange. In: ASIACCS. pp. 207–216 (2014)
6. Alwen, J., Dodis, Y., Wichs, D.: Leakage-resilient public-key cryptography in the bounded-retrieval model. In: CRYPTO. pp. 36–54 (2009)
7. Bellare, M., Canetti, R., Krawczyk, H.: A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In: ACM STOC. pp. 419–428 (1998)
8. Bellare, M., Rogaway, P.: Entity authentication and key distribution. In: CRYPTO. pp. 232–249 (1993)
9. Benhamouda, F., Blazy, O., Chevalier, C., Pointcheval, D., Vergnaud, D.: New techniques for sphfs and efficient one-round PAKE protocols. In: CRYPTO. pp. 449–475 (2013)
10. Biham, E., Shamir, A.: Differential fault analysis of secret key cryptosystems. In: CRYPTO. pp. 513–525 (1997)
11. Bitansky, N., Canetti, R., Halevi, S.: Leakage-tolerant interactive protocols. In: TCC. pp. 266–284 (2012)
12. Boyle, E., Segev, G., Wichs, D.: Fully leakage-resilient signatures. J. Cryptology 26(3), 513–558 (2013)
13. Canetti, R., Krawczyk, H.: Analysis of key-exchange protocols and their use for building secure channels. In: EUROCRYPT. pp. 453–474 (2001)
14. Choo, K.R., Boyd, C., Hitchcock, Y.: Examining indistinguishability-based proof models for key establishment protocols. In: ASIACRYPT. pp. 585–604 (2005)
15. Chow, S.S.M., Dodis, Y., Rouselakis, Y., Waters, B.: Practical leakage-resilient identity-based encryption from simple assumptions. In: ACM CCS. pp. 152–161 (2010)
16. Cramer, R., Shoup, V.: Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In: EUROCRYPT. pp. 45–64 (2002)
17. Dodis, Y., Goldwasser, S., Kalai, Y.T., Peikert, C., Vaikuntanathan, V.: Public-key encryption schemes with auxiliary inputs. In: TCC. pp. 361–381 (2010)
18. Dodis, Y., Haralambiev, K., López-Alt, A., Wichs, D.: Efficient public-key cryptography in the presence of key leakage. In: ASIACRYPT. pp. 613–631 (2010)
19. Dodis, Y., Kalai, Y.T., Lovett, S.: On cryptography with auxiliary input. In: ACM STOC. pp. 621–630 (2009)
20. Dodis, Y., Pietrzak, K.: Leakage-resilient pseudorandom functions and side-channel attacks on feistel networks. In: CRYPTO. pp. 21–40 (2010)
21. Faust, S., Hazay, C., Nielsen, J.B., Nordholt, P.S., Zottarel, A.: Signature schemes secure against hard-to-invert leakage. In: ASIACRYPT. pp. 98–115 (2012)
22. Faust, S., Pietrzak, K., Schipper, J.: Practical leakage-resilient symmetric cryptography. In: CHES. pp. 213–232 (2012)
23. Fujioka, A., Suzuki, K., Xagawa, K., Yoneyama, K.: Strongly secure authenticated key exchange from factoring, codes, and lattices. In: PKC. pp. 467–484 (2012)
24. Gandolfi, K., Mourtel, C., Olivier, F.: Electromagnetic analysis: Concrete results. In: CHES. pp. 251–261. No. Generators (2001)
25. Gennaro, R., Lindell, Y.: A framework for password-based authenticated key exchange. In: EUROCRYPT. pp. 524–543 (2003)

26. Halderman, J.A., Schoen, S.D., Heninger, N., Clarkson, W., Paul, W., Calandrino, J.A., Feldman, A.J., Appelbaum, J., Felten, E.W.: Lest we remember: Cold boot attacks on encryption keys. In: USENIX Security Symposium. pp. 45–60 (2008)
27. Halevi, S., Kalai, Y.T.: Smooth projective hashing and two-message oblivious transfer. J. Cryptology 25(1), 158–193 (2012)
28. Halevi, S., Lin, H.: After-the-fact leakage in public-key encryption. In: TCC. pp. 107–124 (2011)
29. Katz, J., Vaikuntanathan, V.: Signature schemes with bounded leakage resilience. In: ASIACRYPT. pp. 703–720 (2009)
30. Katz, J., Vaikuntanathan, V.: Round-optimal password-based authenticated key exchange. In: TCC. pp. 293–310 (2011)
31. Krawczyk, H.: SIGMA: the 'sign-and-mac' approach to authenticated diffie-hellman and its use in the ike-protocols. In: CRYPTO. pp. 400–425 (2003)
32. Kurosawa, K., Furukawa, J.: 2-pass key exchange protocols from cpa-secure KEM. In: CT-RSA. pp. 385–401 (2014)
33. LaMacchia, B.A., Lauter, K.E., Mityagin, A.: Stronger security of authenticated key exchange. In: ProvSec. pp. 1–16 (2007)
34. Marvin, R.: Google admits an android crypto prng flaw led to bitcoin heist (august 2013). http://sdt.bz/64008.
35. Micali, S., Reyzin, L.: Physically observable cryptography (extended abstract). In: TCC. pp. 278–296 (2004)
36. Moriyama, D., Okamoto, T.: Leakage resilient eck-secure key exchange protocol without random oracles. In: ASIACCS. pp. 441–447 (2011)
37. Naor, M., Segev, G.: Public-key cryptosystems resilient to key leakage. In: CRYPTO. pp. 18–35 (2009)
38. Okamoto, T.: Authenticated key exchange and key encapsulation in the standard model. In: ASIACRYPT. pp. 474–484 (2007)
39. Quisquater, J., Samyde, D.: Electromagnetic attack. In: Encyclopedia of Cryptography and Security, 2nd Ed., pp. 382–385 (2011)
40. Shumow, D., Ferguson, N.: On the possibility of a back door in the nist sp800-90 dual ec prng. http://rump2007.cr.yp.to/15-shumow.pdf.
41. Standaert, F., Pereira, O., Yu, Y., Quisquater, J., Yung, M., Oswald, E.: Leakage resilient cryptography in practice. In: Towards Hardware-Intrinsic Security - Foundations and Practice, pp. 99–134 (2010)
42. Yang, G., Mu, Y., Susilo, W., Wong, D.S.: Leakage resilient authenticated key exchange secure in the auxiliary input model. In: ISPEC. pp. 204–217 (2013)
43. Yu, Y., Standaert, F., Pereira, O., Yung, M.: Practical leakage-resilient pseudorandom generators. In: ACM CCS. pp. 141–151 (2010)
44. Yuen, T.H., Zhang, Y., Yiu, S., Liu, J.K.: Identity-based encryption with post-challenge auxiliary inputs for secure cloud applications and sensor networks. In: ESORICS. pp. 130–147 (2014)
45. Zetter, K.: How a crypto 'backdoor' pitted the tech world against the nsa. http://www.wired.com/threatlevel/2013/09/nsa-backdoor/all/.