Strongly Secure Authenticated Key Exchange from Ideal Lattices

Xiaopeng Yang and Wenping Ma

Abstract

In this paper, we propose an efficient and practical authenticated key exchange (AKE) protocol from ideal lattices, which is well-designed and has some similarity to the HMQV protocol. Using the hardness of the graded discrete logarithm (GDL) problem and graded decisional Diffie-Hellman (GCDH) problem, the proposed protocol is provably secure in the extended Canetti-Krawczyk model.

Index Terms

Lattice-based Cryptography, Authenticated Key Exchange, Graded encoding, eCK Model

I. Introduction

Key Exchange (KE) is an elementary cryptographical original, which permits any two participants to negotiate a shared session key over an open network. In other words, the adversary controls entire communications over the whole network. Because a common session key between any two participants is fundamental to hide the transferred data in an open channel, KE gradually becomes one extensively applied cryptographical instrument in constructing secure networks. In 1976, Diffie and Hellman proposed the wonderful Diffie-Hellman KE Protocol. The KE protocol fall into two categories: one is non-authenticated key exchange protocol; the other one is authenticated key exchange (AKE).

The authors are with the State Key Laboratory of Integrated Service Networks, Xidian University, Xi'an 710071, China(e-mail: xp_yang89xidian@126.com, wp_ma@mail.xidian.edu.cn)

For AKE, every participant owns certain public information, namely a static public key (SPK), which is issued by a trusted third party, such as public key infrastructure (PKI), or certification authority, and the homologous secret information, namely, a static secret key (SSK). During the execution of the agreement, each participant first generates his own ephemeral secret key (ESK) and concomitant ephemeral public key, and exchanges the ephemeral public key (EPK). Then, each participant uses their static public keys, the static secret keys, the ephemeral public keys, and the ephemeral secret keys to compute certain session state. Finally, each participant derives a common session key by using a robust extractor.

How to evaluate a cryptographic protocol is an important study of cryptography. Bellare and Rogaway first presented a security model of AKE called BR model, which was based on the indistinguishability between the real common session key and any random key uniformly chosen from the same distribution. This model is the most widely used security model considered as it is robust enough for some practical applications. In 2001, Canetti and Krawcyk projected CK security model. Moreover, their key exchange protocol realize authentication with message authentication code (MAC), which adds extra computation and communication overheads.

The traditional KE protocols include: KE based on discrete logarithm problems, KE based on RSA problems, KE based on key encapsulation mechanism (KEM), AKE based on bilinear pairings, and KE based on signature or MAC, etc. The design of efficient cryptographic KE protocol is among the core content of the research of cryptography. Especially, with the development of quantum computing technology, it is inspiring to design new alternatives based on other problems, which are recognized to have resistance to quantum attacks.

Menezes, Qu, and Vanstone put forward the MQV protocol. Due to its prominent security properties, the MQV protocol has been selected by National Security Agency (NSA) as the key exchange mechanism preferred to safeguard US government information. In 2001, Canetti and Krawcyk projected Canetti-Krawcyk (CK) security model. Moreover, it pointed that a combination of symmetrical encryption, message authentication code (MAC), and common session key contributes to build a secure channel for Internet.

But Krawczyk pointed that the MQV protocol is not resistant to some attacks such as unknown key share (UKS) attacks, key compromise impersonation (KCI) attacks, and disclosure of DH exponents. Besides, the MQV protocol lacks perfect forward secrecy (PFS). Thus, he proposed HMQV protocol that is resistant to the above attacks. This protocol utilized the Exponential Challenge-Response Signatures to realize authentication. In, LaMacchia et al. provided a security model called extended CK (eCK) model where more capabilities are granted to the adversaries. Specifically, it differentiate the exposure of static secret key and ephemeral private key and permits the exposure of ephemeral private key of test session. Meanwhile the first protocol (NAXOS) that is secure in eCK model is given as well as the new security model. Berkant Ustaoglu projected CMQV protocol, which captures the high performance of HMQV and security in eCK model.

Recently, cryptographic schemes based on lattices have appeared as a prospective replacement to more traditional ones based on the factoring and discrete logarithm problem. Moreover, lattice-based cryptography has several fascinating features. From a security perspective, the best attacks for quantum adversaries on the potential problems require exponential time in the primary security parameter. In addition, strong average-case/worst-case security reductions support security proofs in lattice-based cryptography. Lattice-based cryptography computations should be greatly simple, fast and parallelizable in the name of efficiency. Especially, public key encryptions from LWE and identity-based encryptions from LWE are widely used. But, AKE schemes based on LWE are only provably secure in BR model. In this paper, we build AKE from ideal lattices, which is secure in eCK model.

II. Preliminaries

A. Notations

In this paper, $\mathbb{C}, \mathbb{R}, \mathbb{Z}, \mathbb{Q}$ denote the set of complex numbers, the set of real numbers, the set of integers and the set of rational numbers, respectively.For $q \geq 1$, define $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$. Let λ be the security parameter, if a algorithm A runs in polynomial time (PT) of λ, then it is efficient. If a function $f(\lambda) = o(n^{-c})$,

where $c > 0$, then it is negligible. We make use of the Landau notations. For the algorithm A, if $|\Pr[A(X)] - \Pr[A(Y)]| \leq negl(\lambda)$, then these two distributions are computationally indistinguishable. A lattice $L \subseteq \mathbb{R}^n$ is a discrete additive subgroup of $\mathbb{R}^n$. For each lattice, there exists a set of linearly independent vector $\{\mathbf{b}_i\}_{i=1}^m$ satisfying $L = \{\sum_{i=1}^m z_i \mathbf{b}_i\}$, we say this vector set is the basis of the lattice L.

B. Gaussian Sampler and Gaussian Function

For real $\sigma > 0$, each $\mathbf{x} \in \mathbb{R}^n$, define the spherical gaussian function with parameter σ as

$$\rho_\sigma(\mathbf{x}) = \exp\{-\pi\|\mathbf{x}\|^2/\sigma^2\}.$$

For a rank-n matrix $S \in \mathbb{R}^n$, the ellipsoid Gaussian function on $\mathbb{R}^n$ with parameter S is defined by

$$\rho_S(\mathbf{x}) = \exp\{-\pi\mathbf{x}^T(S^TS)^{-1}\mathbf{x}\}.$$

For each $\mathbf{x} \in L$, the ellipsoid discrete Gaussian distribution over lattice L with parameter S is

$$D_{L,S}(\mathbf{x}) = \rho_S(\mathbf{x})/\rho_S(L).$$

For m linearly independent vectors $\mathbf{x}_i \leftarrow D_{L,S}$, we denote $X = (\mathbf{x}_1|\cdots|\mathbf{x}_m)^T$. We consider the distribution $D_{X,\sigma}$, included by choosing an integer vector $\mathbf{v}$ from a discrete Gaussian over $\mathbb{Z}^n$ with parameter σ and outputting $\mathbf{y} = X^T \cdot \mathbf{x}$, $\varepsilon_{X,\sigma} \doteq \{X^t\mathbf{v} : \mathbf{v} \leftarrow D_{\mathbb{Z}^m,\sigma}\}$. Agrawal proved that with high probability over the choice of X, the distribution $D_{X,\sigma}$ is statistically close to the ellipsoid Gaussian $D_{L,\sigma X}$, and the singular values of X are of size roughly $\sigma\sqrt{m}$.

Theorem 1. Let L be a full-rank lattice over $\mathbb{R}^n$ and B a matrix whose columns form a basis of L. Let $M \in \mathbb{R}^{n\times n}$ be a full rank matrix, and denote $S = M(B^T)^{-1}$, $s_1 = \sigma_1(S)$, $s_n = \sigma_n(S)$, and $\chi = s_1/s_n$. Let ϵ be negligible in n and m, s' be parameter satisfying $m \geq 10\log(8(mn)^{1.5}s_1\chi)$ and $s' \geq 4mn\chi\ln(1/\epsilon)$. If $s_n \geq \eta_\epsilon(\mathbb{Z}^n)$, then when choosing the rows of an m-by-n matrix X from the ellipsoid Gaussian over L, $X \leftarrow (D_{L,M})^m$, we obtain with all but probability $2^{-O(m)}$ over the choice of X, that the statistical distance between $\varepsilon_{X,s'}$ and the ellipsoid Gaussian $D_{L,s',X}$ is at most 2ϵ.

C. Cyclotomic Field, Cyclotomic Ring and Ideal Lattices

For positive integer m, let $K = \mathbb{Q}(\zeta_m)$ represent the m-th cyclotomic field, let $R = \mathbb{Z}[\zeta_m]$ represent the m-th cyclotomic ring, where ζ_m is an m order element. The unique monic polynomial $f(X) \in \mathbb{Q}[X]$ of minimal degree having ζ_m as root is called the m-th cyclotomic polynomial. Its complex roots are in the form of ω_m^i, where $i \in \mathbb{Z}_m^*, \omega_m = \exp(2\pi i/m) \in \mathbb{C}$. So, $[R : \mathbb{Z}] = \varphi(m), R \cong \mathbb{Z}[X]/(\Phi_m(X))$, where $\Phi_m(X) \in \mathbb{Z}[X]$. Specially, $\{\zeta_m^j\}_{j=0}^{\varphi(m)-1}$ is a $\mathbb{Z}$-basis of $R = \mathbb{Z}[\zeta_m]$.

For n a power of 2, let the $2n$-th cyclomic polynomial ring $R = \mathbb{Z}[X]/(X^n + 1)$. For $\mathbf{u} \in R$, we identify it with its coefficient vector of the degree $n-1$ integer polynomial that represent $\mathbf{u}$. Let $R_q = R/qR = \mathbb{Z}_q[X]/(X^n+1)$. For $\mathbf{g} \in R$, $\langle \mathbf{g} \rangle$ represents the ideal in R that is generated by $\mathbf{g}$. We denote it as $\langle \mathbf{g} \rangle = \{\mathbf{g} \cdot \mathbf{u} : \mathbf{u} \in R\}$, and call it an ideal lattice. Let $\mathbf{B}(\mathbf{g}) = \{\mathbf{g}, X\mathbf{g}, X^2\mathbf{g}, \cdots, X^{n-1}\mathbf{g}\}$ denote the basis of the ideal lattice $\langle \mathbf{g} \rangle$. For $\mathbf{u} \in R$, $[\mathbf{u}]_{\mathbf{g}}$ represents the unique element $\mathbf{u}' \in R$ satisfying $\mathbf{u} - \mathbf{u}' \in \langle \mathbf{g} \rangle$ and $\mathbf{u}' = \sum_{i=1}^{n-1} \alpha_i X^i \mathbf{g}$, where all the α_i's are in the interval $[-\frac{1}{2}, \frac{1}{2})$. Similarly, $[t]_p$ represents the reduction of $t \mod p$ into the interval $[-\frac{p}{2}, \frac{p}{2})$, for integers t, p.

D. Encoding Thought

In this subsection, we present our encoding thought, which has some similarity to GGH's Graded Encoding Scheme. In fact, our encoding thought is a self-contained design. Let the cyclotomic Ring $R = \mathbb{Z}[X]/(X^n + 1)$, and let $R_q = R/qR$. We samples a short vector $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$ with $\sigma = \widetilde{O}(\sqrt{n})$. The choice of the generator $\mathbf{g}$ guarantees $\mathbf{g}$ and $\mathbf{g}^{-1}$ are short in $K = \mathbb{Q}[X]/(X^n + 1)$. We samples $\mathbf{z} \in R_q$ uniformly at random. Let a quotient ring R/I. For each coset $\mathbf{e} + I \in R/I$, we encode it as follows:

The level-zero encoding of $\mathbf{e} + I \in R/I$ is to draw a random short vector in R, namely, $\mathbf{d} \leftarrow D_{\mathbb{Z}^n,\sigma}$. The level-one encoding of $\mathbf{e} + I \in R/I$ is a element of the form $\mathbf{c}/\mathbf{z} \in R_q$ with $\mathbf{c} \in \mathbf{e} + I$ is short. For $i = 0, 1$, we define the set of level-zero or one encodings as follows:

$$S_i = \{\mathbf{c}/\mathbf{z} \in R_q : \|\mathbf{c}\| < q^{1/8}\} \text{ and } S_i^{(\mathbf{e}+I)} = \{\mathbf{c}/\mathbf{z}^i \in R_q : \mathbf{c} \in \mathbf{e} + I, \|\mathbf{c}\| \leq q^{1/8}\}.$$

We introduce the graded encoding system below in detail.

System Generation. $\mathbf{params} = (n, q, \mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, s) \leftarrow \mathbf{InstGen}$: and a zero testing parameter $\mathbf{P}_{zt} = [\mathbf{hz}/\mathbf{g}]_q$ with $\mathbf{h} \leftarrow D_{\mathbb{Z}^n,\sqrt{q}}$, where $\mathbf{g}$ is the generator of the ideal lattice $\langle \mathbf{g} \rangle$. $\mathbf{y}$ is the level-one encoding of $\mathbf{1} + I$, namely, we sample $\mathbf{a} \leftarrow D_{\mathbf{1}+I,\sigma n}$, and compute $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$. The specific method of doing this is sampling $\mathbf{y}' \leftarrow D_{J',s}$ with $J' = \frac{1}{\mathbf{z}} + J$ and $J = \langle \mathbf{g}/\mathbf{z} \rangle \subseteq K$, then computing $\mathbf{a} = \mathbf{y}' \cdot \mathbf{z}$, finally setting $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$. A set of m vectors $\{\mathbf{x}_i\}_{i=1}^m$ is used for re-randomization, which are the level-one random encodings of the coset $\mathbf{0}+I$, namely, we sample $\mathbf{b}_i \leftarrow D_{I,\sigma n}$, and then compute $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$. The specific method of doing this is sampling a vector $\mathbf{x}_i' \leftarrow D_{J,s}$ from the fractional ideal $J = \langle \mathbf{g}/\mathbf{z} \rangle \subseteq K$ with $s = \sigma n^2/q$, and then computing $\mathbf{x}_i' = \mathbf{b}_i/\mathbf{z}$, finally setting $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$.

Sampling Level-zero Encodings. $\mathbf{d} \leftarrow \mathbf{samp}(\mathbf{params})$: For arbitrary coset, its level-zero encoding is $\mathbf{d} \leftarrow D_{\mathbb{Z}^n,\sigma n}$ with $\sigma' = n\sigma$. Since $\sigma' > \eta_{2^{-\lambda}}(I)$, the distribution of the samples is statistically close to the uniform distribution with an overwhelming advantage. Observe that the size of the level-zero encoding is at most $\sigma'\sqrt{n}$.

Encoding at level-one. $\mathbf{u}_1 \leftarrow \mathbf{enc}(\mathbf{params}, \mathbf{1}, \mathbf{d})$: Given the level-zero encoding $\mathbf{d}$, we multiply $\mathbf{y}$ by $\mathbf{d}$ to compute $\mathbf{u}_1 = [\mathbf{yd}]_q = [\mathbf{da}/\mathbf{z}]_q$, where $\mathbf{da} \in \mathbf{d} + I$.

re-Randomizations. $\mathbf{u}_1' \leftarrow \mathbf{reRand}(\mathbf{params}, 1, \mathbf{u}_1)$: We denote $\mathbf{X} = (\mathbf{x}_1|\cdots|\mathbf{x}_m)^T$ and $\mathbf{B} = (\mathbf{b}_1|\cdots|\mathbf{b}_m)^T$. In order to obtain re-randomization of level-one encoding $\mathbf{u}_1 = [\mathbf{c}/\mathbf{z}]_q$ with $\|\mathbf{c}\| \leq \gamma$, we sample $\mathbf{r} \leftarrow D_{\mathbb{Z}^m,\sigma^*}$ with $\sigma^* = 2^\lambda\sigma$, then compute and output $\mathbf{u}_1' = [\mathbf{u}_1 + \sum_{i=1}^m \mathbf{r}_i\mathbf{x}_i]_q = [\frac{\mathbf{c}+\sum_i r_i\mathbf{b}_i}{\mathbf{z}}]_q$:. Observe that since $\|\mathbf{b}_i\| \leq \sigma n^4$, then $\|\mathbf{Br}\| \leq \sigma^*\sigma\sqrt{mn}n^4$. If $\|\mathbf{c}\| \leq \gamma$, then $\|\mathbf{c} + \mathbf{Br}\| \leq \gamma + \sigma^*\sigma\sqrt{mn}n^4 < \sigma^*\sigma\sqrt{mn}n^4(1 + 2^{-\lambda})$.

Adding Encodings. $\mathbf{add}(\mathbf{params}, 1, \mathbf{u}_1, \mathbf{u}_2)$: For a level-one encoding $\mathbf{u}_1$ of $\mathbf{e}_1 + I$ and the level-one encoding $\mathbf{u}_2$ of $\mathbf{e}_2 + I$, we compute $\mathbf{u}_1 + \mathbf{u}_2 = [\mathbf{y} \cdot (\mathbf{d}_1 + \mathbf{d}_2)]_q = [\mathbf{a}(\mathbf{d}_1 + \mathbf{d}_2)/\mathbf{z}]_q$, which indicates that the level-one encoding of the coset $(\mathbf{e}_1 + \mathbf{e}_2) + I$. This action means that adding an encodings obtains the encoding of the sum of two cosets.

Type One of Multipying Encodings. $\mathbf{mult}(\mathbf{params}, 1, a, \mathbf{u}_1)$: For an integer $a \in \mathbb{Z}^+$, a level-one encoding $\mathbf{u}_1$ of the coset $\mathbf{e}_1 + I$, we compute $[a \cdot \mathbf{y} \cdot \mathbf{d}_1]_q = [a \cdot \mathbf{ad}_1/\mathbf{z}]_q$.

Type Two of Multipying Encodings. $\mathbf{mult}(\mathbf{params}, 1, \mathbf{u}_0, \mathbf{u}_1)$: For a level-zero encoding $\mathbf{u}_0$ of the coset $\mathbf{c}_0 + I$, a level encoding $\mathbf{u}_1$ of the coset $\mathbf{c}_1 + I$, we compute $\mathbf{u} = \mathbf{u}_0 \cdot \mathbf{u}_1 = [\mathbf{c}_0 \cdot \mathbf{c}_1/\mathbf{z}]_q$.

Zero Testing. $\mathbf{isZero}(\mathbf{params}, \mathbf{P_{zt}}, \mathbf{u}_1)$: We compute $w = [\mathbf{P}_{zt} \cdot \mathbf{u}_1]_q$. If $\|w\| \leq q^{3/4}$, then this can explain that $\mathbf{u}_1$ is a level-one encoding of the coset $) + I$. Otherwise, not.

Robust Extractor. $\mathbf{s} \leftarrow \mathbf{ext}(\mathbf{params}, \mathbf{P}_{zt}, \mathbf{u})$: We compute $\mathbf{w} = [\mathbf{u} \cdot \mathbf{P}_{zt}]_q$, and then collect the $\frac{\log_2 q}{4} - \lambda$ most-significant bits of each coefficient of the result, finally apply a strong randomness extractor to the colllected bits.

E. Hard Assumptions

In this subsection, we review some helpful hard assumptions for our authenticated key exchange (AKE) protocol.

Graded Discrete Logarithm. For an adversary $\mathcal{A}$ and parameters λ, κ and $\mathbf{v}_{zt}$, consider the following:

1. $(\mathbf{params}, \mathbf{P}_{zt}) \leftarrow \mathbf{InstGen}(1^\lambda, 1^\kappa, \mathbf{v}_{zt})$;
2. $(a, b) \leftarrow \mathbf{samp}(\mathbf{params})$;
3. $(\mathbf{u}_1, b_1) \leftarrow \mathbf{reRand}(\mathbf{params}, \mathbf{e}_1, \mathbf{enc}(\mathbf{params}, 1, (a, b)))$;
4. Run the adversary to obtain $a' \leftarrow \mathcal{A}(\mathbf{params}, \mathbf{P}_{zt}, (\mathbf{u}_1, b_1), \ldots, (\mathbf{u}_t, b_t))$.

$\mathcal{A}$ is considered successful if $a' - a \in S_{\mathbf{0}}^{(0)}$, namely, a' and a belong to the same encoding set $S_{\mathbf{0}}^{(\alpha)}$.

In our applications, the adversary can see the public parameters $\mathbf{params} = (\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m)$, where $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ is a level-one encoding of $\mathbf{1} + I$, and each $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$ is a level-one encoding of $\mathbf{0} + I$. Review $I = \langle \mathbf{g} \rangle$ with $\|\mathbf{g}\| = q^{o(1)}$, and a level-one encoding of the coset $\alpha + I$ is an element of the form $\mathbf{u} = [\mathbf{c}/\mathbf{z}]_q$, where $\mathbf{c} \in \alpha + I$ is short with $\|\mathbf{c}\| = q^{o(1)}$. Also, the adversary can see the zero-testing parameter $\mathbf{P}_{zt} = [\mathbf{hz}/\mathbf{g}]_q$ with $\|\mathbf{h}\| = q^{1/2+o(1)}$. Consider the following procedure, on parameters $n, \lambda, q, \kappa = 1, \sigma = poly(n), \sigma^* = \sigma \cdot 2^\lambda$:

- $(\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}) \leftarrow \mathbf{InstGen}(1^n, 1^\kappa)$;
- For $i = 0, 1$,

1) sample $\mathbf{e}_i \leftarrow D_{\mathbb{Z}^n,\sigma}$, and sample $\mathbf{f}_i \leftarrow D_{\mathbb{Z}^n,\sigma}$, where $\mathbf{e}_i \in \eta_i + I$, and $\mathbf{f}_i \in \phi_i + I$;

2) set $\mathbf{u}_i = [\mathbf{e}_i \cdot \mathbf{y} + \sum \mathbf{r}_{ij}\mathbf{x}_{ij}]_q$.

- set $\mathbf{v} = [\mathbf{e}_0\mathbf{u}_i]_q$ // encoding of the real product
- set $\mathbf{v}' = [\mathbf{f}_0\mathbf{u}_i]_q$ // encoding of a random product

GCDH. The Graded Computational Diffie-Hellman Assumption is, on input $((\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}), \mathbf{u}_0, \mathbf{u}_1)$ to output a level-one encoding of $(\mathbf{e}_0 + I)(\mathbf{e}_1 + I)$, particularly $\mathbf{w} \in R_q$ satisfying $\|[\mathbf{P}_{zt}(\mathbf{v} - \mathbf{w})]_q\| < q^{3/4}$.

GDDH. The Graded Decisional Diffie-Hellman Assumption is to distinguish between $\mathbf{v}$ and $\mathbf{v}'$. More specifically, the adversary can differentiate between the distributions

$$\mathcal{D}_{GDDH}\{(\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}), \mathbf{u}_0, \mathbf{u}_1, \mathbf{v}\} \text{ and } \mathcal{D}_{RAND}\{(\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}), \mathbf{u}_0, \mathbf{u}_1, \mathbf{v}'\}.$$

F. Extended Canetti-Krawczyk Model

In this section, we review the extended Canetti-Krawczyk (eCK) model. Let κ be the security parameter. Each party has a static secret key and a homologous static public key, which are guaranteed by a certificate authority.

Session. Each party is activated by an incoming message to execute the protocol Π. Each party is modeled as a probabilistic polynomial time (PPT) Turing machine. An execution of protocol is called a session. Suppose party $\widehat{A}$ and party $\widehat{B}$ are the session initiator and the session responder, respectively. Then party $\widehat{A}$ is activated by the outside call $(\widehat{A}, \widehat{B})$ or $(\widehat{A}, \widehat{B}, Y)$. When activated by $(\widehat{A}, \widehat{B})$, then party $\widehat{A}$ computes its ephemeral public key X and stores its session state. The session identifier in $\widehat{A}$ is initialized with $(\widehat{A}, \widehat{B}, X, *, init)$. When activated by $(\widehat{A}, \widehat{B}, Y)$, then the session identifier in $\widehat{A}$ is updated to $(\widehat{A}, \widehat{B}, X, Y, init)$. Similarly, party $\widehat{B}$ is activated by $(\widehat{B}, \widehat{A}, X)$. When activated, $\widehat{B}$ also computes its ephemeral public key Y and stores its session state. In this case, the session identifier in $\widehat{B}$ is $(\widehat{B}, \widehat{A}, Y, X, resp)$. A session $(\widehat{B}\widehat{A}, Y, X, resp)$ is called to be matching to the session $(\widehat{A}, \widehat{B}, X, Y, init)$. For $(\widehat{A}, \widehat{B}, *, *, role)$, $\widehat{A}$ is the owner of the session while $\widehat{B}$ is called the peer of the session. If the owner of the session has computed the session key, then we said the session is complete.

Adversary. Adversary $\mathcal{A}$ is modeled as a PPT machine, which dominates the whole networks. Specifically, it is allowed to make the following queries:

1. Establish party $\widehat{C}$: $\mathcal{A}$ registers an arbitrary party $\widehat{C}$, whose static public key is on $\mathcal{A}$;s own choice. We say this kind of new registered parties dishonest. We need that when $\mathcal{A}$ makes this query, the certifying center should verify the submitted static public key is in the suitable group and the proof that $\mathcal{A}$ knows the corresponding static private key.

2. Send $(\widehat{A}, m)$: $\mathcal{A}$ sends a message m to $\widehat{A}$. Once $\widehat{A}$ is activated by m, then $\mathcal{A}$ obtains the outgoing message of $\widehat{A}$.

3. Ephemeral key reveal $(\widehat{A})$: $\mathcal{A}$ obtains the ephemeral private key of the session sid.

4. Static key reveal $(\widehat{A})$: $\mathcal{A}$ obtains the static secret key of party $\widehat{A}$. In this case, we call $\widehat{A}$ is dishonest.

5. Session key reveal (sid): $\mathcal{A}$ obtains the session key of the session if the session is completed.

6. Test (sid): $\mathcal{A}$ can make this query only once. After receiving this query, oracle randomly chooses bit $b \in_{\mathcal{R}} \{0, 1\}$. For $b = 0$, $\mathcal{A}$ acquires session key of sid. Otherwise, it gets random key, which is selected from the same distribution of real key.

Experiment. We permit $\mathcal{A}$ to send any series of above mentioned queries. Finally, outputs guess b'. When $b' = b$, we say $\mathcal{A}$ win this game.

Definition 1$(\mathrm{Freshness})$. Let sid be a complete session, owned by honest $\widehat{A}$ with honest peer $\widehat{B}$. If the matching session of sid exists, let sid^* be the session identifer of its matching session. We say sid is to be fresh, when none of the following events occurs:

1. $\mathcal{A}$ makes Session key reveal (sid^*) query or Session key reveal sid^* if sid^* exists;

2. If sid^* exists, $\mathcal{A}$ makes either of the following quires:

(1). Both static key reveal $(\widehat{A})$ and ephemeral key reveal (sid), or

(2). Both static key reveal $(\widehat{B})$ and ephemeral key reveal (sid^*).

3. if sid^* does not exist, $\mathcal{A}$ makes either of the following quires:

(1). Both static key reveal $(\widehat{A})$ and ephemeral key reveal (sid), or

(2). Static key reveal $(\widehat{B})$.

Definition 2(eCK Security). Define the advantage of $\mathcal{A}$ as follows:

$$\mathbf{Adv}_{\Pi}^{\mathbf{AKE}}(\mathcal{A}) = |\mathbf{Pr}[b' = b] - \frac{1}{2}|.$$

We say AKE protocol is secure, when the following situations satisfy:

1. If two honest parties finish matching sessions, then they compute an identical session key with an overwhelming probability;

2. For any PPT adversary $\mathcal{A}$, its adversary is negligible.

III. THE PROTOCOL

We present our AKE protocol via Encoding ideal in this section. Let $\mathbf{H}$ be a hash function. Party $\widehat{A}$ computes $\mathbf{a} \leftarrow \mathbf{samp}(\mathbf{params})$, and serves it as party $\widehat{A}$'s static secret key, and computes $\mathbf{A} \leftarrow \mathbf{reRand}(\mathbf{params}, 1, \mathbf{enc}(\mathbf{params}, 1, \mathbf{a}))$ and uses it as his own static public key. Similarly, Party $\widehat{B}$ computes $\mathbf{b} \leftarrow \mathbf{samp}(\mathbf{params})$, and serves it as party $\widehat{B}$'s static secret key, and computes $\mathbf{B} \leftarrow \mathbf{reRand}(\mathbf{params}, 1, \mathbf{enc}(\mathbf{params}, 1, \mathbf{b}))$ and uses it as his own static public key. From here on, we omit the notation of $\mathbf{params}$ in each encoding operation, for convenience. Our protocol is shown in figure 1.

Initiate.$(\widehat{A}, \widehat{B})$: Party $\widehat{A}$ computes $\mathbf{x} \leftarrow \mathbf{samp}(\mathbf{params})$, and computes $\mathbf{X} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$, finally builds the local session identical $(\widehat{A}, \widehat{B}, \mathbf{X})$ and sends $(\widehat{A}, \mathbf{X})$ to party $\widehat{B}$. Meanwhile, party $\widehat{B}$ computes $\mathbf{y} \leftarrow \mathbf{samp}(\mathbf{params})$, and computes $\mathbf{Y} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$, finally builds the local session identical $(\widehat{A}, \widehat{B}, \mathbf{Y})$ and sends $(\widehat{B}, \mathbf{Y})$ to party $\widehat{B}$.

Respond.$(\widehat{A}, \widehat{B}, \mathbf{Y})$: Party $\widehat{A}$ first checks whether $\mathbf{Y}$ is a level-one encoding of $\mathbf{0} + I$ or not. If it is, then party $\widehat{A}$ computes and outputs $\mathbf{X}$, and builds the local session identical $(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$, finally computes its own session key $\mathbf{s}_{\widehat{A}} \leftarrow \mathbf{ext}(\mathbf{P}_{zt}, \mathbf{z}_{\widehat{A}})$. Simultaneously, party $\widehat{B}$ also checks whether $\mathbf{X}$ is a level-one encoding of $\mathbf{0} + I$ or not. If it is, then party $\widehat{B}$ computes and outputs $\mathbf{Y}$, and builds the local session identical $(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$, finally computes its own session key $\mathbf{s}_{\widehat{B}} \leftarrow \mathbf{ext}(\mathbf{P}_{zt}, \mathbf{z}_{\widehat{B}})$.

Complete.$(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$: Party $\widehat{A}$ checks whether $\mathbf{Y}$ is a level-one encoding of $\mathbf{0}+I$, and whether there exists a session identical $(\widehat{A}, \widehat{B}, \mathbf{Y})$. If any of two conditions fails, then $\widehat{A}$ ignores this activation. Otherwise, $\widehat{A}$ builds the local session identical $(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$, and computes its own session key $\mathbf{s}_{\widehat{A}} \leftarrow \mathbf{ext}(\mathbf{P}_{zt}, \mathbf{z}_{\widehat{A}})$.

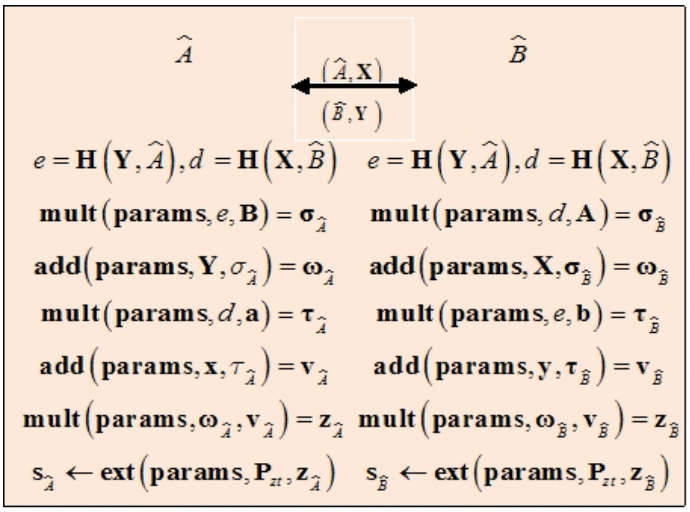

Fig. 1. Our AKE from Ideal Lattices

IV. CHALLENGE-RESPONSE ENCODING SCHEME

We first put forward a new notion of the challenge-response encoding scheme in this section, which is the primary constructing foundation in the design and analysis of our AKE protocol from ideal lattices. Each coder has a pair of ephemeral static key and ephemeral secret key that are used for generating and verifying each encoding. Our challenge-response encoding scheme is different from ordinary encoding schemes. Compared with general encoding schemes, our challenge-response encoding scheme is interactive and demands the receiver of the scheme to send a challenge to the coder, and then the coder can generate the corresponding encoding on a given message. A secure challenge-response encoding scheme has guaranteed that no one other than the coder can generate a encoding that will convince the challenge to accept it as a valid. In addition, the challenge also can verify the legitimacy of the generated encoding.

Definition 3(**Challenge-Response Encoding.**) We use $\widehat{B}$ to denote the coder, who owns a public key

$\mathbf{B} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{b}))$ and a secret key $\mathbf{b} \leftarrow \mathbf{samp}(params)$. Meanwhile, we use $\widehat{A}$ to denote the challenge, who provides a message m and a challenge $\mathbf{X}$ to the coder, where $\mathbf{X} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$, and $\mathbf{x} \leftarrow \mathbf{samp}(\mathbf{params})$. Utilizing challenge $\mathbf{X}$, $\widehat{B}$ codes message m to $(\mathbf{Y}, \mathbf{mult}(\mathbf{params}, \mathbf{X}, m_{\widehat{B_2}}))$, where $\mathbf{mult}(\mathbf{params}, \mathbf{H}(\mathbf{Y}, m), \mathbf{b}) = m_{\widehat{B_1}}$. The challenger $\widehat{A}$ accepts encoding $(\mathbf{Y}, m)$ as valid if and only if $\mathbf{Y}$ is not a level-one encoding of $\mathbf{0}+I$, and $\mathbf{mult}(\mathbf{params}, m_{\widehat{A_2}}, \mathbf{x}) = \sigma$, where $\mathbf{add}(\mathbf{params}, \mathbf{y}, m_{\widehat{A_1}}) = m_{\widehat{A_2}}$, and $\mathbf{mult}(\mathbf{params}, \mathbf{H}(\mathbf{Y}, m), \mathbf{B}) = m_{\widehat{A_1}}$.

Remark 1. Given message m, challenge $\mathbf{X}$, define a procedure of computing the encoding $\mathbf{Encode}_{\widehat{B}}(\mathbf{Y}, m, \mathbf{X})$ as follows:

1) compute $\mathbf{mult}(\mathbf{params}, \mathbf{H}(\mathbf{Y}, m), \mathbf{b}) = m_{\widehat{B_1}}$;
2) compute $\mathbf{add}(\mathbf{params}, \mathbf{y}, m_{\widehat{B_1}}) = m_{\widehat{B_2}}$;
3) compute $\mathbf{mult}(\mathbf{params}, \mathbf{X}, m_{\widehat{B_2}}) = \mathbf{Encode}_{\widehat{B}}(\mathbf{Y}, m, \mathbf{X})$.

Definition 4(**Security of the challenge-response encoding scheme.**) We say a challenge-response encoding scheme is secure, if there does not exist a polynomial time turning $\mathcal{F}$ which can win the game below with non-negligible probability.

The encoding forger $\mathcal{F}$ in Definition

1) $\mathcal{F}$ is given values $\mathbf{B}$ and $\mathbf{X}_0$, where $\mathbf{B} \leftarrow \mathbf{reRand}(\mathbf{params}, 1, \mathbf{enc}(\mathbf{params}, 1, \mathbf{b}))$
2) $\mathcal{F}$ is given to an encoding oracle $\widehat{B}$ (representing a coder $\widehat{B}$ with private key $\mathbf{b}$ and public key $\mathbf{B}$) that on query $(\mathbf{X}, m)$ outputs an encoding pair $(\mathbf{Y}, \mathbf{Encode}_{\widehat{B}}(\mathbf{Y}, m, \mathbf{X}))$, where $\mathbf{Y} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{y}))$, and $\mathbf{y} \leftarrow \mathbf{samp}(\mathbf{params})$ is chosen by $\widehat{B}$ renewedly with each query.
3) $\mathcal{F}$ is permitted a polynomial number of queries to $\widehat{B}$, while each query is chosen adaptively by $\widehat{B}$.
4) $\mathcal{F}$ halts with a symbol "fail" or with a guess $(\mathbf{Y}_0, m_0, \sigma)$. This guess is called 'forgery', if it satisfies the following two conditions:
   a) The pair $(\mathbf{Y}_0, \sigma)$ is a valid challenge-response encoding on message m_0 with regard to challenge $\mathbf{Y}_0$. That is, $\mathbf{Y}_0$ is not an encoding of $) + I$, and $\sigma = \mathbf{Encode}_{\widehat{B}}(\mathbf{Y}_0, m_0, \mathbf{X}_0)$.
   b) The pair $(\mathbf{Y}_0, m_0)$ did not present in any response of $\widehat{B}$ to $\mathcal{F}$'s queries.

We say that $\mathcal{F}$ win the game, if it outputs a successful encoding forgery.

Theorem 2 Under the GCDH assumption, the proposed challenge-response encoding scheme is secure in the random oracle model.

Proof. Given an efficient valid forgery $\mathcal{F}$ against the challenge-response encoding scheme, now we can construct an efficient solver $\mathcal{C}$ for the GCDH problem. That is, $\mathcal{C}$ obtains $((\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}), \mathbf{u}_0, \mathbf{u}_1, \mathbf{v})$, and then with a non-negligible probability outputs a level-one encoding $\mathbf{w} \in R_q$ of $(\mathbf{e}_0 + I) \cdot (\mathbf{e}_1 + I)$ satisfying $\|[\mathbf{P}_{zt} \cdot (\mathbf{v} - \mathbf{w})]_q\| \leq q^{3/4}$. Our idea is that if $\mathcal{F}$ can output a successful encoding forgery with input $(\mathbf{Y}_0, m_0)$ and a given value $(\mathbf{H}(\mathbf{Y}_0, m_0))$, then $\mathcal{F}$ can also output a successful encoding forgery even though $\mathbf{H}(\mathbf{Y}_0, m_0)$ is set to be a random value. We construct $\mathcal{C}$ such that after running $\mathcal{F}$ twice, $\mathcal{C}$ with a non-negligible probability obtains two different encoding forgery with respect to $(\mathbf{Y}_0, m_0)$ but with two different random values $\mathbf{H}(\mathbf{Y}_0, m_0)$. Utilizing these two encoding forgery, $\mathcal{C}$ can solve the GCDH problem. Observe that in the run of $\mathcal{F}$ under $\mathcal{C}$, all the queries to oracle $\widehat{B}$ are responded by $\mathcal{C}$ without knowledge of $\widehat{B}$'s private key $\mathbf{b}$. $\mathcal{C}$ simulates the response in Step1-Step3. $\mathbf{Y}$ is computed by $\mathcal{C}$, and is a re-randomization of a level-one encoding. According to theorem 1, with overwhelming probability over the choice of the randomizers $\mathbf{x}_i'$, the distribution of $\sum_i r_i \mathbf{x}_i'$ is statistically close to an ellipsoid Gaussian distribution over the fractional ideal $\mathcal{J} = \langle g/\mathbf{z} \rangle$. We select a parameter σ^* to ensure that the width of the distribution of $\sum_i r_i \mathbf{x}_i'$ is much larger than the size of $\mathbf{c}/\mathbf{z}$, guaranteeing the distribution of $\mathbf{c}'/\mathbf{z} + \sum_i r_i \mathbf{x}_i'$ is almost irrelevant to the distribution of $\mathbf{c}/\mathbf{z}$. Namely, the re-randomization of a level-one encoding is independent of the initial level-one encoding. then the probability that the the random value $(\mathbf{Y}, m)$ queried from $\mathbf{H}(\cdot)$ is at most $Q/|I|$, where Q is an upper bound on the number of queries to $\mathbf{H}(\cdot)$ that happens in the run of $\mathcal{C}$. Thus, the probability that $\mathcal{F}$ outputs a successful encoding forgery in the interaction with $\mathcal{C}$ is almost identical with, up to a negligible difference, the probability that $\mathcal{F}$ outputs a successful encoding forgery in the real run.

Constructing a solver $\mathcal{C}$ for the GCDH problem from the encoding fogery $\mathcal{F}$

Setup. Given a successful encoding forgery $\mathcal{F}$, we construct a GCDH solver $\mathcal{C}$. On input $(\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m, \mathbf{P}_{zt}, \mathbf{u}_0, \mathbf{u}_1)$,

$\mathcal{C}$ outputs a level-one encoding $\mathbf{w} \in R_q$ satisfying $\|[\mathbf{P}_{zt} \cdot (\mathbf{v} - \mathbf{w})]_q\| \leq q^{3/4}$.

$\mathcal{C}$ **'s Actions.** $\mathcal{C}$ sets $\mathbf{B} = \mathbf{u}_1$ and $\mathbf{X}_0 = \mathbf{u}_0$, then runs $\mathcal{F}$ on input $(\mathbf{B}, \mathbf{X}_0)$ against encoding oracle $\widehat{B}$ with public key $\mathbf{B}$. $\mathcal{C}$ provides a random tape for $\mathcal{F}$ and offers random answers to $\mathbf{H}(\cdot)$ queries. If each query to $\mathbf{H}(\cdot)$ is identical, then each answer from $\mathcal{C}$ is the same as the first one. Each time $\mathcal{F}$ queries $\widehat{B}$ for an encoding on message $(\mathbf{X}, m)$, $\mathcal{C}$ answers queries as follows:

- **Simulation 1.** $\mathcal{C}$ samples $\mathbf{s} \leftarrow \mathbf{samp}(\mathbf{params})$, and samples $\mathbf{e} \in \{0,1\}^{\ell}$;
- **Simulation 2.** $\mathcal{C}$ computes $\mathbf{Y}' \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{s}))$, and computes $\mathbf{Y}'' \leftarrow \mathbf{mult}(\mathbf{params}, \mathbf{e}, \mathbf{B})$;
- **Simulation 3.** $\mathcal{C}$ sets $\mathbf{Y} = \mathbf{Y}' + (-\mathbf{Y}'')$;
- **Simulation 4.** $\mathcal{C}$ sets $\mathbf{H}(\mathbf{Y}, m) = \mathbf{e}$. If $\mathbf{H}(\mathbf{Y}, m)$ has been defined by a previous query to $\mathbf{H}(\cdot)$, then $\mathcal{C}$ halts and outputs 'fail'. $\mathcal{C}$ responses to $\mathcal{F}$ with an encoding value $(\mathbf{Y}, \mathbf{mult}(\mathbf{params}, \mathbf{s}, \mathbf{X}))$. When $\mathcal{F}$ halts its run, $\mathcal{C}$ verifies whether the following conditions hold:
  1) $\mathcal{F}$ outputs a guess $(\mathbf{Y}_0, m_0, \sigma)$, where $\mathbf{Y}_0$ is not a level-one encoding of $\mathbf{0} + I$.
  2) $(\mathbf{Y}_0, m_0, \sigma)$ was not used as the response in the previous encodings generated by $\widehat{B}$.
  3) The value $\mathbf{H}(\mathbf{Y}_0, m_0)$ was queried from the random oracle $\mathbf{H}$.

If these three conditions hold, then $\mathcal{C}$ executes the following repeat experiment. Otherwise, $\mathcal{C}$ halts and outputs "fail".

Repeat experiment. $\mathcal{C}$ runs $\mathcal{F}$ again for the second time using the same inputs $(\mathbf{B}, \mathbf{X}_0)$ and the same random coins for $\mathbf{H}$ and $\widehat{B}$. The difference between the two runs is that all queries to $\mathbf{H}$ executed before the $\mathbf{H}(\mathbf{Y}_0, m_0)$ query are answered identically with the first run. But when $\mathcal{F}$ answers the query to $\mathbf{H}(\mathbf{Y}_0, m_0)$, $\mathcal{F}$ chooses $\mathbf{e}' \in \{0,1\}^{\ell}$ afresh.

Output. If the second run finishes, $\mathcal{F}$ outputs guess $(\mathbf{Y}_0, m_0, \sigma')$ with $\mathbf{e}' = \mathbf{e}$, then $\mathcal{C}$ computes $\mathbf{W}_0$ and outputs it as a guess for the GCDH problem. Now we give the calculation procedure for $\mathbf{W}_0$ as follows:

1) Compute $\mathbf{mult}(\mathbf{params}, \mathbf{e}, \mathbf{B}) = \sigma_{11}$;
2) Compute $\mathbf{add}(\mathbf{params}, \mathbf{Y}, \sigma_{11}) = \sigma_{12}$;
3) Compute $\mathbf{mult}(\mathbf{params}, \mathbf{x}_0, \sigma_{12}) = \sigma$;

4) Compute $\mathbf{mult}(\mathbf{params}, \mathbf{e}', \mathbf{B}) = \sigma'_{11}$;

5) Compute $\mathbf{add}(\mathbf{params}, \mathbf{Y}, \sigma'_{11}) = \sigma'_{12}$;

6) Compute $\mathbf{mult}(\mathbf{params}, \mathbf{x}_0, \sigma'_{12}) = \sigma'$;

7) Compute $\sigma - \sigma'$;

8) Compute $\mathbf{mult}(\mathbf{params}, (\sigma - \sigma'), (\mathbf{e} - \mathbf{e}')^{-1})$.

V. DUAL CHALLENGE-RESPONSE ENCODING SCHEME

Definition 5(**Dual Challenge-Response Encoding Scheme.**) Let $\widehat{A}$ and $\widehat{B}$ be two party with public key $\mathbf{A} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{a}))$ and $\mathbf{B} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{b}))$ respectively. Let m_1 and m_2 be two messages. The dual challenge-response encoding by $\widehat{A}$ and $\widehat{B}$ on message m_1 and m_2 is defined as $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{DEncode}_{\widehat{A},\widehat{B}}(m_1, m_2, \mathbf{X}, \mathbf{Y})$. The special calculation steps is shown as follows:

- $\widehat{A}$ samples $\mathbf{x} \leftarrow \mathbf{samp}(\mathbf{params})$, and computes $\mathbf{X} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$. $\widehat{B}$ samples $\mathbf{y} \leftarrow \mathbf{samp}(\mathbf{params})$, and computes $\mathbf{Y} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{y}))$.
- $\widehat{A}$ computes $\mathbf{d} = \mathbf{H}(\mathbf{X}, m_1)$, and computes $\mathbf{mult}(\mathbf{params}, \mathbf{d}, \mathbf{a}) = m_{11}$.
- $\widehat{A}$ computes $\mathbf{add}(\mathbf{params}, \mathbf{X}, m_{11}) = m_{12}$.
- $\widehat{B}$ computes $\mathbf{e} = \mathbf{H}(\mathbf{Y}, m_{12})$, and computes $\mathbf{mult}(\mathbf{params}, \mathbf{e}, \mathbf{b}) = m_{21}$.
- $\widehat{B}$ computes $\mathbf{add}(\mathbf{params}, \mathbf{y}, m_{12}) = m_{22}$.
- Computes $\mathbf{mult}(\mathbf{params}, m_{12}, m_{22}) = m_3$.
- Computes $\mathbf{enc}(\mathbf{params}, 1, m_3) = \sigma$.
- Computes $\mathbf{s}_\sigma \leftarrow \mathbf{ext}(\mathbf{params}, \mathbf{P}_{zt}, m_3)$.

After exchanging $\mathbf{X}$ and $\mathbf{Y}$, party $\widehat{A}$ and party $\widehat{B}$ can compute the same encoding $\mathbf{Encode}_{\widehat{A},\widehat{B}}(m_1, m_2, \mathbf{X}, \mathbf{Y})$.

The Security of Dual Challenge-Response Encoding Scheme. A dual encoding is dual challenge-response encoding by $\widehat{A}$ on message m_1, under challenge $\omega_{\widehat{A}} = \mathbf{add}(\mathbf{params}, \mathbf{Y}, \sigma_{\widehat{A}})$, and at the same time another dual encoding is dual challenge-response encoding by $\widehat{B}$ on message m_2, under challenge $\omega_{\widehat{B}} = \mathbf{add}(\mathbf{params}, \mathbf{X}, \sigma_{\widehat{B}})$. If there is no efficient algorithm that can win the Encoding forgery game, then we say this dual challenge-response encoding scheme is secure. We will change the

original encoding forgery game as follows: In **Step 2**, the query to $\widehat{B}$ is $(\mathbf{X}, m, m_1)$, while the encoding generated by $\widehat{B}$ is $(\mathbf{Y}, \mathbf{Encode}_{\widehat{B}}(\mathbf{Y}, m, \omega_{\widehat{B}}))$, where $\mathbf{Y} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{y}))$, $\mathbf{y} \leftarrow D_{\mathbb{Z}^n,\sigma}$, and $\omega_{\widehat{B}} = \mathbf{add}(\mathbf{params}, \mathbf{X}, \sigma_{\widehat{B}})$. $(\mathbf{Y}_0, m_0)$ satisfies the condition (b) in the definition of forgery $\mathcal{F}$, where message m_1 is chosen randomly by $\mathcal{F}$. For any random $\mathbf{A} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{a}))$, the challenge-response encoding scheme is secure, then we say the dual challenge-response encoding by $\widehat{B}$ is secure.

Theorem 3 Let $\widehat{A}$ and $\widehat{B}$ be two party with public key $\mathbf{A}$ and $\mathbf{B}$ respectively. Under the GCDH assumption, the dual challenge-response encoding by $\widehat{B}$ with respect to $\mathbf{A}$ is secure, even if the forgery $\mathcal{F}$ has obtained the private key $\mathbf{a}$ of $\widehat{A}$.

Proof. The dual challenge-response encoding by $\widehat{B}$ contains $\mathbf{d} = \mathbf{H}(\mathbf{X}_0, m_1)$, where m_1 is chosen by $\mathcal{F}$, autonomously. m_1 chosen by $\mathcal{F}$ before the repeat experiment is different from m_1 during the repeat experiment. There exist two different random values $\mathbf{d}$ and $\mathbf{d}'$, which corresponds to two encodings σ and σ'. In addition, we redefine the calculation procedure for $\mathbf{W}$ as follows (the inputs of the GCDH problem are $\mathbf{u}_1 = \mathbf{X}$, $\mathbf{u}_0 = \mathbf{B}$): (1) compute $\mathbf{mult}(\mathbf{params}, \mathbf{d}, \mathbf{a}) = w_{11}$; (2) compute $\mathbf{mult}(\mathbf{params}, \mathbf{e}, \mathbf{B}) = w_{12}$; (3) compute $\mathbf{add}(\mathbf{params}, \mathbf{Y}, w_{12}) = w_{13}$; (4) compute $\mathbf{mult}(\mathbf{params}, w_{12}, w_{13}) = w_{14}$; (5) compute $\sigma + (-w_{14})$; (6) compute $\mathbf{mult}(\mathbf{params}, \mathbf{d}', \mathbf{a}) = w_{21}$; (7) compute $\mathbf{mult}(\mathbf{params}, \mathbf{e}', \mathbf{B}) = w_{22}$; (8) compute $\mathbf{add}(\mathbf{params}, \mathbf{Y}, w_{22}) = w_{23}$; (9) compute $\mathbf{mult}(\mathbf{params}, w_{22}, w_{23}) = w_{24}$; (10) compute $\sigma' + (-w_{24}) = w_{25}$; (11) compute $w_{15} + (-w_{25}) = w_3$; (12) compute $\mathbf{mult}(\mathbf{params}, w_3, (\mathbf{e} - \mathbf{e}')^{-1})$.

The rest of the proof is similar to the proof of theorem 2.

VI. THE BASIC SECURITY OF OUR PROTOCOL

The session between $\widehat{A}$ and $\widehat{B}$ contains two level-one encodings $\mathbf{X} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$ and $\mathbf{Y} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{y}))$, and session key $\mathbf{s} \leftarrow \mathbf{ext}(\mathbf{params}, \mathbf{P}_{zt}, \pi)$, where $\pi = \mathbf{DEncode}_{\widehat{A},\widehat{B}}(m_1 = \widehat{B}, m_2 = \widehat{A}, \mathbf{X}, \mathbf{Y})$ is the dual encoding by $\widehat{A}$ or $\widehat{B}$. Below, we denote

$$\pi(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y}) := \mathbf{DEncode}_{\widehat{A},\widehat{B}}(m_1 = \widehat{B}, m_2 = \widehat{A}, \mathbf{X}, \mathbf{Y}).$$

In this section, we show the proposed protocol is secure in CK model.

Theorem 4 Under the GCDH assumption, our AKE protocol captures SK security.

Proof. This proof should use the following two lemmas.

Lemma 1 If $\widehat{A}$ and $\widehat{B}$ have finished the session, then they derive the same session key.

Proof. We analyze the calculations of session key for $\widehat{A}$ and $\widehat{B}$ particularly. For example, party $\widehat{A}$ samples $\mathbf{a} \leftarrow D_{\mathbb{Z}^n,\sigma n}$ and uses it as his own static secret key. Party $\widehat{A}$ computes $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$, $\mathbf{u}_1 = [\mathbf{yd}]_q = [\mathbf{da}/\mathbf{z}]_q$, $\mathbf{A} = [\mathbf{u}_1 + \sum_{i=1}^{m} r_i \mathbf{x}_i]_q = [\frac{\mathbf{da}+\sum_{i=1}^{m} r_i \mathbf{b}_i}{\mathbf{z}}]_q$, and uses $\mathbf{A}$ as his own static public key. $\widehat{A}$ samples $\mathbf{x} \leftarrow D_{\mathbb{Z}^n,\sigma n}$, and uses it as his ephemeral private key. $\widehat{A}$ computes $\mathbf{X} = [\frac{\mathbf{xa}+\sum_{i=1}^{m} r_i \mathbf{b}_i}{\mathbf{z}}]_q$,and uses $\mathbf{X}$ as his ephemeral public key.

After receiving $(\mathbf{Y}, \widehat{B})$, party $\widehat{A}$ computes $e = \mathbf{H}(\mathbf{Y}, \widehat{A}) \in \mathbb{Z}$, $\sigma_{\widehat{A}} = [\frac{e\cdot\mathbf{b}\cdot\mathbf{a}+e\cdot\sum_{i=1}^{m} r_i \mathbf{b}_i}{\mathbf{z}}]_q$, $\omega_{\widehat{A}} = [\frac{(e\cdot\mathbf{b}+\mathbf{y})\cdot\mathbf{a}+\sum_{i=1}^{m} r_i \mathbf{b}_i}{\mathbf{z}}]_q$, $d = \mathbf{H}(\mathbf{X}, \widehat{B}) \in \mathbb{Z}$, $\tau_{\widehat{A}} = d \cdot \mathbf{a}$, $\mathbf{v}_{\widehat{A}} = \mathbf{x} + d \cdot \mathbf{a}$, $\mathbf{z}_{\widehat{A}} = [\frac{(\mathbf{x}+d\cdot\mathbf{a})\cdot(e\cdot\mathbf{b}+\mathbf{y})\cdot\mathbf{a}+(\mathbf{x}+d\cdot\mathbf{a})\cdot(e+1)(\sum_i r_i \mathbf{b}_i)}{\mathbf{z}}]_q$, $\mathbf{P}_{zt} \cdot \mathbf{z}_{\widehat{A}} = [\frac{\mathbf{hz}}{\mathbf{g}} \cdot \frac{(\mathbf{x}+d\cdot\mathbf{a})\cdot(e\cdot\mathbf{b}+\mathbf{y})\cdot\mathbf{a}+(\mathbf{x}+d\cdot\mathbf{a})\cdot(e+1)(\sum_i r_i \mathbf{b}_i)}{\mathbf{z}}]_q$. Similarly, we can obtain $\mathbf{P}_{zt}\cdot\mathbf{z}_{\widehat{B}} = [\frac{\mathbf{hz}}{\mathbf{g}} \cdot \frac{(\mathbf{x}+d\cdot\mathbf{a})\cdot(e\cdot\mathbf{b}+\mathbf{y})\cdot\mathbf{a}+(\mathbf{y}+e\cdot\mathbf{b})\cdot(d+1)(\sum_i r_i \mathbf{b}_i)}{\mathbf{z}}]_q$.

Obviously, party $\widehat{A}$ and party $\widehat{B}$ encode the same level-one encoding of $(\mathbf{x} + d \cdot \mathbf{a}) \cdot (e \cdot \mathbf{b} + \mathbf{y}) \cdot \mathbf{a}$.

Lemma 2 Under the GCDH assumption, there is no PPT adversary which can distinguish the real session key of a unexposed session with a non-negligible probability.

Proof. Given a successful KE attacker $\mathcal{A}$, we can build an encoding forgery for dual encoding scheme. Combining with theorem ?, we can obtain the existence of a solver for the GCDH problem. But there is a contradiction between this existence and the hardness of the GCDH problem. We use $\pi(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$ to represent the session encoding of $(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$, while we use $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$ represent the test session. Accordingly, its session encoding is the test encoding. Observe that the session key of $(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y})$ is generated by computing $\mathbf{s} \leftarrow \mathbf{ext}(\mathbf{params}, \mathbf{P}_{zt}, \pi(\widehat{A}, \widehat{B}, \mathbf{X}, \mathbf{Y}))$. Then adversary $\mathcal{A}$ can distinguish between a real session key and a random key through the following two attacks.

1) **Forging attack.** The adversary computes a successful test encoding, and derives its session key.
2) **Key-replication attack.** The adversary enforce a non-testing session such that this session has the same session key as the test session. The adversary only query the session key with the same key without knowing the special value of the test encoding.

We will analyze the infeasibility of above-mentioned two attacks concretely in the following two subsection.

A. infeasibility of forging attack

Let $\mathcal{A}$ be the adversary. $\mathcal{A}$ acts as KE adversary, and outputs "fail" or $(\mathbf{sid}, \mathbf{guess})$, where $\mathbf{sid}$ is the session identifier, $\mathbf{guess}$ is the guess for the session encoding of $\mathbf{sid}$. We say $\mathcal{A}$ is successful, if $\mathcal{A}$ successfully outputs a correct guess for a session encoding of unexposed session. Let $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$ be the test session, and let $\pi = (\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$ be the test encoding. The generation of $\mathbf{Y}_0$ send to $\widehat{A}$ by $\mathcal{A}$ can fall into one of the following four cases:

1) **Case 1.** $\mathbf{Y}_0$ was never output by $\widehat{B}$ as its outgoing value in any of the sessions activated at $\widehat{B}$, or $\widehat{B}$ outputted $\mathbf{Y}_0$ as its outgoing value for certain session but it never computed the session key.
2) **Case 2.** $\mathbf{Y}_0$ was generated by $\widehat{B}$ in session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}_0)$.
3) **Case 3.** $\mathbf{Y}_0$ was generated by $\widehat{B}$ during a session $(\widehat{B}, \widehat{A}^*, \mathbf{Y}_0, \mathbf{X}^*)$ with $\widehat{A}^* \neq \widehat{A}$, and arbitrary $\mathbf{X}_0$.
4) **Case 4.** $\mathbf{Y}_0$ was generated by $\widehat{B}$ during a session $(\widehat{B}, \widehat{A}^*, \mathbf{Y}_0, \mathbf{X}^*)$ with $\widehat{A}^* = \widehat{A}$ and $\mathbf{X}^*$ and $\mathbf{X}_0$ are not the same encodings.

Since we have assumed that $\mathcal{A}$ succeeds with a non-negligible probability in forging attack, then at least one case of the above cases happens with a non-negligible probability, and satisfies this property: If $\mathcal{A}$ succeeds with a non-negligible probability in **Case i**, then $\mathcal{F}$ makes a successful forgery against **Case i**.

Forgery $\mathcal{F}$ for Case 1-Case 3. We first exhibit forgery $\mathcal{F}$ for the first three cases, which with a non-nebligible probability makes a successful forgery, given any one case of the first three cases. In the description below, we assume when the session is exposed, then $\mathcal{A}$ obtains the encoding values of session rather than the session key. $\mathcal{F}$'s actions are as follows:

1. The inputs of $\mathcal{F}$ contain challenge $\mathbf{X}_0$, public key $\mathbf{B}_0$, and the dual encoding by oracle $\widehat{B}$ under public key $\mathbf{B}_0$. $\mathcal{F}$ outputs either the encoding forgery of the dual encoding, or "fail".

2. $\mathcal{F}$ runs the protocol under $\mathcal{A}$.

3. When the session of $\widehat{B}$ is activated, then $\mathcal{F}$ uses its own encoding oracle to determine the actions of $\widehat{B}$. In particular, when $\mathcal{A}$ activates $\widehat{B}$'s session, either as initiator or responder, with peer $\widehat{A}$ and input $\mathbf{X}_0$, then $\mathcal{F}$ provides $\widehat{A}$ as the message to be encoded and $\mathbf{A}$ as the public key of $\widehat{A}$ with respect to which the dual encoding of $\widehat{B}$ is to generated to encoding oracle $\widehat{B}$. In response, $\mathcal{F}$ obtains the output message $\mathbf{Y}$ from $\widehat{B}$. Then, $\mathcal{F}$ gives $\mathbf{Y}$ to $\mathcal{A}$.

4. When $\mathcal{A}$ sends **state reveal** or **session key reveal**, then $\mathcal{F}$ returns session encoding to $\mathcal{A}$ accordingly.

5. Once all the parties are corrupted by $\mathcal{A}$ except $\widehat{B}$, then $\mathcal{F}$ provides its static secret key and local session state to $\mathcal{A}$. In this case, this corrupted party is completely controlled by $\mathcal{A}$.

6. When $\mathcal{A}$ activates a session of $\widehat{A}$, if the peer is not $\widehat{B}$, then $\mathcal{F}$ halts. Otherwise, $\mathcal{F}$ provides $\mathbf{X}_0$ to $\mathcal{A}$.

7. In the following situations, $\mathcal{F}$ halts: (1) $\mathcal{A}$ halts the test session of non-guessing session; (2) $\mathcal{A}$ corrupts $\widehat{A}$ or $\widehat{B}$; (3) $\mathcal{A}$ exposes the guess session via sending **state reveal** or **session key reveal**; (4) $\mathcal{A}$ exposes the matching session of the guess session via sending **state reveal** or **session key reveal**.

8. If $\mathcal{A}$ halts with the guess-session as its test session (the test session has identifier $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$) with a guess $\overline{\pi_0}$ for test encoding, then $\mathcal{F}$ outputs $(\widehat{A}, \mathbf{Y}_0, \overline{\pi_0})$ as a encoding forgery of $\widehat{B}$'s dual challenge-response encoding on message $m = \widehat{A}$. If $\mathcal{A}$ outputs "fail", then $\mathcal{F}$ outputs "fail".

Lemma 3 When any one of **Case 1**$-$**Case 3** holds, suppose that $\mathcal{A}$ makes a successful encoding forgery with a non-negligible probability, then the forgery mentioned above can makes a successful encoding forgery of dual encoding generated by $\widehat{B}$,with a non-negligible probability.

Proof. Let $\mathcal{A}$ be the adversary who makes a successful encoding forgery with a non-negligible probability in any one of **Case 1** $-$ **Case 3**, then there exists an adversary controlled under $\mathcal{F}$ who generates a successful encoding forgery of dual encoding generated by $\widehat{B}$,with a non-negligible probability. By fixing each random value for $\mathcal{A}$, each party, and random oracle $\mathbf{H}(\cdot)$, we get the determinate execution of the protocol. These determinate values completely determine $\mathcal{A}$'s actions and its views. $\mathcal{F}$ provides all of

the random coins for $\mathcal{A}$ and all of the parties in the protocol except for party $\widehat{B}$. We say a run of $\mathcal{A}$ controlled under $\mathcal{F}$ is perfect, if it is identical with the real run of $\mathcal{A}$ under the same coins set by $\mathcal{F}$ and by $\widehat{B}$. Firstly, consider the simulation by $\mathcal{F}$ with respect to uncorrupted parties rather than $\widehat{B}$. Since $\mathcal{F}$ is aware of all the information of these parties except for $\widehat{B}$, it runs all their actions identically with in a real interaction with $\mathcal{A}$.

But there exists a possible deviation. Namely, when $\mathcal{A}$ activates the guess-session at $\widehat{A}$, $\mathcal{F}$ uses $\mathbf{X}_0$ as the output value, instead of $\mathbf{X} \leftarrow \mathbf{reRand}(1, \mathbf{enc}(1, \mathbf{x}))$. The distribution of $\mathbf{X}_0$ and that of $\mathbf{X}$ is statistically indistinguishable. In particular, if $\mathcal{F}$ does not halt the guess-session, then $\mathcal{F}$ fails to any action to obtain the corresponding level-zero encoding. Since $\mathcal{F}$ knows nothing about the private key $\mathbf{b}$ of $\widehat{B}$, $\mathcal{F}$ uses the encoding oracle as a black box. Since the corruption of $\widehat{B}$ will lead to forgery termination. Thus, $\mathcal{F}$ only need to simulate $\widehat{B}$'s actions with respect to the session initiation and the exposure of session key. According to the construction, every time $\mathcal{A}$ under $\mathcal{F}$ completes and outputs the guess $\overline{\pi_0}$ for the test encoding $\pi_0 = \pi(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$, $\mathcal{F}$ outputs an encoding forgery against $\widehat{B}$'s encoding. Specially, on input a challenge $\mathbf{x}_0$, $\mathcal{F}$ outputs the forgery $(m = \widehat{A}, \mathbf{Y}_0, \overline{\pi_0})$. If $\overline{\pi_0}$ and π_0 are encodings of the same coset, then this forgery is correct. What should be proven is that $\mathbf{Y}_0$ is not an encoding of $\mathbf{0} + I$, and $(m = \widehat{A}, \mathbf{Y} = \mathbf{Y}_0)$ never appeared in the encoding generated by $\widehat{B}$, where $\widehat{B}$ is invoked by $\mathcal{F}$. Suppose that $\mathbf{Y}_0$ is an encoding of $\mathbf{0} + I$, then $\widehat{A}$ would have rejected to take $\mathbf{Y}_0$ as the incoming message of the guess-session. Assume $\overline{\pi_0} = \pi_0$, we verify its validity in any case of the following two cases **C1,C2**.

C1. $\mathbf{Y}_0$ never appeared in the encodings issued by $\widehat{B}$, thus $(m = \widehat{A}, \mathbf{Y}_0, \overline{\pi_0})$ is a valid forgery.

C2. $\mathbf{Y}_0$ appeared in the session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}_0)$, which has never been queried by $\mathcal{A}$. $\mathcal{F}$ never queries the session encoding corresponding to a session of $\widehat{B}$, except if this session has been queried by $\mathcal{A}$ via state reveal or session-key reveal. Since $\mathbf{Y}_0$ only appears in above mentioned session, $(m = \widehat{A}, \mathbf{Y}_0, \overline{\pi_0})$ is valid.

In conclusion, if $\mathcal{A}$ successfully outputs a valid guess of the test encoding with a non-negligible probability, then $\mathcal{F}$ successfully outputs a valid forgery against the $\mathbf{DEncode}$ generated by $\widehat{B}$.

Forgery $\mathcal{F}'$ for Case 4. If there is adversary $\mathcal{A}$ who initiates a successful forgery attack with with a non-negligible probability, then we define a successful forgery $\mathcal{F}'$ against the $\mathbf{DEncode}$ generated by $\widehat{B}$. When $\mathcal{A}$ successfully outputs a valid guess π_0 of the encoding of the test session $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$, and $\mathbf{Y}_0$ is generated by $\widehat{B}$ in another session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$ with $\mathbf{X}^* \neq \mathbf{X}_0$. With this list below we will construct the forgery $\mathcal{F}'$ as follows:

1) In addition to choosing random values i, j, t as $\mathcal{F}$ does, $\mathcal{F}'$ also selects $l \in_{\mathcal{R}} \{0, 1\}$. We use $\widehat{A}$ and $\widehat{B}$ to denote P_i and P_j respectively. The choices of i, j, t represent that which test session will be chosen to act as the t-th session by $\mathcal{F}'$ for $\mathcal{A}$. The choice of l represents the l-th session activated at $\widehat{B}$.
2) $\mathcal{F}'$ halts, if the peer of the l-th session at $\widehat{B}$ is not $\widehat{A}$ or it has $\mathbf{X}^* = \mathbf{X}_0$, and the incoming value provided by $\mathcal{A}$ to $\widehat{A}$ in its t-th session is different to the output value $\mathbf{Y}_0$ in $\widehat{B}$'s l-th session.
3) The subsequent actions of $\mathcal{F}'$ is identical to $\mathcal{F}$'s, except for in the l-th session activation of $\widehat{B}$, $\mathcal{F}'$ chooses a random encoding $\mathbf{Y}_0$, and feeds it to $\mathcal{A}$. If at any time $\mathcal{A}$ queries session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$ via **state reveal** or **session-key reveal**, then $\mathcal{F}'$ selects a random bit string from $\{0, 1\}^{n(\log_2(q/4)-\lambda)}$, and uses this random bit string to answer the query from $\mathcal{A}$.
4) If $\mathcal{A}$ halts without querying session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$, then $\mathcal{F}'$ halts with the same results as $\mathcal{A}$ outputs. That is, if $\mathcal{A}$ fails, then $\mathcal{F}'$ fails. If $\mathcal{A}$ outputs a guess of $\pi(\widehat{A}, \widehat{B}, \mathbf{Y}_0, \mathbf{X}^*)$, then $\mathcal{F}'$ outputs $(\widehat{A}, \mathbf{Y}_0, \overline{\pi_0})$ as forgery of $\widehat{B}$'s encoding on message $m = \widehat{A}$ under challenge $\mathbf{X}_0$.
5) If $\mathcal{A}$ halts after querying $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$, then $\mathcal{F}'$ selects a random bit b' and executes:
    a) When $b = 0$, $\mathcal{F}'$ halts. Its output is decided by **Step 4**.
    b) When $b = 1$, **recoiling step:** $\mathcal{F}'$ recoils $\mathcal{A}$ to its state at the time $\mathcal{A}$ queries session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$. In this case, $\mathcal{F}'$ randomly chooses one answer c of queries to $\mathbf{H}(\cdot)$ by $\mathcal{A}$ instead of answering a random value, and uses $\mathbf{H}(c)$ to response the query to session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$.

Lemma 4 Assume $\mathcal{A}$ outputs a successful guess of the test encoding with a non-negligible probability in **Case 4**, then $\mathcal{F}'$ makes a successful encoding forgery against $\widehat{B}$'s **DEncode** with a non-negligible

probability.

Proof. We say a run of $\mathcal{A}$ is **type-Case 4**, if $\mathcal{A}$ outputs a guess of the session encoding of the unexposed session $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$, where $\mathbf{Y}_0$ is generated in session $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$ with $\mathbf{X}^* \neq \mathbf{X}_0$. Assume $\mathcal{A}$ runs **type-Case 4** successfully with a non-negligible probability. We use $\mathbf{s}_0$ to denote the test session $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$, and let $\mathbf{s}^*$ denote $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$. π_0 denotes the test encoding of $\pi = (\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$. π^* denotes the test encoding of $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$. π_0. We say $\mathcal{A}$ queries $\mathbf{s}^*$ if $\mathcal{A}$ queries $\mathbf{s}^*$ via **state-reveal** or **session-key reveal**. We differentiate the runs of **type-Case 4** into the following three classes:

1) $\mathcal{A}$ does not query session $\mathbf{s}^*$.
2) $\mathcal{A}$ queries $\mathbf{s}^*$ but not π^*.
3) $\mathcal{A}$ queries $\mathbf{s}^*$ and π^*.

In the non-aborting run of $\mathcal{F}'$, the simulations dominated by $\mathcal{F}'$ is perfect, except for the response to the query to $\mathbf{s}^*$ if the query is sent by $\mathcal{A}$, and the subsequent recoiling step that is performed by $\mathcal{A}$ under $\mathcal{F}$. In runs in which $\mathcal{A}$ queries $\mathbf{s}^*$ before it queries $\mathbf{H}(\cdot)$ on π^*, the probability that $\mathcal{A}$ queries $\mathbf{H}(\cdot)$ on π^* is identical whether the response to $\mathbf{s}^*$ was the real $\mathbf{H}(\pi^*)$ or a random value. In class one, the run of $\mathcal{A}$ under $\mathcal{F}'$ is identical to the real run of $\mathcal{A}$. If $b = 0$, then $\mathcal{A}$ under $\mathcal{F}'$ is identical to the real run. Since as long as $\mathcal{A}$ does not query π^*, then the random value answered by $\mathcal{F}'$ to the query to $\mathbf{s}^*$ is undistinguishable with any other random value. In class two, when $b = 1$, $\mathcal{F}'$ just chooses to query $c = \pi^*$. In sum, the run of $\mathcal{A}$ under $\mathcal{F}'$ is identical to the real run, and happens with a non-negligible probability.

When $\mathcal{A}$ under $\mathcal{F}'$ outputs a correct guess π_0 of session $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$, $\mathcal{F}'$ outputs a valid encoding forgery of encoding of $(m = \widehat{A}, \mathbf{Y}_0, \pi_0)$ by $\widehat{B}$ with challenge $\mathbf{X}_0$. Since $\mathbf{Y}_0$ is not an encoding of $\mathbf{0} + I$, and $(m = \widehat{A}, \mathbf{Y}_0, \mathbf{X}_0)$ did not appear in the encodings issued by $\widehat{B}$. Suppose that $\mathbf{Y}_0$ is an encoding of $\mathbf{0} + I$, then $\widehat{A}$ rejects to accept it as the incoming value in the **guess**-session. $\widehat{B}$ uses $\mathbf{Y}_0$ only in session $\mathbf{s}^* = (\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$. But $\mathcal{F}'$ did not query this session. In fact, even if $\mathcal{A}$ queries this session, $\mathcal{F}'$ answers a random value or a previous output value. Thus, $\mathcal{A}$ under $\mathcal{F}'$ outputs valid encoding forgery against $\widehat{B}$'s encoding.

B. infeasibility of key-replication attack

Lemma 5 If there is an efficient adversary $\mathcal{A}$ which can initiate a **key-replication** attack with a non-negligible probability, then $\mathcal{F}$ or $\mathcal{F}'$ can output a successful encoding forgery against the **DEncode** scheme with a non-negligible probability.

Proof. Assume $\mathcal{A}$ initiates a successful **key-replication** attack against session $(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$. That is, $\mathcal{A}$ builds a session $\mathbf{s}' = (\widehat{A}', \widehat{B}', \mathbf{X}', \mathbf{Y}')$, which has the same session key as session $\mathbf{s}$. Consider **Case 1-Case 4** which have to do with the generation of $\mathbf{Y}_0$ in session $\mathbf{s}$. In at least one case of these four cases, $\mathcal{A}$ will initiate a **key-replication** attack with a non-negligible probability. Suppose, this non-negligible probability of success holds for any of **Case 1-Case 3**, and consider $\mathcal{F}$ constructed above and its interaction with the key-replication attacker $\mathcal{A}$. $\mathcal{F}$ provides the session encoding of the unexposed session to $\mathcal{A}$. Therefore, if $\mathcal{A}$ initiates a successful **key-replication** attack, then $\mathcal{A}$ can obtain the test encoding $\pi(\widehat{A}, \widehat{B}, \mathbf{X}_0, \mathbf{Y}_0)$ without exposing the test session or its matching session. In this case, $\mathcal{F}$ can successfully output a valid encoding forgery of the encoding generated by $\widehat{B}$.

In **Case 4**, $\mathcal{F}'$ provides the test encoding rather than $(\widehat{B}, \widehat{A}, \mathbf{Y}_0, \mathbf{X}^*)$ to $\mathcal{A}$. Similarly, if $\mathcal{A}$ initiates a successful **key-replication** attack, then $\mathcal{F}'$ can successfully output a valid encoding forgery of the encoding generated by $\widehat{B}$.

VII. CONCLUSIONS

In this paper, we propose an efficient and practical authenticated key exchange (AKE) protocol from ideal lattices. Compared with the current lattice-based AKE schemes, our protocol not only possesses some graceful characteristics and innovations of HMQV more naturally, but also enjoys many excellent properties of lattice-based cryptography.

VIII. ACKNOWLEDGEMENTS

This work is supported by National Science Foundation of China under grant (No. 61072140, 61373171), Base for Introducing Talents of Discipline to Universities (No. B08038), the Specialized Research Fund

for the Doctoral Program of Higher Education of China under grant (No. 20100203110003.), The National Development Foundation for Cryptological Research (MMJJ201401003).

REFERENCES

[1] W. Diffie and M. Hellman. New directions in cryptography. Information Theory, IEEE Transactions on 22(6):644-654, 1976.

[2] M. Bellare and P. Rogaway. Entity authentication and key distribution. volume 773 of Lecture Notes in Computer Science, pages 232-249. Springer Berlin Heidelberg, 1994.

[3] A. Menezes, M. Qu, and S. Vanstone, Some new key agreement protocols providing mutual implicit authentication, SecondWorkshop on Selected Areas in Cryptography, 1995.

[4] L. Law, A. Menezes, M. Qu, J. Solinas, and S. Vanstone, An efficient Protocol for Authenticated Key Agreement, Designs, Codes and Cryptography, 119-134, 2003.

[5] R. Canetti and H. Krawczyk. Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In B. Pfitzmann, editor, Advances in Cryptology-EUROCRYPT 2001.

[6] H. Krawczyk. HMQV: A High-Performance Secure Diffie-Hellman Protocol. In V. Shoup, editor, Advances in Cryptology-CRYPTO 2005, volume 3621 of Lecture Notes in Computer Science, pages 546-566. Springer Berlin Heidelberg, 2005.

[7] Fujioka. A, Suzuki. K, Xagawa. K, Yoneyama. K. Stronger Secure Authenticated Key Exchange from Factoring, Codes, and Lattices. In Fischlin [22], pages 467-484.

[8] M. Ajtai. Generating Hard Instances of The Short Basis Problem. In ICALP, pages 1-9, 1999.

[9] J. Alwen and C. Peikert. Generating Shorter Bases for Hard Random Lattices. Theory of Computing Systems, pages 535-553, 2011.

[10] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for Hard Lattices and New Cryptographic Constructions. In STOC, pages 197C206, 2008.

[11] V. Lyubashevsky, C. Peikert, and O. Regev. On Ideal Lattices and Learning With Errors over Rings. In EUROCRYPT, pages 1-23, 2010.

[12] Fujioka. A, Suzuki. K, Xagawa. K, Yoneyama. K. Practical and Post-Quantum Authenticated Key Exchange from One-Way Secure Key Encapsulation Mechanism, ASIA CCS'13, May 8-10, 2013, Hangzhou, China. Copyright 2013 ACM 978-1-4503-1767-2/13/05.

[13] Daniele Micciancio, Chris Peikert, Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller, Cryptology ePrint Archive, Report 2011/501 (2011).

[14] Damien Stehl*é*, Ron Steinfeld, Keisuke Tanaka, Keita Xagawa. Efficient Public Key Encryption Based on Ideal Lattices (Extended Abstract). Advances in Cryptology-ASIACRYPT 2009; 617-635.

[15] D. Micciancio and O. Regev. Worst-case to Average-case Reductions Based on Gaussian Measures. SIAMJ. Comput., 37:267-302, April 2007.

[16] Léo Ducas, Phong Q. Nguyen, Faster Gaussian Lattice Sampling using Lazzy Floating-Point Arithmetic, Advances in Cryptology-ASIACRYPT 2012: 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6 2012.

[17] D. Micciancio and O. Regev. Worst-case to Average-case Reductions Based on Gaussian Measures. SIAM J. Comput., 37:267-302, April 2007.

[18] D. Micciancio and O. Regev. Lattice-based Cryptography. In D. Bernstein, J. Buchmann, and E. Dahmen, editors, Post-Quantum Cryptography, pages 147-191. Springer Berlin Heidelberg, 2009.

[19] C. Peikert. Lattice Cryptography for the Internet. Cryptology ePrint Archive, Report 2014/070, 2014.

[20] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D.Stehle. Classical Hardness of Learning with Errors. In Proc. of 45th STOC, pages 575-584. ACM, 2013.

[21] D. Micciancio and C. Peikert. Hardness of SIS and LWE with Small Parameters. In Proc. CRYPTO '13, volume 8042 of Lecture Notes in Computer Science, pages 21-39. Springer, 2013.

[22] C. Peikert. An Efficient and Parallel Gaussian Sampler for Lattices. In Proc. of Crypto '10, LNCS 6223, pages 80-97. Spinger-Verlag, 2010.

[23] Bresson E, Chevassut O, Pointcheval D. Provably Authenticated Group Diffie-Hallman Key Exchange. CCS'01: Proceedings of the 8th ACM conference on Computer and Communications Security. Philadelpia:ACM, 2001.

[24] Katz J,Yung M.Scalable Protocols for Authenticated Group Key Exchange. Advance in Cryptology-Crypto 2003. Springer.

[25] Bohli J, Vasco M, Steinwandt R. Secure Group Key Establishment Revisited. International Journal of Informational Security. 2007. 6(4):243-254.

[26] Katz J, Shin J. Modeling Insider Attacks on Group Key Exchange Protocols. Proceedings of the 12th ACM Conference on Computer and Communication Security-CCS'05. ACM, 2005:180-189.

[27] Manulis M. Survey on Security Requirements and Models for Group Key Exchange. http://eprint.iacr.org /2006/388.

[28] Bresson E, Manulis M. Malicious Participants in Group Key Exchange: Key Control and Contributiveness in the Shadow of Trust. Proceedings of the ATC'07 of LNCS, Springer, 2007.

[29] Bresson E, Manulis M. Securing Group Key Exchange against Strong Corruptions and Key Registration Attacks. UACT, 2008.

[30] Ran Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. Cryptology ePrint Archive, Report 2000/067, 2000. http://eprint.iacr.org/, Version updated on 13 Dec 2005.

[31] X. L. Jintai Ding. A Simple Provably Secure Key Exchange Scheme Based on The Learning With Errors Problem. Cryptology ePrint Archive, Report 2012/688, 2012.

[32] V. Lyubashevsky, C. Peikert, and O. Regev. A Toolkit for Ring-LWE Cryptography. In T. Johansson and P. Nguyen, editors, Advances in Cryptology, 2013.