Identification Schemes from Key Encapsulation Mechanisms

Hiroaki Anada and Seiko Arita

Institute of Information Security, Yokohama, Japan
hiroaki.anada@gmail.com, arita@iisec.ac.jp

February 5, 2011

Abstract. We propose a generic way for deriving an identification (ID) scheme secure against concurrent man-in-the-middle attacks from a key encapsulation mechanism (KEM) secure against chosen ciphertext attacks on one-wayness (one-way-CCA). Then we give a concrete one-way-CCA secure KEM based on the Computational Diffie-Hellman (CDH) assumption. In that construction, the Twin Diffie-Hellman technique of Cash-Kiltz-Shoup is essentially employed. We compare efficiency of the ID scheme derived from our KEM with previously known ID schemes and KEMs. It turns out that our KEM-based ID scheme is faster in one exponentiation than the currently most efficient one derived from the Hanaoka-Kurosawa KEM, whose security is based on the same CDH assumption.
Keywords: identification scheme, key encapsulation mechanism, concurrent man-in-the-middle attack, the computational Diffie-Hellman assumption.

1 Introduction

An identification (ID) scheme enables a prover to convince a verifier that the prover is certainly itself by proving that it knows some secret information. In the public key framework, a prover holds a secret key and a verifier refers to a matching public key. They interact for some rounds doing necessary computations until the verifier feels certain that the prover has the secret key. The secret key is never revealed directly but hidden in messages through those computations by such a technique of honest verifier zero-knowledge.

Historically, there have been two types of ID schemes. One is challenge-and-response type obtained in a natural way from encryption schemes or signature schemes, and another is the Σ-protocol type [7] which is a kind of proofs of knowledge [12, 4] consisting of 3-round interaction. Most of known traditional ID schemes, such as the Schnorr scheme [24] and the Guillou-Quisquater (GQ) scheme [13], are the Σ-protocol type because they are faster than challenge-and-response type.

Now in the Internet environment where everyone is involved, attacks on ID schemes have become fairly strong. One of the strongest is *concurrent man-in-the-middle attacks*. In concurrent man-in-the-middle setting, an adversary stands between a verifier and prover clones which the adversary invokes. Interacting in some cheating way, the adversary collects information of the secret key from the prover clones, while the adversary interacts with the verifier simultaneously trying to impersonate the prover.

Unfortunately, the Schnorr scheme and the GQ scheme are not secure against concurrent man-in-the-middle attacks, hence there have been significant efforts to make ID

schemes have tolerance against such concurrent man-in-the-middle attacks based on the Σ-protocol. For example, Katz [16] made an ID scheme of non-malleable proof of knowledge. But the security model is with timing constraint, not against full concurrent man-in-the-middle attacks. Moreover, the protocol utilizes the so-called OR-Proof technique, so it is rather complicated. Gennaro [11] constructed an ID scheme of (fully) concurrently non-malleable proof of knowledge employing a multi-trapdoor commitment. But it is not so fast as challenge-and-response ID scheme obtained, for instance, from the Cramer-Shoup encryption scheme [8]. Moreover, the security is based on the strong type assumptions (the Strong Diffie-Hellman (SDH) assumption or the Strong RSA assumption).

One of the reason why it is so difficult to construct an ID scheme secure against concurrent man-in-the-middle attacks seems that we are rooted in the category of Σ-protocols. Let us remember that challenge-and-response ID schemes obtained from IND-CCA secure encryption schemes (see, for example, [8]) and EUF-CMA secure signature schemes (see, for example, [2]) are already secure as ID schemes against concurrent man-in-the-middle attacks.

1.1 Our Contribution

In the notion of encryption scheme, key encapsulation mechanism (KEM) is the foundational concept for hybrid construction with data encryption mechanism. As a first contribution in this paper, we propose to use KEM as ID scheme analogous to the usage of encryption scheme. That is, given a KEM, we derive a challenge-and-response ID scheme as follows. A verifier of a KEM-based ID scheme makes a pair of random key and its ciphertext using a public key, and send the ciphertext as a challenge to the prover having the matching secret key. The prover decapsulates the ciphertext and returns the result as a response. The verifier checks whether or not the response is equal to the random key. Although this is a straightforward conversion, it has never been mentioned in the literature, to the best of author's knowledge.

As a generic property, KEM-based ID scheme has an advantage over (non-hybrid) encryption-based ID scheme. That is, KEM encrypts *random* strings and may generate them by itself, while encryption scheme needs to encrypt *any* strings given as input. Consequently, KEM-based ID scheme has a possibility to have simpler and more efficient protocol than encryption-based ID scheme.

In addition, as we will show in Section 3, KEM only has to be one-way-CCA secure for derived ID scheme to have security against concurrent man-in-the-middle attacks (cMiM security). In other words, IND-CCA security, which is stronger than one-way-CCA security, is rather excessive for deriving cMiM secure ID scheme. Nonetheless by this time, most known encryption schemes and KEMs have been designed to possess IND-CCA security (because the purpose is not to make up ID schemes, of course).

Hence there arises a need to provide one-way-CCA secure KEMs. As a second contribution, we give a concrete, discrete logarithm-based one-way-CCA secure KEM. It is true that there have already been a few one-way-CCA secure KEMs in discrete logarithm setting. In contrast to those KEMs, the feature of our KEM is that it needs the smallest amount of computational cost while its security is based on the Computational Diffie-Hellman (CDH) assumption which is weaker than the Decisional Diffie-Hellman (DDH) assumption or Gap-

CDH assumption (see [21] for these assumptions). That feature is achieved by applying the Twin Diffie-Hellman technique [6] to Anada-Arita's scheme [1] whose security is based on the Gap-CDH assumption. [1].

Finally, we point out that the prover in our generic construction of ID scheme from KEM is deterministic. Therefore, the derived ID scheme is prover-resettable [3]. Moreover, they are also verifier-resettable because they consists of 2-round interaction. This is a remarkable property of the generic construction above. As is discussed by Yilek [27], resettable security is crucially helpful, for example, for virtual machine service in the Cloud Computing.

1.2 Related Works

Recently, independently of us, Fujisaki [10] pointed out a fact similar to our generic construction above (that is, the conversion from one-way-CCA secure KEM to cMiM secure ID scheme).

As for concrete constructions, the IND-CCA secure KEM of Shoup [25], which is naturally a one-way-CCA secure KEM, performs comparably efficiently even now, while its security is based on the DDH assumption. Hanaoka-Kurosawa [15] gave a one-way-CCA secure KEM based on the CDH assumption which is weaker than the DDH assumption. It is directly comparable with our KEM and our KEM needs less computational amount in one exponentiation for encapsulation than Hanaoka-Kurosawa's KEM. Both Shoup's KEM and Hanaoka-Kurosawa's KEM are intended for the hybrid encryption construction, while the one-way-CCA KEM of Anada-Arita [1] is intended directly for an ID scheme. It performs better than Shoup's KEM based on the Gap-CDH assumption in its security proof. The Twin Diffie-Hellman technique enables us to remove that "Gap" assumption to lead our one-way-CCA KEM.

1.3 Organization of the Paper

In Section 2, we fix some notations, briefly review the notion of ID scheme, KEM and computational hardness assumption. In Section 3, we propose a generic way for deriving a cMiM secure ID scheme from a one-way-CCA secure KEM. In Section 4, we construct a one-way-CCA secure KEM by the Twin Diffie-Hellman technique. In Section 5, we compare our KEM or ID scheme with previously known KEMs or ID schemes. In Section 6, we conclude our work.

2 Preliminaries

The security parameter is denoted k. On input 1^k, a PPT algorithm $\mathtt{Grp}$ runs and outputs (q, g), where q is a prime of length k and g is a generator of a multiplicative cyclic group G_q of order q. $\mathtt{Grp}$ specifies elements and group operations of G_q. The ring of exponent domain of G_q, which consists of integers from 0 to $q-1$ with modulo q operation, is denoted $\mathbf{Z}_q$.

When an algorithm A on input a outputs z we denote it as $z \leftarrow A(a)$. When A on input a and B on input b interact and B outputs z we denote it as $z \leftarrow \langle A(a), B(b)\rangle$. When A has

[1] The strategy to apply the Twin DH technique to the scheme of [1] was kindly suggested to authors by Prof. Kiltz. at ProvSec 2010 [18].

oracle-access to $\mathcal{O}$ we denote it as $A^{\mathcal{O}}$. When A has concurrent oracle-access to n oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$ we denote it as $A^{\mathcal{O}_1|\cdots|\mathcal{O}_n}$. Here "concurrent" means that A accesses to oracles in arbitrarily interleaved order of messages.

A probability of an event X is denoted Pr[X]. A probability of an event X on conditions $\mathrm{Y}_1, \ldots, \mathrm{Y}_m$ is denoted $\Pr[\mathrm{Y}_1; \cdots; \mathrm{Y}_m : \mathrm{X}]$.

2.1 Identification Scheme

An *identification scheme* $\mathtt{ID}$ is a triple of PPT algorithms (K, P, V). K is a key generator which outputs a pair of a public key and a matching secret key $(\mathtt{pk}, \mathtt{sk})$ on input 1^k. P and V implement a prover and a verifier strategy, respectively. We require ID to satisfy the completeness condition that boolean decision by $\mathtt{V}(\mathtt{pk})$ after interaction with $\mathtt{P}(\mathtt{sk})$ is True with probability one. We say that $\mathtt{V}(\mathtt{pk})$ *accepts* if its boolean decision is True.

Concurrent Man-in-the-Middle Attack on Identification Scheme [3, 5] The aim of an adversary $\mathcal{A}$ that attacks on an ID scheme ID is impersonation. We say that $\mathcal{A}$ *wins* when $\mathcal{A}(\mathtt{pk})$ succeeds in making $\mathtt{V}(\mathtt{pk})$ accept.

An adversary $\mathcal{A}$ performs concurrent man-in-the-middle (cMiM) attack in the following way.

$$\begin{array}{l}\mathbf{Experiment}^{\text{imp-cmim}}_{\mathcal{A},\mathtt{ID}}(1^k)\\ \quad (\mathtt{pk},\mathtt{sk}) \leftarrow \mathtt{K}(1^k), \text{decision} \leftarrow \langle \mathcal{A}^{\mathtt{P}_1(\mathtt{sk})|\cdots|\mathtt{P}_n(\mathtt{sk})}(\mathtt{pk}), \mathtt{V}(\mathtt{pk})\rangle\\ \quad \text{If decision} = 1 \wedge \pi^* \notin \{\pi_i\}_{i=1}^n \text{ then return Win else return Lose.}\end{array}$$

In the above experiment, we denoted a transcript of interaction between $\mathtt{P}_i(\mathtt{sk})$ and $\mathcal{A}(\mathtt{pk})$ as π_i and a transcript between $\mathcal{A}(\mathtt{pk})$ and $\mathtt{V}(\mathtt{pk})$ as π^*. As a rule, man-in-the-middle adversary $\mathcal{A}$ is prohibited from relaying a transcript of a whole interaction with some prover clone to the verifier $\mathtt{V}(\mathtt{pk})$, as is described $\pi^* \notin \{\pi_i\}_{i=1}^n$ in the experiment above. This is a standard and natural constraint to keep the attack meaningful.

We define *$\mathcal{A}$'s imp-cMiM advantage over* $\mathtt{ID}$ as:

$$\mathbf{Adv}^{\text{imp-cmim}}_{\mathcal{A},\mathtt{ID}}(k) \stackrel{\text{def}}{=} \Pr[\mathbf{Experiment}^{\text{imp-cmim}}_{\mathcal{A},\mathtt{ID}}(1^k) \text{ returns Win}].$$

We say that an ID is secure against concurrent man-in-the-middle attacks (cMiM secure, for short) if, for any PPT algorithm $\mathcal{A}$, $\mathbf{Adv}^{\text{imp-cmim}}_{\mathcal{A},\mathtt{ID}}(k)$ is negligible in k.

Suppose that the adversary $\mathcal{A}$ consists of two algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. The following experiment is called a 2-phase concurrent attack.

$$\begin{array}{l}\mathbf{Experiment}^{\text{imp-2pc}}_{\mathcal{A},\mathtt{ID}}(1^k)\\ \quad (\mathtt{pk},\mathtt{sk}) \leftarrow \mathtt{K}(1^k), st \leftarrow \mathcal{A}_1^{\mathtt{P}_1(\mathtt{sk})|\cdots|\mathtt{P}_n(\mathtt{sk})}(\mathtt{pk}), \text{decision} \leftarrow \langle \mathcal{A}_2(st), \mathtt{V}(\mathtt{pk})\rangle\\ \quad \text{If decision} = 1 \text{ then return Win else return Lose.}\end{array}$$

2-phase concurrent attack is weaker model than cMiM attack because of the constraint that the learning phase of $\mathcal{A}_1$ is limited to before the impersonation phase of $\mathcal{A}_2$.

2-phase concurrent attacks and cMiM attacks are classified to active attacks. On the contrary, there is a passive attack described below. Let us denote a transcript of a whole interaction between $\mathtt{P}(\mathtt{sk})$ and $\mathtt{V}(\mathtt{pk})$ as $\pi = |\langle \mathtt{P}(\mathtt{sk}), \mathtt{V}(\mathtt{pk})\rangle|$.

$$
\begin{array}{l}
\mathbf{Experiment}^{\text{imp-pa}}_{\mathcal{A},\mathtt{ID}}(1^k) \\
\quad (\mathtt{pk}, \mathtt{sk}) \leftarrow \mathtt{K}(1^k) \\
\quad \text{If } \mathcal{A}_1(\mathtt{pk}) \text{ makes a query, reply } \pi_i \leftarrow |\langle \mathtt{P}(\mathtt{sk}), \mathtt{V}(\mathtt{pk})\rangle| \\
\quad st \leftarrow \mathcal{A}_1(\{\pi_i\}_{i=1}^n), \text{decision} \leftarrow \langle \mathcal{A}_2(st), \mathtt{V}(\mathtt{pk})\rangle \\
\quad \text{If decision} = 1 \text{ then return } \textsc{Win} \text{ else return } \textsc{Lose}.
\end{array}
$$

Passive attack is weaker model than 2-phase concurrent attack because of the constraint that $\mathcal{A}$ cannot choose messages in the learning phase.

2.2 Key Encapsulation Mechanism

A *key encapsulation mechanism (KEM)* $\mathtt{KEM}$ is a triple of PPT algorithms ($\mathtt{K}$, $\mathtt{Enc}$, $\mathtt{Dec}$). $\mathtt{K}$ is a key generator which outputs a pair of a public key and a matching secret key $(\mathtt{pk}, \mathtt{sk})$ on input 1^k. $\mathtt{Enc}$ is an encapsulation algorithm which, on input $\mathtt{pk}$, outputs a pair (K, ψ), where K is a random string called a *random key* and ψ is a *ciphertext* of K. $\mathtt{Dec}$ is a decapsulation algorithm which, on input $(\mathtt{sk}, \psi)$, outputs the decapsulation $\widehat{K}$ of ψ. We require $\mathtt{KEM}$ to satisfy the completeness condition that the decapsulation $\widehat{K}$ of a consistently generated ciphertext ψ by $\mathtt{Enc}$ is equal to the original random key K with probability one.

Adaptive Chosen Ciphertext Attack on One-Wayness of KEM [22, 15] An adversary $\mathcal{A}$ on a KEM performs adaptive chosen ciphertext attack on one-wayness of a KEM (one-way-CCA, for short) in the following way.

$$
\begin{array}{l}
\mathbf{Experiment}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM}}(1^k) \\
\quad (\mathtt{pk}, \mathtt{sk}) \leftarrow \mathtt{K}(1^k), (K^*, \psi^*) \leftarrow \mathtt{Enc}(\mathtt{pk}), \widehat{K^*} \leftarrow \mathcal{A}^{\mathcal{DEC}(\mathtt{sk},\cdot)}(\mathtt{pk}, \psi^*) \\
\quad \text{If } \widehat{K^*} = K^* \wedge \psi^* \notin \{\psi_i\}_{i=1}^{q_{dec}} \text{ then return } \textsc{Win} \text{ else return } \textsc{Lose}.
\end{array}
$$

In the above experiment, $\psi_i, i = 1, \ldots, q_{dec}$ mean ciphertexts for which $\mathcal{A}$ queries its decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ for the answer. Here the number q_{dec} of queries is polynomially many in k. The challenge ciphertext ψ^* itself must not be queried to $\mathcal{DEC}(\mathtt{sk}, \cdot)$ as in the experiment above.

We define *$\mathcal{A}$'s one-way-CCA advantage over $\mathtt{KEM}$* as:

$$
\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM}}(k) \stackrel{\text{def}}{=} \Pr[\mathbf{Experiment}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM}}(1^k) \text{ returns } \textsc{Win}].
$$

We say that a $\mathtt{KEM}$ is secure against adaptive chosen ciphertext attacks against one-wayness (one-way-CCA secure, for short) if, for any PPT algorithm $\mathcal{A}$, $\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM}}(k)$ is negligible in k. Note that if a KEM is IND-CCA secure [8], then it is one-way-CCA secure. So IND-CCA security is stronger notion than one-way-CCA security.

Suppose that the adversary $\mathcal{A}$ consists of two algorithms $\mathcal{A}_1$ and $\mathcal{A}_2$. The following experiment is called a non-adaptive chosen ciphertext attack on one-wayness of a KEM.

$\mathbf{Experiment}^{\text{ow-cca1}}_{\mathcal{A},\mathtt{KEM}}(1^k)$
$(\mathtt{pk},\mathtt{sk}) \leftarrow \mathtt{K}(1^k), st \leftarrow \mathcal{A}_1^{\mathcal{DEC}(\mathtt{sk},\cdot)}(\mathtt{pk}), (K^*,\psi^*) \leftarrow \mathtt{Enc}(\mathtt{pk}), \widehat{K^*} \leftarrow \mathcal{A}_2(st,\psi^*)$
If $\widehat{K^*} = K^*$ then return Win else return Lose.

Non-adaptive chosen ciphertext attack is weaker model than adaptive one because of the constraint that the learning phase of $\mathcal{A}_1$ is limited to before the solving phase of $\mathcal{A}_2$.

Adaptive and non-adaptive chosen ciphertext attacks are classified to active attacks. On the contrary, there is a passive attack on one-wayness of a KEM described below.

$\mathbf{Experiment}^{\text{ow-pa}}_{\mathcal{A},\mathtt{KEM}}(1^k)$
$(\mathtt{pk},\mathtt{sk}) \leftarrow \mathtt{K}(1^k)$
If $\mathcal{A}_1(\mathtt{pk})$ makes a query, reply $(K_i,\psi_i) \leftarrow \mathtt{Enc}(\mathtt{pk})$
$st \leftarrow \mathcal{A}_1(\{(K_i,\psi_i)\}_{i=1}^{q_{dec}}), (K^*,\psi^*) \leftarrow \mathtt{Enc}(\mathtt{pk}), \widehat{K^*} \leftarrow \mathcal{A}_2(st,\psi^*)$
If $\widehat{K^*} = K^*$ then return Win else return Lose.

Passive attack is weaker model than non-adaptive chosen ciphertext attack because of the constraint that $\mathcal{A}$ cannot choose ciphertexts in the learning phase.

2.3 The Computational Diffie-Hellman assumption and the Twin Diffie-Hellman Technique

We say a solver $\mathcal{S}$, a PPT algorithm, *wins* when $\mathcal{S}$ succeeds in solving a computational problem instance.

A quadruple (g,X,Y,Z) of elements in G_q is called a *Diffie-Hellman (DH) tuple* if the quadruple is written as (g,g^x,g^y,g^{xy}) for some elements x,y in $\mathbf{Z}_q$. A CDH problem instance is a triple $(g,X=g^x,Y=g^y)$, where the exponents x,y are uniformly random in $\mathbf{Z}_q$. A CDH problem solver is a PPT algorithm which, given a CDH problem instance (g,X,Y) as input, tries to return $Z=g^{xy}$.

$\mathbf{Experiment}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(1^k)$
$(q,g) \leftarrow \mathtt{Grp}(1^k), x,y \leftarrow \mathbf{Z}_q, X := g^x, Y := g^y, Z \leftarrow \mathcal{S}(g,X,Y)$
If $Z = g^{xy}$ then return Win else return Lose.

We define *$\mathcal{S}$'s CDH advantage over `Grp`* as:

$$\mathbf{Adv}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(k) \stackrel{\text{def}}{=} \Pr[\mathbf{Experiment}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(1^k) \text{ returns Win}].$$

We say that the CDH assumption [21] holds for `Grp` if, for any PPT algorithm $\mathcal{S}$, $\mathbf{Adv}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(k)$ is negligible in k.

A 6-tuple (g,X_1,X_2,Y,Z_1,Z_2) of elements in G_q is called a *twin Diffie-Hellman (DH) tuple* if the tuple is written as $(g,g^{x_1},g^{x_2},g^y,g^{x_1y},g^{x_2y})$ for some elements x_1,x_2,y in $\mathbf{Z}_q$.

The following lemma of Cash-Kiltz-Shoup is used to decide whether or not a tuple is a twin DH-tuple in security proof for our concrete KEM in Section 4.

Lemma (Cash-Kiltz-Shoup [6] Theorem 2, "Trap Door Test") *Let X_1, r, s be mutually independent random variables, where X_1 takes values in G_q, and each of r, s is uniformly distributed over $\mathbf{Z}_q$. Define the random variable $X_2 := X_1^{-r} g^s$. Suppose that $\widehat{Y}, \widehat{Z_1}, \widehat{Z_2}$ are random variables taking values in G_q, each of which is defined independently of r. Then the probability that the truth value of*

$$\widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s$$

does not agree with the truth value of

$$(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2}) \text{ being a twin DH-tuple}$$

is at most $1/q$. Moreover, if $(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2})$ is a twin DH-tuple, then $\widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s$ certainly holds.

3 Identification Scheme from Key Encapsulation Mechanism

In this section, we show a generic way for deriving an ID scheme secure against concurrent man-in-the-middle attacks from a one-way-CCA secure KEM.

3.1 Construction

Let $\mathtt{KEM} = (\mathtt{K}, \mathtt{Enc}, \mathtt{Dec})$ be a KEM. Then an ID scheme $\mathtt{ID}$ is derived in a natural way as shown in the Fig.1. The key generation algorithm is the same as that of $\mathtt{KEM}$. The verifier $\mathtt{V}$, given a public key $\mathtt{pk}$ as input, invokes the encapsulation algorithm $\mathtt{Enc}$ on $\mathtt{pk}$ and gets its output (K, ψ). $\mathtt{V}$ sends ψ to $\mathtt{P}$. The prover $\mathtt{P}$, given a secret key $\mathtt{sk}$ as input and receiving ψ as input message, invokes the decapsulation algorithm $\mathtt{Dec}$ on $(\mathtt{sk}, \psi)$ and gets its output $\widehat{K}$. $\mathtt{P}$ sends $\widehat{K}$ to $\mathtt{V}$. Finally the verifier $\mathtt{V}$, receiving $\widehat{K}$ as input message, verifies whether or not $\widehat{K}$ is equal to K. If so, then $\mathtt{V}$ returns 1 and otherwise, 0.

Theorem 1 *If a key encapsulation mechanism $\mathtt{KEM}$ is one-way-CCA secure, then the derived identification scheme $\mathtt{ID}$ is cMiM secure. More precisely, for any PPT adversary $\mathcal{A}$ that attacks $\mathtt{ID}$ in cMiM setting, there exists an PPT adversary $\mathcal{B}$ that attacks $\mathtt{KEM}$ in one-way-CCA setting satisfying the following inequality;*

$$\mathbf{Adv}^{\text{imp-cmim}}_{\mathcal{A},\mathtt{ID}}(k) \leqslant \mathbf{Adv}^{\text{ow-cca}}_{\mathcal{B},\mathtt{KEM}}(k).$$

3.2 Proof of Theorem 1

Let $\mathtt{KEM}$ be a one-way-CCA secure KEM and $\mathtt{ID}$ be the derived ID scheme by the construction of Section 3.1. Let $\mathcal{A}$ be any given cMiM adversary on $\mathtt{ID}$. Using $\mathcal{A}$ as subroutine, we construct a PPT one-way-CCA adversary $\mathcal{B}$ that attacks $\mathtt{KEM}$ as follows.

Key Generation
– $\mathtt{K}$: the same as that of $\mathtt{KEM}$

Interaction
– $\mathtt{V}$: given $\mathtt{pk}$ as input;
- Invoke $\mathtt{Enc}$ on $\mathtt{pk}$: $(K, \psi) \leftarrow \mathtt{Enc}(\mathtt{pk})$
- Send ψ to $\mathtt{P}$

– $\mathtt{P}$: given $\mathtt{sk}$ as input and receiving ψ as input message;
- Invoke $\mathtt{Dec}$ on $(\mathtt{sk}, \psi)$: $\widehat{K} \leftarrow \mathtt{Dec}(\mathtt{sk}, \psi)$
- Send $\widehat{K}$ to $\mathtt{V}$

– $\mathtt{V}$: receiving $\widehat{K}$ as input message;
- If $\widehat{K} = K$ then return 1 else return 0

Fig. 1. An ID scheme $\mathtt{ID}$ Derived from a KEM $\mathtt{KEM}$=($\mathtt{K}$,$\mathtt{Enc}$,$\mathtt{Dec}$).

On input $\mathtt{pk}$ and the challenge ciphertext ψ^*, $\mathcal{B}$ initializes its inner state and invokes $\mathcal{A}$ on input $\mathtt{pk}$.

In case that $\mathcal{A}$ queries $\mathtt{V}(\mathtt{pk})$ for the challenge message, $\mathcal{B}$ sends ψ^* to $\mathcal{A}$ as the challenge message.

In case that $\mathcal{A}$ sends a challenge message ψ to a prover clone $\mathtt{P}(\mathtt{sk})$, $\mathcal{B}$ checks whether or not ψ is equal to ψ^*. If so, then $\mathcal{B}$ puts $K = \perp$. Otherwise, $\mathcal{B}$ queries its decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ for the answer for the ciphertext ψ. and gets K. $\mathcal{B}$ sends K to $\mathcal{A}$ as the response message.

In case that $\mathcal{A}$ sends the response message $\widehat{K^*}$ to $\mathtt{V}(\mathtt{pk})$, $\mathcal{B}$ returns $\widehat{K^*}$ as the answer for the challenge ciphertext ψ^*.

Given $\mathtt{pk}$ as input;

Initial Setting
– Initialize the inner state
– Invoke $\mathcal{A}$ on $\mathtt{pk}$

Answering $\mathcal{A}$'s Queries
– In case that $\mathcal{A}$ queries $\mathtt{V}(\mathtt{pk})$ for the challenge message
- Send ψ^* to $\mathcal{A}$

– In case that $\mathcal{A}$ sends ψ to a prover clone $\mathtt{P}(\mathtt{sk})$
- If $\psi = \psi^*$, then put $K := \perp$
- else Query $\mathcal{DEC}$ for the answer for ψ: $K \leftarrow \mathcal{DEC}(\mathtt{sk}, \psi)$
- Send K to $\mathcal{A}$

– In case that $\mathcal{A}$ sends $\widehat{K^*}$ to $\mathtt{V}(\mathtt{pk})$
- Return $\widehat{K^*}$ as the answer for ψ^*

Fig. 2. A One-Way-CCA Adversary $\mathcal{B}$ for the Proof of Theorem 1.

The view of $\mathcal{A}$ in $\mathcal{B}$ is the same as the real view of $\mathcal{A}$. This is obvious except the case that ψ is equal to ψ^*. When $\mathcal{A}$ sent $\psi = \psi^*$, the transcript of the interaction between $\mathtt{P}(\mathtt{sk})$ and $\mathcal{A}(\mathtt{pk})$ would be wholly equal to that between $\mathcal{A}(\mathtt{pk})$ and $\mathtt{V}(\mathtt{pk})$, because the prover $\mathtt{P}$ is deterministic. This is ruled out, so $\mathcal{B}$ must send $K = \perp$ to $\mathcal{A}$ in that case.

If $\mathcal{A}$ wins, then $\mathcal{B}$ wins. Hence the inequality of advantages in Theorem 1 follows. (*Q.E.D.*)

Remark 1. In analogous ways, we can show the following facts. If a KEM is secure against non-adaptive chosen ciphertext attacks on one-wayness, then the derived identification scheme ID is secure against 2-phase concurrent attacks. If a KEM is secure against passive attacks on one-wayness, then the derived identification scheme ID is secure against passive attacks.

Remark 2. The prover P in the Fig.1 is deterministic. Therefore, the derived ID scheme ID is prover-resettable [3]. Moreover, they are also verifier-resettable because ID consists of 2-round interaction.

4 A One-Way-CCA Secure KEM Based on the CDH assumption

In this section, we propose a one-way-CCA secure KEM based on the CDH assumption. The challenge-and-response ID scheme of Anada-Arita [1] can be viewed as a one-way-CCA secure KEM based on the Gap-CDH assumption. Our strategy is to remove the gap assumption of it by applying the Twin Diffie-Hellman technique of Cash-Kiltz-Shoup [6, 18].

In the construction, we employ a target collision resistant (TCR) hash function family. The definition of a TCR hash function family $\mathit{Hfam}(1^k) = \{H_\mu\}_{\mu \in \mathit{Hkey}(1^k)}$ and advantage $\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},\mathit{Hfam}}(k)$ of a PPT collision finder $\mathcal{CF}$ over *Hfam* are noted in Appendix A.

4.1 Construction

The construction of a KEM KEM1 is shown in the Fig.3.

On input 1^k, a key generator K runs as follows. A group generator Grp outputs (q, g) on input 1^k. Then K chooses $x_1, x_2, y_1, y_2 \in \mathbf{Z}_q$ and computes $X_1 = g^{x_1}, X_2 = g^{x_2}, Y_1 = g^{y_1}, Y_2 = g^{y_2}$. In addition, K chooses a hash key μ from a hash key space $\mathit{Hkey}(1^k)$. The hash key μ indicates a specific hash function H_μ with values in $\mathbf{Z}_q$ in a hash function family $\mathit{Hfam}(1^k) = \{H_\mu\}_{\mu \in \mathit{Hkey}(1^k)}$. K sets $\mathtt{pk} = (q, g, X_1, X_2, Y_1, Y_2, \mu)$ and $\mathtt{sk} = (q, g, x_1, x_2, y_1, y_2, \mu)$. Then K returns (pk, sk).

On input pk, an encapsulation algorithm Enc runs as follows. Enc chooses $a \in \mathbf{Z}_q$ at random and computes $h = g^a$. Then Enc computes the hash value $\tau \leftarrow H_\mu(h)$ and computes $d_1 = (X_1^\tau Y_1)^a, d_2 = (X_2^\tau Y_2)^a$. Finally, Enc computes $K = X_1^a$. The ciphertext is $\psi = (h, d_1, d_2)$ and the random key is K. Enc returns a pair (K, ψ).

On input sk and $\psi = (h, d_1, d_2)$, an decapsulation algorithm Dec runs as follows. Dec computes the hash value $\tau \leftarrow H_\mu(h)$. Then Dec verifies whether $\psi = (h, d_1, d_2)$ is a consistent ciphertext, that is, whether $(g, X_1^\tau Y_1, X_2^\tau Y_2, h, d_1, d_2)$ is a twin DH-tuple or not. For this sake, Dec checks whether $h^{\tau x_1 + y_1} = d_1$ and $h^{\tau x_2 + y_2} = d_2$ hold. If they do not hold,

then Dec puts $K = \perp$. Otherwise Dec computes the decapsulation $K = h^{x_1}$. Note here that (g, X_1, h, K) is a DH-tuple. Finally, Dec returns K.

Key Generation
– K: given 1^k as input;
- $(q, g) \leftarrow \mathtt{Grp}(1^k), \mu \leftarrow \mathit{Hkey}(1^k)$
- $x_1, x_2, y_1, y_2 \leftarrow \mathbf{Z}_q, X_1 := g^{x_1}, X_2 := g^{x_2}, Y_1 := g^{y_1}, Y_2 := g^{y_2}$
- $\mathtt{pk} := (q, g, X_1, X_2, Y_1, Y_2, \mu), \mathtt{sk} := (q, g, x_1, x_2, y_1, y_2, \mu)$
- Return $(\mathtt{pk}, \mathtt{sk})$

Encapsulation
– Enc: given pk as input;
- $a \leftarrow \mathbf{Z}_q, h := g^a, \tau \leftarrow H_\mu(h)$
- $d_1 := (X_1^\tau Y_1)^a, d_2 := (X_2^\tau Y_2)^a, \psi = (h, d_1, d_2), K := X_1^a$
- Return (K, ψ)

Decapsulation
– Dec: given $\mathtt{sk}, \psi = (h, d_1, d_2)$ as input;
- $\tau \leftarrow H_\mu(h)$
- If $h^{\tau x_1 + y_1} \neq d_1$ or $h^{\tau x_2 + y_2} \neq d_2$ then $K := \perp$ else $K := h^{x_1}$
- Return K

Fig. 3. A One-Way-CCA Secure KEM KEM1.

Theorem 2 *The key encapsulation mechanism KEM1 is one-way-CCA secure based on the CDH assumption and the target collision resistance of an employed hash function. More precisely, for any PPT one-way-CCA adversary $\mathcal{A}$ on KEM1 that queries decapsulation oracle at most q_{dec} times, there exist a PPT CDH problem solver $\mathcal{S}$ on Grp and a PPT collision-finder $\mathcal{CF}$ on Hfam which satisfy the following tight reduction.*

$$\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM1}}(k) \leqslant \frac{q_{dec}}{q} + \mathbf{Adv}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(k) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},\mathit{Hfam}}(k).$$

4.2 Proof of Theorem 2

Let $\mathcal{A}$ be any given adversary that attacks on KEM1 in one-way-CCA setting. Using $\mathcal{A}$ as subroutine, we construct a PPT CDH problem solver $\mathcal{S}$, where an algebraic trick [17] and the Twin DH technique [6] is essentially used.

$\mathcal{S}$ is given (q, g) and $X = g^x, Y = g^y$ as input, where x and y are random and hidden. $\mathcal{S}$ initializes its inner state. $\mathcal{S}$ chooses $a^* \in \mathbf{Z}_q$ at random and computes $h^* = Y g^{a^*}$. Then $\mathcal{S}$ chooses μ from $\mathit{Hkey}(1^k)$ and computes $\tau^* \leftarrow H_\mu(h^*)$. $\mathcal{S}$ chooses $r, s \in \mathbf{Z}_q$ at random, and puts $X_1 = X, X_2 = X_1^{-r} g^s$. $\mathcal{S}$ chooses $u_1, u_2 \in \mathbf{Z}_q$ at random, and computes $W_1 = X_1^{-\tau^*} g^{u_1}, W_2 = X_2^{-\tau^*} g^{u_2}$. $\mathcal{S}$ chooses $d_1^* = (h^*)^{u_1}, d_2^* = (h^*)^{u_2}$. $\mathcal{S}$ sets $\mathtt{pk} = (q, g, X_1, X_2, Y_1, Y_2, \mu), \psi^* = (h^*, d_1^*, d_2^*)$ and invokes $\mathcal{A}$ on input pk and ψ^*. Note that pk is correctly distributed. Note also that $\mathcal{S}$ does not know x_1, x_2, w_1, w_2 at all, where x_1, x_2, w_1, w_2 are the discrete log of X_1, X_2, W_1, W_2, respectively;

$$w_i = \log_g(W_i) = -\tau^* x_i + u_i, \quad i = 1, 2. \tag{1}$$

$\mathcal{S}$ replies to $\mathcal{A}$'s queries as follows.

In case that $\mathcal{A}$ queries its decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ for the answer for $\psi = (h, d_1, d_2)$, $\mathcal{S}$ checks whether ψ is equal to ψ^* or not. If $\psi = \psi^*$, then $\mathcal{S}$ puts $K = \perp$. Otherwise, $\mathcal{S}$ computes $\tau \leftarrow H_\mu(h)$ and verifies whether $(g, X_1^\tau W_1, X_2^\tau W_2, h, d_1, d_2)$ is consistent (Call this case Consistency-Check). That is, $\mathcal{S}$ verifies whether it is a twin DH-tuple. $\mathcal{S}$ does it in the following way. Put $\widehat{Y} = h^{\tau - \tau^*}, \widehat{Z_1} = d_1/h^{u_1}, \widehat{Z_2} = d_2/h^{u_2}$. If $\widehat{Z_1}^r \widehat{Z_2} \neq \widehat{Y}^s$, then it is not a twin DH-tuple and $\mathcal{S}$ puts $K = \perp$. Otherwise, $\mathcal{S}$ decides that it is a twin DH-tuple. Then, if $\tau \neq \tau^*$, $\mathcal{S}$ computes $K = \widehat{Z_1}^{1/(\tau - \tau^*)}$ (Call this case $\mathscr{D}$). If $\tau = \tau^*$, then $\mathcal{S}$ aborts (Call this case Abort). Then $\mathcal{S}$ replies K to $\mathcal{A}$ except the case Abort.

Given $q, g, X = g^x, Y = g^y$ as input;

Initial Setting
– Initialize the inner state
– $a^* \leftarrow \mathbf{Z}_q, h^* := Y g^{a^*}$
– $\mu \leftarrow Hkey(1^k), \tau^* \leftarrow H_\mu(h^*)$
– $r, s \leftarrow \mathbf{Z}_q, X_1 := X, X_2 := X_1^{-r} g^s$
– $u_1, u_2 \leftarrow \mathbf{Z}_q, W_1 := X_1^{-\tau^*} g^{u_1}, W_2 := X_2^{-\tau^*} g^{u_2}$
– $d_1^* := (h^*)^{u_1}, d_2^* := (h^*)^{u_2}$
– $\mathtt{pk} := (q, g, X_1, X_2, W_1, W_2, \mu), \psi^* := (h^*, d_1^*, d_2^*)$
– Invoke $\mathcal{A}$ on $\mathtt{pk}$ and ψ^*

Answering $\mathcal{A}$'s Queries

– In case that $\mathcal{A}$ queries $\mathcal{DEC}(\mathtt{sk}, \cdot)$ for the answer for $\psi = (h, d_1, d_2)$
  • If $\psi = \psi^*$, then put $K := \perp$
  • else (: the case Consistency-Check)
    $\tau \leftarrow H_\mu(h), \widehat{Y} := h^{\tau - \tau^*}, \widehat{Z_1} := d_1/h^{u_1}, \widehat{Z_2} := d_2/h^{u_2}$
    If $\widehat{Z_1}^r \widehat{Z_2} \neq \widehat{Y}^s$, then $K := \perp$
    else
      If $\tau \neq \tau^*$, then $K := \widehat{Z_1}^{1/(\tau - \tau^*)}$ (: the case $\mathscr{D}$)
      else abort (: the case Abort)
  • Reply K to $\mathcal{A}$

– In case that $\mathcal{A}$ replies $\widehat{K^*}$ as the answer for ψ^*
  • $Z := \widehat{K^*}/X^{a^*}$
  • Return Z

Fig. 4. A CDH Problem Solver $\mathcal{S}$ for the Proof of Theorem 2.

$\mathcal{S}$ can simulates the real view of $\mathcal{A}$ perfectly until the case Abort happens except a negligible case, as we see below.

Firstly, the challenge ciphertext $\psi^* = (h^*, d_1^*, d_2^*)$ which $\mathcal{S}$ gives is consistent and correctly distributed. This is because the distribution of (h^*, d_1^*, d_2^*) is equal to that of the real consistent ciphertext $\psi = (h, d_1, d_2)$. To see it, note that $y + a^*$ is substituted for a:

$$h^* = g^{y+a^*},\ d_i^* = (g^{y+a^*})^{u_i} = (g^{u_i})^{y+a^*} = (X_i^{\tau^*} W_i)^{y+a^*},\quad i = 1, 2.$$

Secondly, $\mathcal{S}$ simulates the decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ perfectly except a negligible case. To see it, note that the consistency check really works though it may involve a negligible error case, which is explained by the following two claims.

Claim 1 *$(g, X_1^\tau W_1, X_2^\tau W_2, h, d_1, d_2)$ is a twin-DH tuple if and only if $(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2})$ is a twin-DH tuple for $\widehat{Y} = h^{\tau-\tau^*}$, $\widehat{Z_1} = d_1/h^{u_1}$ and $\widehat{Z_2} = d_2/h^{u_2}$.*

Proof of Claim 1 is done by direct calculations and noted in Appendix B.

Claim 2 *If $\widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s$ holds for $\widehat{Y} = h^{\tau-\tau^*}$, $\widehat{Z_1} = d_1/h^{u_1}$ and $\widehat{Z_2} = d_2/h^{u_2}$, then $(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2})$ is a twin-DH tuple except an error case that occurs at most $1/q$ probability. Conversely, if $(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2})$ is a twin-DH tuple, then $\widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s$ certainly holds.*

Proof of Claim 2. We observe that each of $\widehat{Y} = h^{\tau-\tau^*}$, $\widehat{Z_1} = d_1/h^{u_1}$ and $\widehat{Z_2} = d_2/h^{u_2}$ is given independently of r. So we can apply the Lemma in Section 2 to get the claim. (*Q.E.D.*)

Let us define the event Overlook as:

$$\textsc{Overlook} \stackrel{\text{def}}{=} \begin{cases} & \widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s \text{ holds} \\ \text{and} & (g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2}) \text{ is not a twin DH-tuple.} \end{cases}$$

Then, by the Claim 2, the probability that Overlook occurs is at most $1/q$ for each consistency check. So for at most q_{dec} times consistency checks Consistency-Check$_i$, $i = 1, \ldots, q_{dec}$, the probability that the corresponding Overlook$_i$ occurs at least one time is at most q_{dec}/q. Hence we have

$$\Pr[\bigvee_{i=1}^{q_{dec}} \textsc{Overlook}_i] \leqslant \frac{q_{dec}}{q}. \tag{2}$$

q_{dec} is polynomial in k and q is exponential in k, so q_{dec}/q, and hence the left hand side, are negligible in k.

Suppose $\mathcal{S}$ has confirmed that a decapsulation query $\psi = (h, d_1, d_2)$ passed the consistency check. In that case, $(g, X_1^\tau W_1, X_2^\tau W_2, h, d_1, d_2)$ is a twin-DH tuple except a negligible case Overlook, so $d_1 = h^{\tau x_1 + w_1}$ holds. If, in addition, $\mathcal{S}$ is in the case $\mathscr{D}$ (that is, $\tau \neq \tau^*$), then $\mathcal{S}$ correctly replies the answer for ψ to $\mathcal{A}$. This is because $K = \widehat{Z_1}^{1/(\tau-\tau^*)} = (d_1/h^{u_1})^{1/(\tau-\tau^*)}$ is equal to h^{x_1} by the following equalities.

$$d_1/h^{u_1} = h^{\tau x_1 + w_1 - u_1} = h^{(\tau-\tau^*)x_1 + (\tau^* x_1 + w_1 - u_1)} = h^{(\tau_i - \tau^*)x_1},$$

where we use the equality (1).

Hence $\mathcal{S}$ simulates the decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ perfectly except the negligible case Overlook.

As a whole, $\mathcal{S}$ simulates the real view of $\mathcal{A}$ perfectly except the negligible case.

Now we evaluate the advantage of $\mathcal{S}$. When $\mathcal{A}$ wins, $(g, X, h^*, \widehat{K^*})$ is a DH-tuple, so the followings hold.

$$\widehat{K^*} = X^{y+a^*} = g^{x(y+a^*)} = g^{xy+xa^*}.$$

Hence the output Z is equal to $\widehat{K^*}/X^{a^*} = g^{xy}$. That is, Z is the answer for a CDH-problem instance (g, X, Y), which means that $\mathcal{S}$ wins. Therefore the probability that $\mathcal{S}$ wins is lower bounded by the probability that $\mathcal{A}$ wins and $\textsc{Overlook}_i$ never occurs for $i = 1, \ldots, q_{dec}$ and Abort does not happen:

$$\begin{aligned}
\Pr[\mathcal{S} \text{ wins}] &\geqslant \Pr[\mathcal{A} \text{ wins } \wedge (\bigwedge_{i=1}^{q_{dec}} (\neg \textsc{Overlook}_i)) \wedge (\neg \textsc{Abort})] \\
&\geqslant \Pr[\mathcal{A} \text{ wins}] - \Pr[(\bigvee_{i=1}^{q_{dec}} \textsc{Overlook}_i) \vee \textsc{Abort}] \\
&\geqslant \Pr[\mathcal{A} \text{ wins}] - \frac{q_{dec}}{q} - \Pr[\textsc{Abort}],
\end{aligned}$$

where we use the inequality (2). Hence we get

$$\mathbf{Adv}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(k) \geqslant \mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM1}}(k) - \frac{q_{dec}}{q} - \Pr[\textsc{Abort}].$$

So our task being left is to show that $\Pr[\textsc{Abort}]$ is negligible in k.

Claim 3 *The probability* $\Pr[\textsc{Abort}]$ *is negligible in* k.

Proof of Claim 3. Using $\mathcal{A}$ as subroutine, we construct a PPT target collision finder $\mathcal{CF}$ on *Hfam* as follows. Given 1^k as input, $\mathcal{CF}$ initializes its inner state. $\mathcal{CF}$ gets (q, g) from $\mathtt{Grp}(1^k)$. $\mathcal{CF}$ chooses $a^* \in \mathbf{Z}_q$ at random, computes $h^* = g^{a^*}$ and outputs h^*. $\mathcal{CF}$ receives a random hash key μ and computes $\tau^* \leftarrow H_\mu(h^*)$. Then $\mathcal{CF}$ makes a secret key and public key honestly by itself : $\mathtt{sk} = (q, g, x_1, x_2, y_1, y_2, \mu), \mathtt{pk} = (q, g, X_1, X_2, Y_1, Y_2, \mu)$. Finally, $\mathcal{CF}$ computes $d_1^* = (X_1^{\tau^*} Y_1)^{a^*}, d_2^* = (X_2^{\tau^*} Y_2)^{a^*}$ and puts $\psi^* = (h^*, d_1^*, d_2^*)$. $\mathcal{CF}$ invokes $\mathcal{A}$ on $\mathtt{pk}$ and ψ^*.

In case that $\mathcal{A}$ queries the decapsulation oracle $\mathcal{DEC}(\mathtt{sk}, \cdot)$ for the answer for $\psi = (h, d_1, d_2)$, $\mathcal{CF}$ checks whether ψ is equal to ψ^* or not. If $\psi = \psi^*$, then $\mathcal{CF}$ replies $K = \perp$ to $\mathcal{A}$. Else if $\psi \neq \psi^*$, $\mathcal{CF}$ computes $\tau \leftarrow H_\mu(h)$ and verifies whether $(g, X_1^\tau Y_1, X_2^\tau Y_2, h, d_1, d_2)$ is consistent (that is, a twin DH-tuple). $\mathcal{CF}$ can do it in the same way as the $\mathtt{Dec}$ does because $\mathcal{CF}$ has the secret key $\mathtt{sk}$. If it does not hold, $\mathcal{CF}$ replies $K = \perp$ to $\mathcal{A}$. Otherwise, if $\tau \neq \tau^*$, then $\mathcal{CF}$ replies $K = h^{x_1}$ to $\mathcal{A}$. If $\tau = \tau^*$, then $\mathcal{CF}$ returns h and stops (Call this case Collision).

Note that the view of $\mathcal{A}$ in $\mathcal{CF}$ is the same as the real view until the case Collision happens. Therefore, the view of $\mathcal{A}$ in $\mathcal{CF}$ is the same as the view of $\mathcal{A}$ in $\mathcal{S}$ except the negligible case Overlook. By the inequality (2), we have

$$|\Pr[\textsc{Collision}] - \Pr[\textsc{Abort}]| \leqslant \frac{q_{dec}}{q}. \tag{3}$$

Notice that the case Collision implies the followings.

$$\begin{cases} & (g, X_1^{\tau} Y_1, X_2^{\tau} Y_2, h, d_1, d_2) \text{ is a twin DH-tuple} \\ \text{and} & (g, X_1^{\tau^*} Y_1, X_2^{\tau^*} Y_2, h^*, d_1^*, d_2^*) \text{ is a twin DH-tuple} \\ \text{and} & \tau = \tau^*. \end{cases}$$

If, in addition to the above conditions, h were equal to h^*, then (d_1, d_2) would be equal to (d_1^*, d_2^*). This means that ψ is equal to ψ^*, a contradiction. Hence it must hold that

$$h \neq h^*.$$

So in the case Collision, $\mathcal{CF}$ succeeds in making a collision. That is;

$$\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},Hfam}(k) = \Pr[\textsc{Collision}]. \tag{4}$$

Combining (3) and (4), we get

$$|\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},Hfam}(k) - \Pr[\textsc{Abort}]| \leqslant \frac{q_{dec}}{q},$$

that means

$$\Pr[\textsc{Abort}] \leqslant \mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},Hfam}(k) + \frac{q_{dec}}{q}.$$

The two terms of right hand side are both negligible in k according to the assumptions, so $\Pr[\textsc{Abort}]$ is also negligible in k. (*Q.E.D.*)

4.3 A Tuning for More Efficiency and the Corresponding Identification Scheme

To reduce the length of ciphertext $\psi = (h, d_1, d_2)$, we can replace the term d_2 with its hash value $v_2 := H(d_2)$. Let us call this KEM `KEM2`. In `KEM2`, the ciphertext turns to $\psi = (h, d_1, v_2)$, so the consistency check for index 2 in $\mathtt{Dec}(\mathtt{sk}, \psi)$ becomes $H(h^{\tau x_2 + y_2}) \stackrel{?}{=} v_2$. In addition, the trapdoor test in the security proof, $\widehat{Z_1}^r \widehat{Z_2} \stackrel{?}{=} \widehat{Y}^s$, is deformed as follows.

$$\begin{aligned} \widehat{Z_1}^r \widehat{Z_2} = \widehat{Y}^s &\iff (d_1/h^{u_1})^r (d_2/h^{u_2}) = (h^{\tau - \tau^*})^s \\ &\iff d_1^{-r} h^{ru_1 + u_2 + s(\tau - \tau^*)} = d_2 \\ &\Longrightarrow H(d_1^{-r} h^{ru_1 + u_2 + s(\tau - \tau^*)}) = v_2. \end{aligned}$$

The last equality may cause collision, so the security statement for `KEM2` needs collision resistance assumption of the employed hash function H (the name of game "cr" in $\mathbf{Adv}^{\mathrm{cr}}_{\mathcal{CF}',Hfam}(k)$ below means collision resistance).

Corollary of Theorem 2 *The key encapsulation mechanism `KEM2` is one-way-CCA secure based on the CDH assumption and the target collision resistance and the collision resistance of an employed hash function. More precisely, for any PPT one-way-CCA adversary $\mathcal{A}$ on `KEM2` that queries decapsulation oracle at most q_{dec} times, there exist a PPT CDH problem*

solver $\mathcal{S}$ on $\mathtt{Grp}$ and a PPT collision-finder $\mathcal{CF}, \mathcal{CF}'$ on Hfam which satisfy the following tight reduction.

$$\mathbf{Adv}^{\text{ow-cca}}_{\mathcal{A},\mathtt{KEM2}}(k) \leqslant \frac{q_{dec}}{q} + \mathbf{Adv}^{\text{cdh}}_{\mathcal{S},\mathtt{Grp}}(k) + \mathbf{Adv}^{\text{tcr}}_{\mathcal{CF},Hfam}(k) + \mathbf{Adv}^{\text{cr}}_{\mathcal{CF}',Hfam}(k).$$

The ID scheme derived from KEM2 is shown in the Fig.5. The maximum message length of the ID scheme from KEM1 (that is, the length of challenge message) is three elements in Grp. By the tuning, the maximum message length of the ID scheme from KEM2 reduces to two elements in Grp plus one hash value.

Key Generation

– K: given 1^k as input;
- $(q, g) \leftarrow \mathtt{Grp}(1^k), \mu \leftarrow Hkey(1^k)$
- $x_1, x_2, y_1, y_2 \leftarrow \mathbf{Z}_q, X_1 := g^{x_1}, X_2 := g^{x_2}, Y_1 := g^{y_1}, Y_2 := g^{y_2}$
- $\mathtt{pk} := (q, g, X_1, X_2, Y_1, Y_2, \mu), \mathtt{sk} := (q, g, x_1, x_2, y_1, y_2, \mu)$
- Return $(\mathtt{pk}, \mathtt{sk})$

Interaction

– V: given pk as input;
- $a \leftarrow \mathbf{Z}_q, h := g^a, \tau \leftarrow H_\mu(h)$
- $d_1 := (X_1^\tau Y_1)^a, v_2 := H((X_2^\tau Y_2)^a), \psi = (h, d_1, v_2), K := X_1^a$
- Send ψ to P

– P: given sk as input and receiving $\psi = (h, d_1, v_2)$ as input message;
- $\tau \leftarrow H_\mu(h)$
- If $h^{\tau x_1 + y_1} \neq d_1$ or $H(h^{\tau x_2 + y_2}) \neq v_2$ then $\widehat{K} :=\perp$ else $\widehat{K} := h^{x_1}$
- Send $\widehat{K}$ to V

– V: receiving $\widehat{K}$ as input message;
- If $\widehat{K} = K$ then return 1 else return 0

Fig. 5. The ID Scheme Derived from KEM2.

5 Efficiency Comparison

In this section, we evaluate the efficiency of our KEM-based ID schemes comparing with other ID schemes secure against concurrent man-in-the-middle attacks in the standard model. It turns out that our schemes are faster than the ID scheme from the one-way-CCA KEM of Hanaoka-Kurosawa [15].

Comparable schemes are divided into four categories. The first category is Σ-protocols, the second category is challenge-and-response ID schemes obtained from EUF-CMA signature schemes, the third category is the ones obtained from IND-CCA encryption schemes and the fourth category is the ones obtained from one-way-CCA KEMs. Note that we are considering schemes whose security proofs are in the standard model.

In the first category, to the best of author's knowledge, the Gennaro scheme is the most efficient but is no more efficient than the Cramer-Shoup-based ID scheme [8, 25, 9]. As for the second category, all the known signature schemes in the standard model, including

the Short Signature [2] and the Water's Signature [26], are far more inefficient than the Cramer-Shoup-based ID scheme.

In the third category, the Cramer-Shoup-based ID scheme is the most efficient. Note that the Cramer-Shoup KEM [25, 9] (which we call `Sh00KEM`) is also usable as an ID scheme, because the KEM is IND-CCA and hence one-way-CCA secure. On the contrary, we remark that the KEM part of Kurosawa-Desmedt Encryption scheme [19] is not comparable, because the KEM is not one-way-CCA secure [14].

In the fourth category the one-way-CCA KEM of Hanaoka-Kurosawa [15] (which we call `HK08KEM`) is vary comparable, as its security is reduced to the CDH assumption. A recently proposed ID scheme of Anada-Arita [1] is also comparable as it can be considered an ID scheme derived from one-way-CCA secure KEM (which we call `AA10KEM`). A one-time signature in that scheme can be replaced by a TCR hash function value, so it is directly comparable.

Table 1 shows comparison of these KEMs with our KEMs `KEM1` and `KEM2`. In the table, we are comparing computational amount by counting the number of exponentiation. We are also comparing the maximum message length.

Table 1. Efficiency Comparison of `KEM1` and `KEM2` with Previous KEMs. G-et. and hash mean an element in G_q and a hash value in $\mathbf{Z}_q$, respectively. For the DDH assumption and the Gap-CDH assumption, see [8] and [21], respectively. OW-CCA means one-way-CCA security.

KEM	Assump.	Security as KEM	Security as ID scheme	Exponentiation `V(Enc)`	Exponentiation `P(Dec)`	Max. Msg. Length (Challenge Msg.)
`Sh00KEM`	DDH	IND-CCA	cMiM	5	3	3 G-et.
`AA10KEM`	Gap-CDH	OW-CCA	cMiM	4	2	2 G-et.
`HK08KEM`	CDH	OW-CCA	cMiM	7	3	3 G-et.
Our `KEM1`	CDH	OW-CCA	cMiM	6	3	3 G-et.
Our `KEM2`	CDH	OW-CCA	cMiM	6	3	2 G-et.+ 1 hash

As shown in Table 1, the ID schemes derived from `KEM1` and `KEM2` are faster in one exponentiation for verifier than the one derived from `HK08KEM` based on the same CDH assumption, which is the weakest in the three assumptions in the table. We can also look at the table as a trade off between strength of assumption and computational amount to execute protocols.

6 Conclusion

We showed a generic way for deriving a cMiM secure ID scheme from a one-way-CCA secure KEM. Then we gave a concrete discrete logarithm-based one-way-CCA secure KEM utilizing the Twin Diffie-Hellman technique. The obtained ID scheme from the KEM performs better than the currently most efficient scheme whose security is based on the same CDH assumption.

Acknowledgements

We appreciate thoughtful suggestions [18] offered by Prof. Kiltz at ProvSec 2010. We also thank Prof. Kurosawa for inspiring comments at ProvSec 2010.

References

1. H. Anada, S. Arita, *"Identification Schemes of Proofs of Ability Secure against Concurrent Man-in-the-Middle Attacks"*. In Proc. of *ProvSec 2010*, Malacca, Malaysia, Oct. 13-15, 2010, Lecture Notes in Computer Science, vol. 6402, pp. 18-34, Springer-Verlag, Berlin, Germany.
2. D. Boneh, X. Boyen, *"Short Signatures without Random Oracles"*. In Proc. of *EUROCRYPT 2004*, Interlaken, Switzerland, May 2-6, 2004, Lecture Notes in Computer Science, vol. 3027, pp. 56-73, Springer-Verlag, Berlin, Germany.
3. M. Bellare, M. Fischlin, S. Goldwasser, S. Micali, *"Identification Protocols Secure against Reset Attacks"*. In Proc. of *EUROCRYPT 2001*, Innsbruck, Austria, May 6-10, 2001, Lecture Notes in Computer Science, vol. 2045, pp. 495-511, Springer-Verlag, Berlin, Germany.
4. M. Bellare, O. Goldreich, *"On Defining Proofs of Knowledge"*. In Proc. of *CRYPTO '92*, Santa Barbara, CA, USA, Aug. 16-20, 1992, Lecture Notes in Computer Science, vol. 740, pp. 390-420, Springer-Verlag, Berlin, Germany.
5. M. Bellare, A. Palacio, *"GQ and Schnorr Identification Schemes: Proofs of Security against Impersonation under Active and Concurrent Attacks"*. In Proc. of *CRYPTO 2002*, Santa Barbara, CA, USA, Aug. 18-22, 2002, Lecture Notes in Computer Science, vol. 2442, pp. 162-177, Springer-Verlag, Berlin, Germany.
6. D. Cash, E. Kiltz, V. Shoup, *"The Twin Diffie-Hellman Problem and Applications"*. In Proc. of *EUROCRYPT 2008*, Istanbul, Turkey, April 13-17, 2008, Lecture Notes in Computer Science, vol. 4965, pp. 127-145, Springer-Verlag, Berlin, Germany. Also available at Cryptology ePrint Archive, 2008/067, http://eprint.iacr.org/
7. R. Cramer, I. Damgård, J. B. Nielsen, *"Multiparty Computation from Threshold Homomorphic Encryption"*. In Proc. of *EUROCRYPT 2001*, Innsbruck, Austria, May 6-10, 2001, Lecture Notes in Computer Science, vol. 2045, pp. 280-300, Springer-Verlag, Berlin, Germany.
8. R. Cramer, V. Shoup, *"A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack"*. In Proc. of *CRYPTO '98*, Santa Barbara, CA, USA, Aug. 23-27, 1998, Lecture Notes in Computer Science, vol. 1462, pp. 13-25, Springer-Verlag, Berlin, Germany.
9. R. Cramer, V. Shoup, *"Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack"*. SIAM Journal on Computing, vol. 33, num. 1, pp. 167-226, Aug. 2003.
10. E. Fujisaki, *"New Constructions of Efficient Simulation-Sound Commitments Using Encryption"*. In Proc. of *The 2011 Symposium on Cryptography and Information Security*, Kokura, Japan, Jan. 25-28, 2011. The Institute of Electronics, Information and Communication Engineers.
11. R. Gennaro, *"Multi-trapdoor Commitments and their Applications to Non-Malleable Protocols"*. In Proc. of *CRYPTO 2004*, Santa Barbara, CA, USA, Aug. 15-19, 2004, Lecture Notes in Computer Science, vol. 3152, pp. 220-236, Springer-Verlag, Berlin, Germany.
12. S. Goldwasser, S. Micali, C. Rackoff, *"The Knowledge Complexity of Interactive Proof Systems"*. SIAM Journal on Computing, vol. 18, num. 1, pp. 186-208, Feb. 1989.
13. L. Guillou, J. J. Quisquater, *"A Paradoxical Identity-Based Signature Scheme Resulting from Zero-Knowledge"*. In Proc. of *CRYPTO '88*, Santa Barbara, CA, USA, Aug. 21-25, 1988, Lecture Notes in Computer Science, vol. 403, pp. 216-231, Springer-Verlag, Berlin, Germany.
14. J. Herranz, D. Hofheinz, E. Kiltz, *"The Kurosawa-Desmedt Key Encapsulation is not Chosen-Ciphertext Secure"*. Cryptology ePrint Archive, 2006/207, http://eprint.iacr.org/
15. G. Hanaoka, K. Kurosawa, *"Efficient Chosen Ciphertext Secure Public Key Encryption under the Computational Diffie-Hellman Assumption"*. In Proc. of *ASIACRYPT 2008*, Melbourne, Australia, Dec. 7-11, 2008, Lecture Notes in Computer Science, vol. 5350, pp. 308-325, Springer-Verlag, Berlin, Germany. Also available at Cryptology ePrint Archive, 2008/211, http://eprint.iacr.org/
16. J. Katz, *"Efficient Cryptographic Protocols Preventing "Man-in-the-Middle" Attacks"*. Doctor of Philosophy Dissertation, Columbia University, New York, NY, USA, 2002.

17. E. Kiltz, *"Chosen-Ciphertext Security from Tag-Based Encryption"*. In Proc. of *TCC 2006*, New York, NY, USA, March 4-7, 2006, Lecture Notes in Computer Science, vol. 3876, pp. 581-600, Springer-Verlag, Berlin, Germany.
18. E. Kiltz, Personal communication at ProvSec 2010, Malacca, Malaysia, Oct. 13-15, 2010.
19. K. Kurosawa, Y. Desmedt, *"A New Paradigm of Hybrid Encryption Scheme"*. In Proc. of *CRYPTO 2004*, Santa Barbara, CA, USA, Aug. 15-19, 2004, Lecture Notes in Computer Science, vol. 3152, pp. 426-442, Springer-Verlag, Berlin, Germany.
20. M. Naor, M. Yung, *"Universal One-Way Hash Functions and their Cryptographic Applications"*. In Proc. of *the 21st Symposium on Theory of Computing*, Seattle, Washington, USA, May 14-17, 1989, pp. 33-43, Association for Computing Machinery.
21. T. Okamoto, D. Pointcheval, *"The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes"*. In Proc. of *PKC 2001*, Cheju Island, Korea, February 13-15, 2001, Lecture Notes in Computer Science, vol. 1992, pp. 104-118, Springer-Verlag, Berlin, Germany.
22. D. Pointcheval, *"Chosen-Ciphertext Security for Any One-Way Cryptosystem"*. In Proc. of *PKC 2000*, Melbourne, Victoria, Australia, January 18-20, 2000, Lecture Notes in Computer Science, vol. 1751, pp. 129-146, Springer-Verlag, Berlin, Germany.
23. J. Rompel, *"One-Way Functions are Necessary and Sufficient for Secure Signatures"*. In Proc. of *the 22nd Annual Symposium on Theory of Computing*, Baltimore, MD, USA, May 13-17, 1990, pp.387-384, Association for Computing Machinery.
24. C. P. Schnorr, *"Efficient Identification and Signatures for Smart Cards"*. In Proc. of *CRYPTO '89*, Santa Barbara, CA, USA, Aug. 20-24, 1989, Lecture Notes in Computer Science, vol. 435, pp. 239-252, Springer-Verlag, Berlin, Germany.
25. V. Shoup, *"Using Hash Functions as a Hedge against Chosen Ciphertext Attack"*. In Proc. of *EUROCRYPT 2000*, Bruges, Belgium, May 14-18, 2000, Lecture Notes in Computer Science, vol. 1807, pp. 275-288, Springer-Verlag, Berlin, Germany.
26. B. Waters, *"Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions"*. In Proc. of *CRYPTO 2009*, Santa Barbara, CA, USA, Aug. 16-20, 2009, Lecture Notes in Computer Science, vol. 5677, pp. 619-636, Springer-Verlag, Berlin, Germany.
27. S. Yilek, *"Resettable Public-Key Encryption: How to Encrypt on a Virtual Machine"*. In Proc. of *the Cryptographers' Track at the RSA Conference 2010*, San Francisco, CA, USA, March 1-5, 2010, Lecture Notes in Computer Science, vol. 5985, pp. 41-56, Springer-Verlag, Berlin, Germany.

Appendix

A Target Collision Resistant Hash Functions

Target collision resistant (TCR) hash functions [20, 23] are treated as a family. Let us denote a function family as $\mathit{Hfam}(1^k) = \{H_\mu\}_{\mu \in \mathit{Hkey}(1^k)}$. Here $\mathit{Hkey}(1^k)$ is a hash key space, $\mu \in \mathit{Hkey}(1^k)$ is a hash key and H_μ is a function from $\{0,1\}^*$ to $\{0,1\}^k$. We may assume that H_μ is from $\{0,1\}^*$ to $\mathbf{Z}_q$, where q is a prime of length k.

Given a PPT algorithm $\mathcal{CF}$, a collision finder, we consider the following experiment.

$$\begin{array}{l}\mathbf{Experiment}^{\mathrm{tcr}}_{\mathcal{CF},\mathit{Hfam}}(1^k)\\ \quad m \leftarrow \mathcal{CF}(1^k), \mu \leftarrow \mathit{Hkey}(1^k), m' \leftarrow \mathcal{CF}(\mu)\\ \quad \text{If } H_\mu(m) = H_\mu(m') \text{ then return } \textsc{Win} \text{ else return } \textsc{Lose}.\end{array}$$

Then we define *$\mathcal{CF}$'s advantage over Hfam in the game of target collision resistance* as follows.

$$\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},\mathit{Hfam}}(k) \stackrel{\mathrm{def}}{=} \Pr[\mathbf{Experiment}^{\mathrm{tcr}}_{\mathcal{CF},\mathit{Hfam}}(1^k) \text{ returns } \textsc{Win}].$$

We say that *Hfam* is *a TCR function family* if, for any PPT algorithm $\mathcal{CF}$, $\mathbf{Adv}^{\mathrm{tcr}}_{\mathcal{CF},Hfam}(k)$ is negligible in k.

TCR hash function families can be constructed based on the existence of a one-way function [20, 23].

B Proof of Claim 1

Assume that $(g, X_1^{\tau}W_1, X_2^{\tau}W_2, h, d_1, d_2)$ is a twin-DH tuple and put

$$X_i^{\tau}W_i =: g^{\alpha_i}, h =: g^{\beta}, d_i =: g^{\alpha_i\beta},\ i = 1, 2.$$

Then $h^{\tau-\tau^*} = g^{\beta(\tau-\tau^*)}$. Note that we have set

$$W_i := X_i^{-\tau^*} g^{u_i},\ i = 1, 2.$$

So $X_i^{\tau}W_i = X_i^{\tau}X_i^{-\tau^*}g^{u_i} = X_i^{\tau-\tau^*}g^{u_i}$ and we have

$$g^{\alpha_i-u_i} = X_i^{\tau-\tau^*},\ i = 1, 2.$$

Hence

$$d_i/h^{u_i} = g^{\alpha_i\beta}/g^{\beta u_i} = g^{(\alpha_i-u_i)\beta} = X_i^{\beta(\tau-\tau^*)},\ i = 1, 2.$$

This means $(g, X_1, X_2, \widehat{Y}, \widehat{Z_1}, \widehat{Z_2})$ is a twin-DH tuple for $\widehat{Y} = h^{\tau-\tau^*}, \widehat{Z_1} = d_1/h^{u_1}$ and $\widehat{Z_2} = d_2/h^{u_2}$.

The converse is also verified by setting the goal to be $d_i = g^{\alpha_i\beta},\ i = 1, 2$ and starting from the assumption that $\widehat{Z}_i = d_i/h^{u_i} = X_i^{\beta(\tau-\tau^*)},\ i = 1, 2$. (*Q.E.D.*)