LETTER *Special Section on Information and Communication System Security*

CCA-Secure Public Key Encryption without Group-Dependent Hash Functions

Yang CUI[†,††a)], **Goichiro HANAOKA**[†], ***Nonmembers***, ***and*** **Hideki IMAI**[†,††], ***Fellow***

SUMMARY So far, in almost all of the practical public key encryption schemes, hash functions which are dependent on underlying cyclic groups are necessary, e.g., $H : \{0,1\}^* \to \mathbb{Z}_p$ where p is the order of the underlying cyclic group, and it could be required to construct a dedicated hash function for each public key. The motivation of this note is derived from the following two facts: 1). there is an important technical gap between hashing to a specific prime-order group and hashing to a certain length bit sequence, and this could cause a security hole; 2). surprisingly, to our best knowledge, there is no explicit induction that one could use the simple construction, instead of tailor-made hash functions. In this note, we investigate this issue and provide the *first rigorous* discussion that in many existing schemes, it is possible to replace such hash functions with a target collision resistant hash function $H : \{0,1\}^* \to \{0,1\}^k$, where k is the security parameter. We think that it is very useful and could drastically save the cost for the hash function implementation in many practical cryptographic schemes.

key words: *CCA-secure public-key encryption, group-dependent hash*

1. Introduction

1.1 Background

Chosen-ciphertext security (CCA-security, for short) [7], [13] is nowadays considered as a standard notion of security for public-key encryption in practice. Furthermore, this security also implies universally composable security [4].

It is well-known that for almost all of practical CCA-secure scheme, hash functions which depend on the underlying cyclic group are necessary, and this means that a dedicated hash function could be required for each public key. For example, in the Cramer-Shoup cryptosystem [5], a target collision resistant hash function $H : \{0,1\}^* \to \mathbb{Z}_p$, where p is the order of the underlying cyclic group. For the ease of explanation, we call such kind of hash function that output depends on group as, *group-dependent* hash function.

Cost of Group-dependent Hash Function. Although it is yet safe to follow the instructions of standardization, it is costly and more complicated to choose a carefully designed group-dependent hash function. More precisely, it is worthwhile noting that it is not trivial at all, to smoothly map the cryptographic hash function output to certain group. An inappropriate implementation probably results in a serious security hole. For example,

- if one lets $H(x) = H'(x) \bmod p$ where $H' : \{0,1\}^* \to \{0,\ldots,2^{|p|}-1\}$, then $\Pr_x[H(x) = X]$ for any $X \in \{0,\ldots,2^{|p|}-p-1\}$ becomes twice as $\Pr_x[H(x) = X']$ for any $X' \in \{2^{|p|}-p,\ldots,p-1\}$, assuming that the distribution of $H'(x)$ is uniform over $\{0,\ldots,2^{|p|}-1\}$.
- if the hash function $H'(x)$ which distribution is uniform over $\{0,\ldots,2^{|p|}-1\}$ is simply truncated to $\{0,\ldots,p-1\}$ to get a new hash $H(x)$, then the distribution of the truncated hash $H(x)$ will change and need to be carefully dealt with, which sometime is likely to use a dedicated hash function.

Furthermore, even though the distribution of the hash output could be considered as acceptable in some applications (e.g. set the value of the hash function modulo p carefully), implementing such a dedicated hash function apparently takes more cost. See the following example, which is used in IETF working drafts [11]. It shows that how expensive it is to build a group-dependent hash function, rather than a regular hash function (In [11], $f_1(\cdot)$ is required to output to 128-bit sequence, while we do not need this condition).

$$f_1(x) := \mathrm{SHA}-1(x)$$
$$f_2(x,j) := f_1(<[1]|x>)|\ldots|f_1(<[j]|x>)$$
$$kdf(x) := f_1(<len(x)|x>)$$
$$hash(x) := (f_2(<len(x)|x>,9)/(N-1)) + 1$$

In the above description, "|" denotes the concatenation, and the function $len(x)$ denotes the bit length of x in a high order.

The $kdf(\cdot)$ and $hash(\cdot)$ function are both based on SHA-1, where the $hash(\cdot)$ is the group-dependent hash function, i.e. with output in $\mathbb{Z}_p$. It is clear to see that an iterative operation has been used to make the distribution of output of $hash(\cdot)$ more uniformly over $\mathbb{Z}_p$, say 9 times. In addition, a modular calculation is also needed. On the other hand, $kdf(\cdot)$ could be simply achieved by SHA-1 with padding, because it does not need the group-dependence. Therefore, it is easy to see from the underlying example of implementation that a group-dependent hash function is much more costly to build rather than a hash function with output of fixed length, say $\{0,1\}^k$.

Hash-free Construction. Besides, though it is noted to possibly design a hash-free CCA-secure encryption in [5], [6],

Manuscript received August 4, 2008.

[†]The authors are with the Research Center for Information Security (RCIS), National Institute of Advanced Industrial Science & Technology (AIST), Tokyo, 101–0021 Japan.

[††]The authors are with the Chuo University, Tokyo, 112–8551 Japan.

a) E-mail: y-cui@aist.go.jp

DOI: 10.1587/transinf.E92.D.967

it is generally not practical because, 1). an injective mapping in replacement of TCR hash function is not always available,

> "We cannot in general expect to find an easy-to-compute, injective map from $\mathbb{G}$ to $\mathbb{Z}_p$"

(cited from [6], Sect. 6.4); 2). the proposed hash-free variants of CS [5], [6] is not as efficient as the CS encryption implemented with hash function.

Therefore, it is very helpful in practice eliminating necessity of such group-dependent hash functions.

1.2 Main Result

In this short note, we show that it is possible to replace group-dependent hash functions in popular public-key encryption schemes with standard ones such as $H : \{0,1\}^* \to \{0,1\}^k$, where k is the security parameter ($2^k - 1$ is not necessarily to be a prime).

More specifically, we demonstrate that it is safe to use target collision resistant (TCR) hash functions $H : \{0,1\}^* \to \{0,1\}^k$ instead of TCR hash functions $H : \{0,1\}^* \to \mathbb{Z}_p$ in Cramer-Shoup (CS) [5], Kurosawa-Desmedt (KD) [10], Boyen-Mei-Waters (BMW) [3], and Kiltz [9], where $|p| = 2k$. It will be of importance for secure hash function implementations.

2. Preliminaries

2.1 Target Collision Resistant Hash Functions

TCR [6] is aimed to thwart the second pre-image collision attack, and is a special kind of *Universal One-Way Hash Function (UOWHF)* [12]. More precisely, UOWHF assumes that adversary commits a "target" input which is independent of hash function's key at the beginning, whereas the input of TCR is a random tuple of group element. Hence, it is obvious to see that UOWHF suffices to achieve the TCR. In particular, any secure implementable UOWHF hash (such as SHA-1, SHA-2 family) could be used to implement TCR hash function.

2.2 Decisional Diffie-Hellman Assumption

Informally, Decisional Diffie-Hellman (DDH) assumption means the inability to distinguish two quadruples $\mathcal{R}, \mathcal{T}$ in $\mathbb{G}$, s.t. $\mathcal{R} = \{g, g^x, g^y, g^z\}$ and $\mathcal{T} = \{g, g^x, g^y, g^{xy}\}$, where g is a random generator of $\mathbb{G}$, and random $x, y, z \in \mathbb{Z}_p$.

2.3 Cramer-Shoup PKE [5]

The Cramer-Shoup (CS) encryption [5], [6] is the first efficient CCA secure public-key encryption in the standard model, i.e. without random oracle model [1]. We recall the CS† scheme **(Gen, Enc, Dec)** in the following.

Gen: On input of secret parameter k, produce random elements $g_1, g_2 \in \mathbb{G}$, and $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_p$. Then compute the following,

$$c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2}, h = g_1^z$$

A group-dependent hash function $\mathbf{H}$ is randomly chosen from TCR hash family, *where the hash functions map long bit strings to elements of* $\mathbb{Z}_p$.

$$pk = (g_1, g_2, c, d, h, \mathbf{H}), sk = (x_1, x_2, y_1, y_2, z)$$

Enc: For $m \in \mathbb{G}$, choose $r \in \mathbb{Z}_p$ at random, compute

$$u_1 = g_1^r,\ u_2 = g_2^r,\ e = h^r m,$$
$$\alpha = \mathbf{H}(u_1, u_2, e),\ v = (cd^\alpha)^r$$

Output ciphertext (u_1, u_2, e, v).

Dec: Given a ciphertext (u_1, u_2, e, v), compute $\alpha = \mathbf{H}(u_1, u_2, e)$, and check the consistency of v if

$$u_1^{x_1 + y_1\alpha} u_2^{x_2 + y_2\alpha} = v$$

If the above holds, output $m = e/u_1^z$, otherwise output $\perp$.

3. Group-Independence of Hash Function

Now we will investigate the output of TCR hash and show that it is possibly group-independent. To show the sufficiency of a secure TCR hash in $\{0,1\}^k$ rather than in $\mathbb{Z}_p$, we have to prove that 1). TCR hash in $\{0,1\}^k$ keeps the whole security as the same as 2^k; 2). there is no obstacle to remove the group-dependent hash functions.

At first, it is apparent that in the encryption process, the hash output α only appears in the exponent $x_1 + y_1\alpha$, $x_2 + y_2\alpha$ and $r\alpha$, as a multiple of y_1, y_2, r which are all uniformly distributed over $\mathbb{Z}_p$. Besides, since α is easy to know by computing public hash function $\mathbf{H}$, it will be no influence to constrain $\alpha \in \{0,1\}^k$.

Simulation Strategy of CS. We next focus on the security proof. The CS encryption relies on DDH assumption and security of TCR hash function. Construct a simulator who mimics the joint distribution of both adversary's view and encryption oracle output. According to the input (g_1, g_2, u_1, u_2), the simulator randomly chooses $(x_1, x_2, y_1, y_2, z_1, z_2) \in \mathbb{Z}_p^6$ and computes

$$c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2}, h = g_1^{z_1} g_2^{z_2}$$

Referring to the proof of CS scheme [5], [6], it is simply good for decryption oracle to reject all invalid ciphertexts, except with negligible probability. The situation is divided in two cases, corresponding to the input of simulator (g_1, g_2, u_1, u_2), which is from either distribution of $\mathcal{R}$, or distribution of $\mathcal{T}$.

Case 1. If (g_1, g_2, u_1, u_2) is from distribution of $\mathcal{T}$, an adversary will solve the DDH problem with non-negligible probability, because the simulation is almost perfect.

†The binary output of hash function should be converted into integer when computing the exponentiation [14]. For the simplicity of explanation, we take it implicitly included in the algorithm.

Case 2. If (g_1, g_2, u_1, u_2) is from distribution of $\mathcal{R}$, an adversary's successful probability will be exactly the same as the random guessing, because the random coin b is completely independent of his view.

Claim 1. In Case 1., the security holds the same for $\alpha \in \mathbb{Z}_p$ and $\alpha \in \{0,1\}^k$.

Proof. In the first case, outputs of encryption oracle have right forms. Let $u_1 = g_1^r$, $u_2 = g_2^r$, where random r is from $\mathbb{Z}_p$, there is

$$u_1^{x_1} u_2^{x_2} = c^r, u_1^{y_1} u_2^{y_2} = d^r, u_1^{z_1} u_2^{z_2} = h^r$$

and $v = c^r d^{r\alpha}$. Since $r \in \mathbb{Z}_p$, a multiple of r is also in $\mathbb{Z}_p$. Hence, consider an exponent $r \times \alpha$ of d, it does not make great difference for $\alpha \in \{0,1\}^k$, or $\alpha \in \mathbb{Z}_p$.

Again, we have a look at rejection of decryption oracle. The proof given by CS scheme [5], [6] has an analysis of the distribution of point $\mathbf{P} = (x_1, x_2, y_1, y_2) \in \mathbb{Z}_p^4$ conditioned on the adversary's view. Denote $\log(\cdot)$ as $\log_{g_1}(\cdot)$, and $w = \log g_2$. From the public key, it is easy to see

$$\log c = x_1 + w x_2 \tag{1}$$

and

$$\log d = y_1 + w y_2 \tag{2}$$

which further implies hyperplane $\mathcal{H}$ defined by (3), because v has a right distribution.

$$\log v = r x_1 + w r x_2 + \alpha(r y_1 + r w y_2) \tag{3}$$

Assume that adversary could submit an invalid ciphertext (u_1', u_2', v', e'), where $\log u_1' = r_1'$, $\log u_2' = w r_2'$ and $r_1' \neq r_2'$. The decryption oracle will reject the invalid ciphertext, unless the hyperplane $\mathcal{H}'$ defined by (4) contains $\mathbf{P}$.

$$\log v' = r_1' x_1 + w r_2' x_2 + \alpha'(r_1' y_1 + r_2' w y_2) \tag{4}$$

This will happen in the line which $\mathcal{H}$ and $\mathcal{H}'$ intersect, where adversary can only hope to find the point $\mathbf{P}$ with probability $1/p$ (in his first query). Since all above parameters are in $\mathbb{Z}_p$, and α and α' can also be computed by adversary, it is obvious to see that changing $\alpha, \alpha' \in \mathbb{Z}_p$ to $\alpha, \alpha' \in \{0,1\}^k$ will not degrade the security than we expect. □

Claim 2. In Case 2., the security holds the same for $\alpha \in \mathbb{Z}_p$ and $\alpha \in \{0,1\}^k$, when $|p| = 2k$.

Proof. In the second case, outputs of encryption oracle have incorrect forms with overwhelming probability, thus, let $u_1 = g_1^{r_1}, u_2 = g_1^{w r_2}$, where $r_1 \neq r_2$.

Next we focus on the part that TCR hash $\mathbf{H}$ involved. Similar to the analysis in Case 1., the distribution of $\mathbf{P} = (x_1, x_2, y_1, y_2) \in \mathbb{Z}_p^4$, is on the line defined by hyperplanes (3) and

$$\log v = r_1 x_1 + w r_2 x_2 + \alpha(r_1 y_1 + r_2 w y_2) \tag{5}$$

Now suppose that adversary submits an invalid ciphertext $(u_1', u_2', v', e') \neq (u_1, u_2, v, e)$, then there is, $\log u_1' = r_1', \log u_2' = w r_2'$.

- When $(u_1', u_2', e') = (u_1, u_2, e)$, but $v \neq v'$. This leads to an immediate rejection.
- When $(u_1', u_2', e') \neq (u_1, u_2, e)$ and $\alpha' = \alpha$. This leads to find a collision of TCR hash function, where $\mathbf{H}(u_1', u_2', e') = \mathbf{H}(u_1, u_2, e)$
- When $(u_1', u_2', e') \neq (u_1, u_2, e)$ and $\alpha' \neq \alpha$. Equations (1), (2), (4), (5) are linearly independent, and construct matrix $\mathbf{M}$ over $\mathbb{Z}_p$.

$$\det \begin{pmatrix} 1 & w & 0 & 0 \\ 0 & 0 & 1 & w \\ r_1' & w r_2' & \alpha' r_1' & \alpha' w r_2' \\ r_1 & w r_2 & \alpha r_1 & \alpha w r_2 \end{pmatrix}$$
$$= w^2 (r_2 - r_1)(r_2' - r_1')(\alpha - \alpha') \neq 0$$

In the underlying proof, for α, α' in a different range, the only change that may deny the use of $\mathbf{M}$ over $\mathbb{Z}_p$, is that $\alpha, \alpha' \in \{0,1\}^k$, s.t.

$$\begin{cases} \alpha - \alpha' \neq 0 \\ \alpha - \alpha' \equiv 0 \pmod{p} \end{cases} \tag{6}$$

However, since $|p| > k$, Eq. (6) has no solution. As a consequence, for TCR hash $\mathbf{H}$, a output range of $\{0,1\}^k$ ($|p| = 2k$), appears good for the CS proof. It is not necessarily to restrict $\mathbf{H}$ to $\mathbb{Z}_p$. □

4. Extensions

Inspired by CS scheme, several CCA-secure encryptions, such as, KD [10], BMW [3], Kiltz [9], were proposed in the standard model. We examine in the following that all these schemes are possibly implemented without group-dependent hash function.

KD. In KD scheme [10], there is no explicit description of TCR hash function output, but has implicitly assumed that TCR is mapped to the $\mathbb{Z}_p$ ([10], Sect. 4.3). KD scheme has a similar construction and makes use of independent linear equations over $\mathbb{Z}_p$ as in CS proof. According to previous analysis in proof of *Claim 2.*, we can conclude that Eq. (6) has no solution when $|p| = 2k$. Hence, to change the output of TCR hash is not going to degrade the security of KD scheme.

BMW. In their paper (Sect. 4.1, [3]), a (target) collision resistant hash function $H_s(g^c) = w$ mapping to $\mathbb{Z}_p$ is employed in the key encapsulation scheme. Analogously to the analysis in Sect. 3, we first investigate that it takes at least 2^k complexity to break the hash function just according to the security definition of TCR, because unlike the collision resistant hash function, the birthday attack [15] does not make difference for TCR. Then we would like to explain that why the security proof does not need group-dependent hash, in the following.

Let us recall BMW's security proof. It appears a resemble of the Boneh-Boyen selective identity-based encryption [2]. The key point of the proof is that a target value w^* is selected by the adversary priorly, and the simulator can use

some algebraic tricks to cope with all decryption queries to w except the target value w^*. In selective-identity-based encryption scheme, the target value w^* is set to be unavailable, while in BMW scheme, this property relies on the target collision resistance of hash function.

The answer corresponding to query w is,

$$\begin{aligned} K &= \frac{e\left(C_1, (h^b)^{\frac{-\delta}{w-w^*}} v_1 v_2^w\right)}{e\left(C_2, (h^b)^{\frac{-1}{w-w^*}} h\right)} \\ &= \left(\frac{e(g,h)^{\delta + a(w-w^*) - b\frac{\delta}{w-w^*}}}{e(g,h)^{\delta + a(w-w^*) - b\frac{\delta}{w-w^*} - ab\frac{w-w^*}{w-w^*}}} \right)^t \\ &= e(g,h)^{abt} \end{aligned} \tag{7}$$

Note that all of the exponents in Eq. (7) will be kept in $\mathbb{Z}_p$, since we have set $|p| = 2k$. More importantly, the situation $w - w^* \neq 0$, but $w - w^* \equiv 0 \pmod{p}$ will not happen, because $|p| > k$.

Kiltz. Kiltz [9] presented a simple CCA-secure encryption scheme based on a security assumption different from what CS [5] and KD [10] used. With a similar analysis, we focus on the security proof of his scheme, where the simulation must answer queries from the adversary with high successful probability.

More precisely, the corresponding answer (Sect. 4.2, [9]) is

$$K = H((\pi / c^d)^{\frac{1}{t-t^*}}) \tag{8}$$

This Eq. (8) helps the simulator answer the queries, if $t - t^* \neq 0$. According to our analysis above, it is easy to see that $t - t^* \neq 0$, but $t - t^* \equiv 0 \pmod{p}$ does not happen. Hence, we prove that our change to the range of hash function will destroy the provable security of those schemes.

Acknowledgement

Yang Cui is supported by the Japan Society for the Promotion of Science (JSPS) Postdoctoral Fellowship.

References

[1] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," Proc. CCS'93, pp.62–73, 1993.
[2] D. Boneh and X. Boyen, "Efficient selective-ID secure identity-based encryption without random oracles," Proc. EUROCRYPT 2004, pp.223–238, 2004.
[3] X. Boyen, Q. Mei, and B. Waters, "Direct chosen ciphertext security from identity-based techniques," Proc. CCS'05, pp.320–329, 2005.
[4] R. Canetti, "Universally composable security: A new paradigm for cryptographic protocols," Proc. FOCS'01, pp.136–145, 2001.
[5] R. Cramer and V. Shoup, "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack," Proc. Crypto'98, pp.13–25, 1998.
[6] R. Cramer and V. Shoup, "Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack," SIAM J. Comput., vol.33, pp.167–226, 2003.
[7] D. Dolev, C. Dwork, and M. Naor, "Non-malleable cryptography," Proc. STOC'91, pp.542–552, 1991.
[8] S. Goldwasser and S. Micali, "Probabilistic encryption," J. Comput. Syst. Sci., vol.28, no.2, pp.270–299, 1984.
[9] E. Kiltz, "Chosen-ciphertext secure key-encapsulation based on gap hashed Diffie-Hellman," Proc. PKC'07, pp.282–297, 2007.
[10] K. Kurosawa and Y. Desmedt, "A new paradigm of hybrid encryption scheme," Proc. Crypto'04, pp.426–442, 2004.
[11] P. MacKenzie, "PAK: Password-authenticated key exchange for iSCSI," http://ietfreport.isoc.org/all-ids/draft-mackenzie-ips-iscsi-pak-00.txt
[12] M. Naor and M. Yung, "Universal one-way hash functions and their cryptographic applications," Proc. STOC'89, pp.33–43, 1989.
[13] C. Rackoff and D.R. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack," Proc. Crypto'91, pp.433–444, 1991.
[14] V. Shoup, "A proposal for an ISO standard for public key encryption (version 2.0)," A preliminary version of ISO 18033-2: A Standard for Public-Key Encryption, Available at http://www.shoup.net/, 2001.
[15] D. Wagner, "A generalized birthday problem," Proc. Crypto 2002, pp.288–303, 2002.