Short Exponent Diffie-Hellman Problems

Takeshi Koshiba[12] and Kaoru Kurosawa[3]

[1] Secure Computing Lab., Fujitsu Laboratories Ltd.
[2] ERATO Quantum Computation and Information Project,
Japan Science and Technology Agency,
Matsuo Bldg.2F, 406 Iseyacho, Kawaramachi Marutamachi,
Kamigyo-ku, Kyoto 602-0873, Japan
`koshiba@acm.org`
[3] Department of Computer and Information Sciences, Ibaraki University,
4-12-1 Nakanarusawa, Hitachi, Ibaraki 316-8511, Japan
`kurosawa@cis.ibaraki.ac.jp`

Abstract. In this paper, we study *short* exponent Diffie-Hellman problems, where significantly many lower bits are zeros in the exponent. We first prove that the decisional version of this problem is as hard as two well known hard problems, the standard decisional Diffie-Hellman problem (DDH) and the short exponent discrete logarithm problem. It implies that we can improve the efficiency of ElGamal scheme and Cramer-Shoup scheme under the two *widely accepted* assumptions. We next derive a similar result for the computational version of this problem.

1 Introduction

The discrete logarithm (DL) problem and the Diffie-Hellman (DH) problems are basis of many applications in modern cryptography.

1.1 Previous works on DL problem

Blum and Micali [1] presented the first cryptographically secure pseudo-random bit generators (PRBG) under the DL assumption over Z_p^*, where p is a prime. Long and Wigderson [6], and Peralta [9] showed that up to $O(\log\log p)$ pseudo-random bits can be extracted by a single modular exponentiation of the Blum-Micali generator.

The discrete logarithm with short exponent (DLSE) assumption is also useful. It claims that the DL problem is still hard even if the exponent is small. Van Oorschot and Wiener studied under what condition the DLSE assumption remains difficult (Their concern was to speed-up the key agreement method of Diffie-Hellman) [12]. They showed that the known attacks are precluded if safe primes p are used for Z_p^* (that is, $p-1=2q$ for a prime q) or prime-order groups are used. Especially, the latter is highly recommended. Under the DLSE assumption, Patel and Sundaram [8] showed that it is possible to extract up to $n-\omega(\log n)$ bits from one iteration of the Blum-Micali generator by using safe primes p, where n is the bit length of p. Gennaro [4] further improved this result in such a way that each full modular exponentiation can be replaced with a short modular exponentiation.

1.2 Our contribution on DH problems

Let G_q be a finite Abelian group of prime order q. Let g be a generator, that is, $G_q = \langle g \rangle$. Then the computational Diffie-Hellman (CDH) problem is to compute g^{ab} from (g, g^a, g^b). The decisional Diffie-Hellman (DDH) problem is to distinguish between (g, g^a, g^b, g^{ab}) and (g, g^a, g^b, g^c), where a, b and c are uniformly and randomly chosen from Z_q.

	Short exp $\approx$ Full exp	Short DL
$\lvert Z_p^* \rvert =$ even	Gennaro [4]	Application to PRBG [8, 4]

Table 1. Previous works over Z_p^*

	Short $\approx$ Full	Short exp. DDH	Short exp. CDH
$\lvert G_q \rvert =$ prime	This paper	DDH+Short DL Application to encryption	CDH+Short DL Application to OT

Table 2. Our work over G_q

In this paper, we study *short* exponent variants of the DDH problem and the CDH problem over G_q, where significantly many lower bits are zeros in the exponent. More precisely, the short exponent DDH problem has two sub-problems, a (Short, Full)-DDH problem in which a is small, and a (Short, Short)-DDH problem in which both a and b are small. The short exponent CDH problem has two sub-problems, similarly.

We first prove that each of the short exponent DDH problems is as hard as two well known hard problems, the standard DDH problem and the DLSE problem. That is, we show our equivalence:

$$\text{(Short, Full)-DDH} \Longleftrightarrow \text{(Short, Short)-DDH} \Longleftrightarrow DDH + DLSE.$$

To prove these equivalence, we show that short exponents $\{g^s \mid s \text{ is small}\}$ and full exponents $\{g^x \mid x \in Z_q\}$ are indistinguishable under the DLSE assumption over prime-order groups G_q. A similar result was proved for Z_p^* by Gennaro [4] based on [8], where p is a safe prime. Our proof shows that the indistinguishability can be proved much simpler over G_q than over Z_p^*. (Remember that prime-order groups are highly recommended for the DLSE assumption by van Oorschot and Wiener [12]. It is also consistent with the DDH problem which is defined over prime-order groups.)

Our result implies that we can improve the efficiency of ElGamal encryption scheme and Cramer-Shoup encryption scheme directly under the two widely accepted assumptions, the DDH assumption and the DLSE assumption. Indeed, we present such variants of ElGamal scheme and Cramer-Shoup scheme. They are much faster than the original encryption algorithms because short exponents are used instead of full exponents. (Remember that under the DDH assumption, ElGamal encryption scheme [3] is secure in the sense of indistinguishability against chosen plaintext attack (IND-CPA) and Cramer-Shoup scheme [2] is secure in the sense of indistinguishability against chosen ciphertext attack (IND-CCA).)

We next show a similar result for the CDH problem. That is, we prove the equivalence such that

$$\text{(Short, Full)-CDH} \Longleftrightarrow \text{(Short, Short)-CDH} \Longleftrightarrow CDH + DLSE.$$

This result implies that we can improve the efficiency of the oblivious transfer protocols of [7, 5] under the CDH assumption plus the DLSE assumption.

We believe that there will be many other applications of our results.

2 Preliminaries

2.1 Notation

$|x|$ denotes the bit length of x. $x \in_R X$ means that x is randomly chosen from a set X. We sometimes assume the uniform distribution over X. Throughout the paper, an "efficient algorithm" means a probabilistic polynomial time algorithm.

Let n denote the bit length of q, where q is the prime order of G_q. Let $c = \omega(\log n)$. It means that 2^c grows faster than any polynomial in n. Let $lsb_k(z)$ be the function that returns the least significant k bits of z and $msb_k(z)$ the function that returns the most significant k bits of z. If we write $b = msb_k(z)$, we sometimes mean that the binary representation of b is $msb_k(z)$.

2.2 Discrete Logarithm with Short Exponent (DLSE) Assumption

Let $f(g, z) = (g, g^z)$, where g is a generator of G_q. The discrete logarithm (DL) problem is to compute the inverse of f. The DL assumption says that the DL problem is hard.

We next define the discrete logarithm with short exponent (DLSE) problem as follows. Let $f^{\text{se}}(g, u||0^{n-c}) = (g, g^{u||0^{n-c}})$, where $|u| = c$ and $||$ denotes concatenation. That is, the exponent of $g^{u||0^{n-c}}$ is short. Then the DLSE problem is to compute the inverse of f^{se}. The DLSE assumption says that the DLSE problem is hard. Formally,

Assumption 1. *(DLSE assumption)* There exists no efficient algorithm which solves the DLSE problem with non-negligible probability.

3 Short EXP ≈ Full EXP

In this section, we prove that full exponents and short exponents are indistinguishable under the DLSE assumption. More formally, define A_0 and A_{n-c} as

$$A_0 = \{(g, g^x) \mid x \in R_0\} \quad \text{and} \quad A_{n-c} = \{(g, g^x) \mid x \in R_{n-c}\},$$

where

$$R_0 = \{u \mid 0 \le u < q\} \quad \text{and} \quad R_{n-c} = \{2^{n-c}u \mid 0 \le 2^{n-c}u < q\}.$$

Theorem 1. *A_0 and A_{n-c} are indistinguishable under the DLSE assumption.*

A proof is given in Appendix A. We show a sketch of the proof here. For $1 \le i \le n-c$, let

$$A_i = \{(g, g^x) \mid x \in R_i\}, \text{ where } R_i = \{2^i u \mid 0 \le 2^i u < q\}.$$

Suppose that there exists a distinguisher D which can distinguish A_{n-c} from A_0. Then by using a hybrid argument, there exists j such that A_j and A_{j+1} are distinguishable.

We will show that (i) the j can be found in polynomial time and (ii) the DLSE problem can be solved by using the (D, j). We briefly sketch below how to solve the DLSE problem by using the (D, j). (Remember that the DLSE problem is to find x from (g, g^x) in A_{n-c}.)

1. The difference between A_j and A_{j+1} appears in the $(j+1)$-th least significant bit b_{j+1} of exponents x. That is,
$$b_{j+1} = \begin{cases} 1 \text{ } \mathit{if} \text{ } (g, g^x) \in A_j \setminus A_{j+1} \\ 0 \text{ } \mathit{if} \text{ } (g, g^x) \in A_{j+1} \end{cases}$$
Hence we can show that (D, j) can be used as a prediction algorithm of b_{j+1}.
2. We can compute $g^{x/2}$ from g^x because the order of G_q is a prime q. This enables us to use (D, j) to predict all higher bits of x as well as b_{j+1} (except several most significant bits γ).
3. Suppose that $(g, y) \in A_{n-c}$ is given, where $y = g^{v||0^{n-c}}$. In order to find the $b_1 = lsb_1(v)$, we carefully randomize y so that the exponent is uniformly distributed over R_j. For this randomization, we need to search some most significant bits γ of v exhaustively, but in polynomial time.
4. After all, by taking the majority vote, we can find $b_1 = lsb_1(v)$ with overwhelming probability. Next let
$$y_1 = (y(g^{2^{n-c}})^{-b_1})^{1/2} = g^{0||v'||0^{n-c}},$$
where $v = v'||b_1$. Applying the same process, we can find $lsb_1(v')$ similarly. By repeating this algorithm, we can finally find v with overwhelming probability.

4 (*Short*, Full)-DDH = Standard DDH + DLSE

The standard DDH assumption claims that

$$B_0 = \{(g, g^x, g^y, g^{xy}) \mid x \in Z_q, y \in Z_q\} \quad \text{and}$$
$$C_0 = \{(g, g^x, g^y, g^z) \mid x \in Z_q, y \in Z_q, z \in Z_q\}$$

are indistinguishable.

We now define the (Short, Full)-DDH assumption as follows. Let

$$B_{n-c} = \{(g, g^x, g^y, g^{xy}) \mid x \in R_{n-c}, y \in Z_q\} \quad \text{and}$$
$$C_{n-c} = \{(g, g^x, g^y, g^z) \mid x \in R_{n-c}, y \in Z_q, z \in Z_q\},$$

where $c = \omega(\log n)$ with $n = |q|$. The (Short, Full)-DDH assumption claims that B_{n-c} and C_{n-c} are still indistinguishable. Note that x is short and y is of full length.

We then prove the (Short, Full)-DDH assumption is equivalent to the standard DDH assumption and the DLSE assumption. We first show that the standard DDH assumption and the DLSE assumption implies the (Short, Full)-DDH assumption.

Theorem 2. *Suppose that the DDH assumption and the DLSE assumption are true. Then the (Short, Full)-DDH assumption is true.*

Proof. From Theorem 1, A_0 and A_{n-c} are indistinguishable under the DLSE assumption, where

$$A_0 = \{(g, g^x) \mid x \in R_0\} \quad \text{and} \quad A_{n-c} = \{(g, g^x) \mid x \in R_{n-c}\}.$$

First it is clear that C_0 and C_{n-c} are indistinguishable because y and z are random independently of x.

Next we prove that B_0 and B_{n-c} are indistinguishable. Suppose that there exists a distinguisher D which distinguishes B_{n-c} from B_0. Then we show that there exists a distinguisher D' which distinguishes A_{n-c} from A_0. On input (g, g^x), D' chooses $y \in Z_q$ at random and computes g^y and $(g^x)^y$. D' then gives $(g, g^x, g^y, (g^x)^y)$ to D. Note that

$$(g, g^x, g^y, (g^x)^y) \in_R \begin{cases} B_0 & \textit{if } (g, g^x) \in_R A_0, \\ B_{n-c} & \textit{if } (g, g^x) \in_R A_{n-c}. \end{cases}$$

D' finally outputs the output bit of D. Then it is clear that D' can distinguish A_{n-c} from A_0. However, this is against Theorem 1. Hence B_0 and B_{n-c} are indistinguishable.

Consequently we obtain that $B_{n-c} \approx B_0 \approx C_0 \approx C_{n-c}$, where $\approx$ means indistinguishable. ($B_0 \approx C_0$ comes from the standard DDH assumption.) Therefore, B_{n-c} and C_{n-c} are indistinguishable. □

We next show that the (Short, Full)-DDH assumption implies the standard DDH assumption and the DLSE assumption.

Theorem 3. *Suppose that the (Short, Full)-DDH assumption is true. Then the DDH assumption and the DLSE assumption are true.*

Proof. First suppose that there exists an efficient algorithm M which can solve the DLSE problem with some non-negligible probability ϵ. Then we show that there exists a distinguisher D between B_{n-c} and C_{n-c}.

On input (g, g^x, g^y, α), D gives g^x to M. If M does not output x correctly, then D outputs a random bit b. Suppose that M outputs x correctly. Then D outputs b such that

$$b = \begin{cases} 1 \text{ } \mathit{if} \text{ } \alpha = (g^y)^x \\ 0 \text{ } \mathit{if} \text{ } \alpha \neq (g^y)^x. \end{cases}$$

Then it is easy to see that D distinguishes between B_{n-c} and C_{n-c}.

Next suppose that there exists a distinguisher D_0 which breaks the DDH assumption. Then we show that there exists a distinguisher D_1 which breaks the (Short, Full)-DDH assumption.

Let (g, g^x, g^y, g^a) be an input to D_1, where $a = xy \bmod q$ or random. D_1 chooses $r \neq 0$ at random and gives $(g, (g^x)^r, g^y, (g^a)^r)$ to D_0. It is easy to see that

$$(g, (g^x)^r, g^y, (g^a)^r) \in_R \begin{cases} B_0 \text{ } \mathit{if} \text{ } (g, g^x, g^y, g^a) \in_R B_{n-c}, \\ C_0 \text{ } \mathit{if} \text{ } (g, g^x, g^y, g^a) \in_R C_{n-c}. \end{cases}$$

Finally D_1 outputs the output bit of D_0. Then it is clear that D_1 distinguishes between B_{n-c} and C_{n-c}. □

From Theorem 2 and Theorem 3, we obtain the following corollary.

Corollary 1. *The (Short, Full)-DDH assumption is equivalent to both the DDH assumption and the DLSE assumption.*

5 Extension to (Short, *Short*)-DDH

We define the (Short, *Short*)-DDH assumption as follows. Let

$$B'_{n-c} = \{(g, g^x, g^y, g^{xy}) \mid x \in R_{n-c}, y \in R_{n-c}\} \quad \text{and}$$
$$C'_{n-c} = \{(g, g^x, g^y, g^z) \mid x \in R_{n-c}, y \in R_{n-c}, z \in Z_q\}.$$

Then the (Short, *Short*)-DDH assumption claims that B'_{n-c} and C'_{n-c} are indistinguishable. Note that both x and y are short in B'_{n-c} and C'_{n-c}.

We first show that the (Short, Full)-DDH assumption implies the (Short, Short)-DDH assumption.

Theorem 4. *Suppose that the (Short, Full)-DDH assumption is true. Then the (Short, Short)-DDH assumption is true.*

Proof. First suppose that the (Short, Full)-DDH assumption is true. From Theorem 3, both the DLSE assumption and the DDH assumption are true. From Theorem 1, A_0 and A_{n-c} are indistinguishable. Then it is clear that C_{n-c} and C'_{n-c} are indistinguishable because x and z are random independently of y.

Next we prove that B_{n-c} and B'_{n-c} are indistinguishable. Suppose that there exists a distinguisher D which distinguishes B_{n-c} and B'_{n-c}. Then we show that there exists a distinguisher D' which distinguishes A_{n-c} from A_0. On input

(g, g^y), D' chooses $x \in R_{n-c}$ at random and computes g^x and $(g^y)^x$. D' then gives $(g, g^x, g^y, (g^y)^x)$ to D. Note that

$$(g, g^x, g^y, (g^x)^y) \in_R \begin{cases} B_{n-c} \text{ if } (g, g^x) \in_R A_0, \\ B'_{n-c} \text{ if } (g, g^x) \in_R A_{n-c}. \end{cases}$$

D' finally outputs the output bit of D. Then it is clear that D' can distinguish A_{n-c} from A_0. However, this contradicts that A_0 and A_{n-c} are indistinguishable. Hence B_{n-c} and B'_{n-c} are indistinguishable.

Consequently we obtain that

$$B'_{n-c} \approx B_{n-c} \approx C_{n-c} \approx C'_{n-c},$$

where $\approx$ means indistinguishable. Therefore, B'_{n-c} and C'_{n-c} are indistinguishable. □

We next show that the (Short, Short)-DDH assumption implies the (Short, Full)-DDH assumption.

Theorem 5. *Suppose that the (Short, Short)-DDH assumption is true. Then the (Short, Full)-DDH assumption is true.*

Proof. First suppose that the (Short, Full)-DDH assumption is false. Then, from Theorem 2, either the DDH assumption or the DLSE assumption is false.

Further suppose that the DLSE assumption is false. That is, there exists an efficient algorithm M which can solve the DLSE problem with some non-negligible probability ϵ. Then we show that there exists a distinguisher D between B'_{n-c} and C'_{n-c}.

On input (g, g^x, g^y, α), D gives g^x to M. If M does not output x correctly, then D outputs a random bit b. Suppose that M outputs x correctly. Then D outputs b such that

$$b = \begin{cases} 1 \text{ if } \alpha = (g^y)^x \\ 0 \text{ if } \alpha \neq (g^y)^x. \end{cases}$$

Then it is easy to see that D distinguishes between B'_{n-c} and C'_{n-c}.

Next suppose that the DDH assumption is false. That is, there exists a distinguisher D_0 which breaks the DDH assumption. Then we show that there exists a distinguisher D_1 which breaks the (Short, Short)-DDH assumption.

Let (g, g^x, g^y, g^a) be an input to D_1, where $a = xy \bmod q$ or random. D_1 chooses $r_1, r_2 \neq 0$ at random and gives $(g, (g^x)^{r_1}, (g^y)^{r_2}, (g^a)^{r_1 r_2})$ to D_0. It is easy to see that

$$(g, (g^x)^{r_1}, (g^y)^{r_2}, (g^a)^{r_1 r_2}) \in_R \begin{cases} B_0 \text{ if } (g, g^x, g^y, g^a) \in_R B'_{n-c}, \\ C_0 \text{ if } (g, g^x, g^y, g^a) \in_R C'_{n-c}. \end{cases}$$

Finally D_1 outputs the output bit of D_0. Then it is clear that D_1 distinguishes between B'_{n-c} and C'_{n-c}. □

Corollary 2. *The (Short, Short)-DDH assumption is equivalent to the (Short, Full)-DDH assumption.*

From Corollary 1, we obtain the following corollary.

Corollary 3. *The (Short, Short)-DDH assumption is equivalent to both the DDH assumption and the DLSE assumption.*

6 Short Computational DH

Remember that the computational Diffie-Hellman (CDH) problem is to compute g^{xy} from g, g^x, g^y, where $x, y \in Z_q$. The CDH assumption says that the CDH problem is hard.

In this section, we introduce two variants of the CDH assumption, (Short, Full)-CDH assumption and (Short, Short)-CDH assumption. We then prove that each of them is equivalent to the standard CDH assumption and the DLSE assumption.

Short variants of the CDH assumption are defined as follows.

Assumption 2. *((Short, Full)-CDH assumption)* There exists no efficient algorithm for computing g^{xy} with non-negligible probability from g, g^x, g^y, where $x \in R_{n-c}$ and $y \in Z_q$.

Assumption 3. *((Short, Short)-CDH assumption)* There exists no efficient algorithm for computing g^{xy} with non-negligible probability from g, g^x, g^y, where $x \in R_{n-c}$ and $y \in R_{n-c}$.

We first show that the standard CDH assumption and the DLSE assumption imply the (Short, Full)-CDH assumption.

Theorem 6. *Suppose that the CDH assumption and the DLSE assumption are true. Then the (Short, Full)-CDH assumption is true.*

Proof. Suppose that there exists an efficient algorithm A which computes g^{xy} from g, g^x, g^y such that $x \in R_{n-c}$ and $y \in Z_q$.

If the CDH problem is easy, then our claim holds. Suppose that the CDH problem is hard. We then show an efficient algorithm B which distinguishes between A_0 and A_{n-c}. On input (g, g^x), B chooses $y \in Z_q$ randomly and computes g^y. B gives (g, g^x, g^y) to A. Suppose that A outputs z. B checks if $z = (g^x)^y$.

Now from our assumption, if $(g, g^x) \in A_{n-c}$, then $z = g^{xy}$ with non-negligible probability. If $(g, g^x) \in A_0$, then $z = g^{xy}$ with negligible probability. This means that B can distinguish between A_0 and A_{n-c}. From Theorem 1, this means that the DLSE assumption is false. $\square$

We can prove the converse of Theorem 6 similarly to Theorem 3. Therefore, we obtain the following corollary.

Corollary 4. *(Short, Full)-CDH = Standard CDH + DLSE.*

We next show that the (Short, Short)-CDH assumption is equivalent to the standard CDH assumption and the DLSE assumption.

Theorem 7. *(Short, Short)-CDH = Standard CDH + DLSE.*

The proof is based on the same argument for the (Short, Full)-CDH assumption and the random self-reducibility of the discrete logarithm problem. The details will be given in the final paper.

7 Applications

In this section, we present fast variants of ElGamal encryption scheme and Cramer-Soup encryption scheme. Each variant uses a short random exponent r such that r is essentially c bits long, where $c = \omega(\log |q|)$ and q is the order of the underlying group.

Note that computing g^r requires at most $2c$ modulo multiplications in our variants while it requires at most $2n$ modulo multiplications in the original algorithms. Hence our variants are much faster than the original encryption algorithms.

We can prove their security easily from our results. They are semantically secure under the DDH assumption and the DLSE assumption (i.e., our variant of ElGamal scheme is IND-CPA and our variant of Cramer-Shoup scheme is IND-CCA, respectively). They are one-way under the CDH assumption and the DLSE assumption.

7.1 Security of Public Key Cryptosystem

A public key encryption scheme is called one-way if it is hard to compute the message m from a public key pk and a ciphertext C.

The security in the sense of indistinguishability is defined as follows. Consider the following model of adversaries. In the find stage, the adversary chooses two messages m_0, m_1 on input pk. She then sends these to an encryption oracle. The encryption oracle chooses a random bit b, and encrypts m_b. In the guess stage, the ciphertext C_b is given to the adversary. The adversary outputs a bit b'. We say that the public key cryptosystem is secure in the sense of indistinguishability against chosen plaintext attack (IND-CPA) if $|\Pr(b' = b) - 1/2|$ is negligibly small (as a function of the security parameter).

The security against chosen-ciphertext attack (IND-CCA) is defined similarly except for that the adversary gets the decryption oracle and is allowed to query any ciphertext C, where it must be $C \neq C_b$ in the guess stage.

7.2 (Short, Full) ElGamal Encryption Scheme

ElGamal encryption scheme is (1) one-way under the CDH assumption and (2) IND-CPA under the DDH assumption. Now our variant of ElGamal encryption scheme is described as follows.

(Key generation) Choose a generator G_q and $x \in Z_q$ randomly. Let $\hat{g} = g^{2^{n-c}}, \hat{y} = \hat{g}^x$. The public key is $(\hat{g}, \hat{y})$ and the secret key is x.

(Encryption) Given a message $m \in G$, first choose r such that $2^{n-c}r \in R_{n-c}$ randomly. Next compute $c_1 = \hat{g}^r \ (= g^{2^{n-c}r}), c_2 = m\hat{y}^r$. The ciphertext is (c_1, c_2).

(Decryption) Given a ciphertext (c_1, c_2), compute

$$c_2/c_1^x = m\hat{y}^r/(g^{2^{n-c}r})^x = m(g^{2^{n-c}x})^r/(g^{2^{n-c}r})^x = m.$$

Note that the encryption is very efficient because small r is used. The security is proved as follows.

Theorem 8. *The above scheme is still one-way under the CDH assumption and the DLSE assumption.*

Theorem 9. *The above scheme is still IND-CPA under the DDH assumption and the DLSE assumption.*

7.3 (Short, Full) Cramer-Shoup Encryption Scheme

We next show our variant of Cramer-Shoup scheme.

(Key generation) Choose two generator g_1 and g_2 at random. Also Choose $x_1 x_2, y_1, y_2, z \in Z_q$ randomly. Let $\hat{g}_1 = g_1^{n-c}$ and $\hat{g}_2 = g_2^{n-c}$. Also let

$$\hat{c} = \hat{g}_1^{x_1}\hat{g}_2^{x_2}, \ \hat{d} = \hat{g}_1^{y_1}\hat{g}_2^{y_2}, \ \hat{h} = \hat{g}_1^{z}.$$

The public key is $(\hat{g}_1, \hat{g}_2, \hat{c}, \hat{d}, \hat{h}, H)$ and the secret key is $(x_1 x_2, y_1, y_2, z)$, where H is a randomly chosen universal one-way hash function.

(Encryption) Given a message $m \in G$, first choose r such that $2^{n-c}r \in R_{n-c}$ randomly. Next compute

$$u_1 = \hat{g}_1^r, u_2 = \hat{g}_2^r, e = \hat{h}^r m, \alpha = H(u_1, u_2, e), v = (\hat{c}\hat{d}^\alpha)^r.$$

The ciphertext is (u_1, u_2, e, v).

(Decryption) Given a ciphertext (u_1, u_2, e, v), first compute $\alpha = H(u_1, u_2, e)$ and test if $u_1^{x_1+y_1\alpha}u_2^{x_2+y_2\alpha} = v$. If this condition does not hold, the decryption algorithm outputs "reject". Otherwise, it outputs $m = e/u_1^z$.

The encryption algorithm is very efficient because small r is used. Cramer-Shoup scheme is IND-CCA under the DDH assumption [2]. The proposed scheme is secure under the following assumption.

Theorem 10. *The above scheme is still IND-CCA under the DDH assumption and the DLSE assumption.*

The proof is almost the same as the proof of [2]. We use Corollary 1. The details will be given in the final paper.

7.4 (Short, Short) Versions

We can construct (Short, Short) versions of ElGamal scheme and Cramer-Shoup scheme, and prove their security. The details will be given in the final paper.

References

1. M. Blum and S. Micali: "How to generate cryptographically strong sequences of pseudo-random bits", SIAM J. Computing 13(4), pp.850–864 (1984)
2. R. Cramer and V. Shoup: "A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack", Advances in Cryptology – Crypto'98 Proceedings, Lecture Notes in Computer Science 1462, Springer, pp.13–25 (1998)
3. T. ElGamal: " A public key cryptosystem and a signature scheme based on discrete logarithms", IEEE Trans. Information Theory IT-31(4), pp.469–472 (1985)
4. R. Gennaro: "An improved pseudo-random generator based on discrete log", Advances in Cryptology – Crypto 2000 Proceedings, Lecture Notes in Computer Science 1880, Springer, pp.469–481 (2000)
5. K. Kurosawa and Q. V. Duong: "How to design efficient multiple-use 1-out-n oblivious transfer", IEICE Trans. Fundamentals E87A(1), (2004)
6. D. L. Long and A. Wigderson: "The discrete logarithm hides $O(\log n)$ bits", SIAM J. Computing 17(2), pp.363–372 (1988)
7. M. Naor and B. Pinkas: "Efficient oblivious transfer protocols," Proc. the 12th ACM-SIAM Symposium on Discrete Algorithms (SODA), pp.448–457 (2001)
8. S. Patel and G. S. Sundaram: "An efficient discrete log pseudo random generator", Advances in Cryptology – Crypto'98 Proceedings, Lecture Notes in Computer Science 1462, Springer, pp.304–317 (1998)
9. R. Peralta: "Simultaneous security of bits in the discrete log", Advances in Cryptology – Eurocrypt'85 Proceedings, Lecture Notes in Computer Science 219, Springer, pp.62–72 (1986)
10. J. M. Pollard: "Kangaroos, monopoly and discrete logarithms", J. Cryptology 13(4), pp.437–447 (2000)
11. C. Schnorr: "Security of almost all discrete log bits", Electronic Colloquium on Computational Complexity. TR-98-033. http:/www.eccc.uni-trier.de.eccc/
12. P. C. van Oorschot and M. J. Wiener: "On Diffie-Hellman key agreement with short exponents", Advances in Cryptology – Eurocrypt'96 Proceedings, Lecture Notes in Computer Science 1070, Springer, pp.332–343 (1996)
13. P. C. van Oorschot and M. J. Wiener: "Parallel Collision Search with Cryptographic Applications", J. Cryptology 12(1), pp.1–28 (1999)

Appendix

A Proof of Theorem 1

Before giving a proof of Theorem 1, we show some technical lemmas. Remember that $c = \omega(\log n)$.

Lemma 1. We consider an index i which can be computed in probabilistic polynomial time. Suppose that there exists an efficient algorithm D that on input $(g, g^{u||0^i}) \in_R A_i$, outputs the *lsb* of u with probability $1/2 + \epsilon$, where ϵ is non-negligible. Then for any fixed $g \in G$, there exists an efficient algorithm that on input $g^{u||0^i}$, outputs the *lsb* of u with probability $1/2 + \epsilon$, where $u||0^i \in_R R_i$.

Lemma 1 is easily obtained from the random self-reducibility such that computing z from (g, g^z) is equivalent to computing z from (g^r, g^{rz}). Next let g be a generator of G_q.

Lemma 2. We consider an index i such that $i < n - c$ which can be computed in probabilistic polynomial time. Suppose that there exists an efficient algorithm D that on input g and $g^{u||0^i}$, outputs the *lsb* of u with probability $1/2+\epsilon$, where $u||0^i \in_R R_i$ and ϵ is non-negligible.

Then there exists an efficient algorithm D' that on input $y = g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs the *lsb* of v with probability at least $1/2 + \epsilon - (2/t)$.

Proof. Let D be an efficient algorithm as stated above. We construct an efficient algorithm D' that, given g and $g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs the *lsb* of v with probability at least $1/2 + \epsilon - (2/t)$. Let $\gamma = msb_{\log t}(v)$. That is, $v = \gamma||v'$ for some v'. We will find the *lsb* of v' by using D (because $lsb_1(v) = lsb_1(v')$).

(1) First, D' zeros the $\log t$ most significant bits of v by computing

$$y_1 = y \cdot g^{-\gamma \cdot 2^{n-\log t}} = g^{0^{\log t}||v'||0^{n-c}}.$$

(2) Next D' computes

$$y_2 = y_1^e, \text{ where } e = 1/2^{n-c-i} \bmod q.$$

Note that the exponent of y_1 is shifted to the right $n - c - i$ bits. Therefore, y_2 is written as $y_2 = g^s$ in such a way that

$$s = 0^{n-c-i+\log t}||v'||0^i.$$

(3) D' chooses $r \in R_i$ randomly and computes

$$y' = y_2 \cdot g^r = g^{s+r}.$$

(Note that $r = 2^i r'$ for some r' since $r \in R_i$.)
(4) D' invokes D with input (g, y').
(5) Suppose that D outputs a bit α. (If D outputs neither 0 nor 1, D' chooses a bit α randomly.) Then D' outputs $\beta = \alpha \oplus lsb_1(r')$.

Let $u = s + r$. Then u is uniformly distributed over $\{s' : s \leq s' \leq s + r_{max} \text{ and } 2^i|s'\}$, where r_{max} is the maximum element of R_i. Since $2^i \mid u$, we let $u' = u/2^i$. Then

$$u' = v' + r'.$$

If $u < q$ and $\alpha = lsb_1(u')$, then

$$\alpha = lsb_1(u') = lsb_1(v) \oplus lsb_1(r').$$

Hence

$$lsb_1(v) = \alpha \oplus lsb_1(r') = \beta (= \text{the output of } D').$$

Therefore,

$$\begin{aligned}\Pr(D' \text{ suceeds}) &\geq \Pr(u < q \text{ and } \alpha = lsb_1(u')) \\ &= \Pr(u < q \text{ and } D(g, g^u) = lsb_1(u'))\end{aligned}$$

For a fixed random tape C of D, let

$$GOOD(C) = \{x \mid x \in R_i,\ D(g, g^x) = lsb_1(x/2^i)\}$$

(It is clear that $2^i \mid x$ for $x \in R_i$.) Then

$$\begin{aligned}\Pr(D' \text{ suceeds}) &\geq \Pr(u < q \text{ and } D(g, g^u) = lsb_1(u')) \\ &= E_C[\Pr(u < q \text{ and } u \in GOOD(C))]\end{aligned}$$

where E_C denotes the expected value over C.

It is easy to see that "$u < q$ and $u \in GOOD(C)$)" is equivalent to $u \in GOOD(C)$. Therefore,

$$\Pr(D' \text{ suceeds}) \geq E_C[\Pr(u \in GOOD(C))]$$

Further, since u is uniformly distributed over $\{s' : s \leq s' \leq s + r_{max} \text{ and } 2^i | s'\}$, we obtain

$$\begin{aligned}E_C[\Pr(u \in GOOD(C))] &\geq E_C[\Pr_{y \in R_i}(y \in GOOD(C)) - \Pr_{y \in R_i}(y < s)] \\ &\geq E_C[\Pr_{y \in R_i}(y \in GOOD(C))] - E_C[\Pr_{y \in R_i}(y < s)] \\ &\geq \Pr_{y \in R_i}(y \in GOOD(C)) - \Pr_{y \in R_i}(y < s) \\ &\geq 1/2 + \epsilon - 2/t.\end{aligned}$$

Consequently,

$$\Pr(D' \text{ suceeds}) \geq 1/2 + \epsilon - 2/t.$$

□

Lemma 3. *In Lemma 2, let $t = 4/\epsilon$. Then there exists an efficient algorithm that on input $g, y = g^{v||0^{n-c}}$ and $msb_{\log t}(v)$, outputs v with overwhelming probability.*

Proof. In Lemma 2, D' outputs $lsb_1(v)$ with probability at least $1/2+\epsilon/2$ because $t = 4/\epsilon$. Here $\epsilon/2$ is non-negligible from the assumption of Lemma 2. Then by running D' polynomially many times (i.e., $2/\epsilon^3$ times) independently and taking the majority vote, we can obtain $b_1 = lsb_1(v)$ with overwhelming (i.e., $e^{-1/\epsilon}$) probability.

Next let

$$y_1 = (y(g^{2^{n-c}})^{-b_1})^{1/2} = g^{0||v'||0^{n-c}},$$

where $v = v'||b_1$. Applying the same process, we can find $lsb_1(v')$ similarly. By repeating this algorithm, we can find v with overwhelming probability. □

Now, we are ready to prove Theorem 1.

Proof. Suppose that A_0 and A_{n-c} are distinguishable. Then we will show that we can solve the DLSE problem. Assume that there exists a distinguisher D between A_0 and A_{n-c}, namely,

$$|\Pr[D(A_0) = 1] - \Pr[D(A_{n-c}) = 1]| > \frac{1}{p(n)}$$

for infinitely many n for some polynomial $p(\cdot)$. (A_0 and A_{n-c} in the above equation denote the uniform distribution over the set A_0 and A_{n-c}, respectively.)

Then, for some j such that $0 \leq j \leq n - c - 1$,

$$|\Pr[D(A_j) = 1] - \Pr[D(A_{j+1}) = 1]| > \frac{1}{np(n)}. \qquad (1)$$

We first show that we can find such an index j in polynomial time.

Let $p_i = \Pr[D(A_i) = 1]$ for $0 \leq i \leq n - c - 1$. We estimate each p_i by the sampling method of m experiments. Let $\hat{p}_i$ denote the estimated value. By using the Chernoff bound, we can show that

$$\Pr[|\hat{p}_i - p_i| > 1/8np(n)] \leq 2e^{-2m/64(np(n))^2}.$$

In other words, we can estimate all p_i with accuracy $\pm 1/8np(n)$ with high probability by using $m = 2048n^3(p(n))^2$ random samples. This means that we have, for the j of eq.(1),

$$|\hat{p}_{j+1} - \hat{p}_j| > 1/np(n) - 2/8np(n) = 3/4np(n).$$

Therefore, there exists at least one j which satisfies the above equation.

Our algorithm first finds an index i such that

$$|\hat{p}_{i+1} - \hat{p}_i| > 3/4np(n),$$

by using $\hat{p}_i$. For this i, we see that

$$|p_{i+1} - p_i| > 1/2np(n) \qquad (2)$$

by using the same argument as above.

We next show that D can be used as a prediction algorithm of Lemma 1. Wlog, we assume that $p_i - p_{i+1} > 1/2np(n)$ from eq.(2). Then we can show that

$$\frac{1}{2}\Pr(D(A_{i+1}) = 0) + \frac{1}{2}\Pr(D(A_i \setminus A_{i+1}) = 1) > \frac{1}{2} + \frac{1}{2np(n)}.$$

This means that

$$\Pr[D(g, g^{u||b||0^i}) = b) > \frac{1}{2} + \frac{1}{2np(n)}.$$

Thus D can be used as a prediction algorithm of Lemma 1 with $\epsilon = 1/2np(n)$.

We finally show that we can solve the DLSE problem by using Lemma 3. Suppose that we are given (g, y) such that $y = g^{v||0^{n-c}}$. In order to apply Lemma 3, we first let $t = 4/\epsilon = 8np(n)$. We next guess the value of $msb_{\log t}(v)$. For each guessed value γ, we apply Lemma 3 and obtain $\tilde{v}$. We then check if $y = g^{\tilde{v}||0^{n-c}}$. If so, we have found that $v = \tilde{v}$. Otherwise, we try another guessed value. The number of possible values of $msb_{\log t}(v)$ is

$$2^{\log t} = t = 8np(n).$$

Therefore, the exhaustive search on $msb_{\log t}(v)$ runs in polynomial time. Consequently, we can find v in polynomial time with overwhelming probability. $\square$