Zero-Knowledge Argument for Simultaneous Discrete Logarithms

Sherman S.M. Chow[1], Changshe Ma[2], and Jian Weng[3,4,5]

[1] Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, NY 10012, USA
schow@cs.nyu.edu
[2] School of Computer, South China Normal University, Guangzhou, 510631, China
changshema@gmail.com
[3] Department of Computer Science, Jinan University, Guangzhou 510632, China
[4] State Key Laboratory of Information Security
Institute of Software, Chinese Academy of Sciences, Beijing 100080, China
[5] State Key Laboratory of Networking and Switching Technology
Beijing University of Posts and Telecommunications, Beijing 100876, China
cryptjweng@gmail.com

Abstract. In Crypto'92, Chaum and Pedersen introduced a widely-used protocol (CP protocol for short) for proving the equality of two discrete logarithms (EQDL) with unconditional soundness, which plays a central role in DL-based cryptography. Somewhat surprisingly, the CP protocol has never been improved for nearly two decades since its advent. We note that the CP protocol is usually used as a non-interactive proof by using the Fiat-Shamir heuristic, which inevitably relies on the random oracle model (ROM) and assumes that the adversary is computationally bounded. In this paper, we present an EQDL protocol in the ROM which saves ≈40% of the computational cost and ≈33% of the prover's uploading bandwidth. Our idea can be naturally extended for simultaneously showing the equality of n discrete logarithms with $O(1)$-size commitment, in contrast to the n-element adaption of the CP protocol which requires $O(n)$-size. This improvement benefits a variety of interesting cryptosystems, ranging from signatures and anonymous credential systems, to verifiable secret sharing and threshold cryptosystems. As an example, we present a signature scheme that only takes one (offline) exponentiation to sign, without utilizing pairing, relying on the standard decisional Diffie-Hellman assumption.

Keywords: Equality of discrete logarithms, simultaneous discrete logarithms, honest-verifier zero-knowledge argument, online/offline signature, Diffie-Hellman problem.

1 Introduction

The protocol for proving the equality of two discrete logarithms plays a central role in cryptography. Such a protocol, executed by a prover $\mathcal{P}$ and a verifier $\mathcal{V}$,

M.T. Thai and S. Sahni (Eds.): COCOON 2010, LNCS 6196, pp. 520–529, 2010.

can be roughly explained as follows: Suppose $\mathbb{G}$ is a cyclic group with prime order q, and g and h are two random elements chosen from $\mathbb{G}$. $\mathcal{P}$ knows $x \in \mathbb{Z}_q$ such that $y_1 = g^x$ and $y_2 = h^x$. After executing the protocol, $\mathcal{P}$ must convince $\mathcal{V}$ that y_1 and y_2 indeed have the same exponent with respect to the base g and h respectively, i.e., $\log_g y_1 = \log_h y_2$. The idea can be naturally extended to the *simultaneous discrete logarithm* problem, where one wants to show that $y_0 = g_0^x, \cdots, y_{n-1} = g_{n-1}^x$ when given n generators $g_0, g_1, \cdots, g_{n-1}$ of group $\mathbb{G}$.

Chaum and Pedersen [1] introduced a famous protocol (hereinafter referred as CP protocol) for proving the equality of two discrete logarithms, which has been widely used to design a variety of cryptosystems. Examples include but not limited to digital signatures [2, 3], verifiable secret sharing [4], threshold cryptosystems [5, 6], fair exchange protocols [7], pseudonym [8], electronic payment [9], electronic voting[10, 11], etc. Any improvement on the CP protocol is influential since it benefits all these cryptosystems accordingly.

Somewhat surprisingly, the CP protocol has never been improved for nearly two decades since its advent. A natural question to ask is whether we can improve the CP protocol. In this paper, we answer the question motivating the above question instead – can we have a more efficient protocol which "benefits" the cryptosystems utilizing the CP protocol?

To answer our question, we investigate how the CP protocol is typically used and what kinds of its properties have been utilized in cryptographic or security applications. In many cases, interactive proof is undesirable, if not impossible. Examples include digital signatures which provide non-repudiation property and multi-signatures [12, 13] which require all signers to be online if it is done in an interactive setting. So the CP protocol is turned into a non-interactive proof by the Fiat-Shamir heuristic [14], where the prover "challenges" herself by treating the output of the hash function as the challenge given by the verifier. This inevitably relies on the random oracle model [15] (ROM). On the other hand, the CP protocol can be proven secure without relying on the ROM.

As a zero-knowledge proof system, the CP protocol provides unconditional soundness. Roughly speaking, this (soundness) means the prover cannot cheat without being caught in most cases, without any condition imposed on the prover, in particular, the prover may possess unlimited computational power. For real-world applications, we may feel comfortable to assume that the computational abilities of an adversary are bounded, and hence a zero-knowledge *argument*, a system with computational soundness, is sufficient.

1.1 Our Contributions

Based on the above two simple yet important observations, we propose a protocol which gives a zero-knowledge argument system for simultaneous discrete logarithm. In the CP protocol, the prover $\mathcal{P}$ firstly generates two group elements as the commitments. We *carefully* integrate these two elements together in our protocol, which gives a higher performance in terms of both computational cost and communication overhead. More concretely, our protocol offers about 40% computational cost saving, and saves one group element computation overhead.

We remark that it is not entirely trivial to combine two commitments into a single element. We show that a class of trial attempts is insecure.

Our protocol can be naturally extended for simultaneous discrete logarithm for n elements where n is an integer greater than 2. Compared with the n-element adaption of the CP protocol which uses $O(n)$ commitments, we achieve $O(1)$ size.

Finally, we use our protocol to construct an efficient signature scheme which only takes one offline exponentiation and no online exponentiation to sign, under the standard decisional Diffie-Hellman assumption, without utilizing pairing.

2 Preliminaries

2.1 Notations and Complexity Problems

We explain some notations used in the rest of this paper. For a prime q, $\mathbb{Z}_q$ denotes the set $\{0, 1, 2, \cdots, q-1\}$, and $\mathbb{Z}_q^*$ denotes $\mathbb{Z}_q \backslash \{0\}$. For a finite set S, $x \in_R S$ means choosing an element x uniformly at random from S. We use $\mathbb{G}$ to denote a cyclic group with prime order q and $\ell_q = |\mathbb{G}|$ to denote the bit-length of the binary representation of an element in $\mathbb{G}$. We use PPT to mean probabilistic polynomial time.

Definition 1. *The discrete logarithm (DL) problem is, given $g, y \in \mathbb{G}$, to compute $x \in \mathbb{Z}_q$ such that $y = g^x$. We say that an algorithm $\mathcal{B}$ can solve the DL problem in group $\mathbb{G}$ with advantage ϵ if $\Pr[\mathcal{B}(g, y) = x | g \in_R \mathbb{G}, x \in_R \mathbb{Z}_q, y = g^x] \geqslant \epsilon$.*

Definition 2. *The computational Diffie-Hellman (CDH) problem is that, given $g, g^a, g^b \in \mathbb{G}$ with unknown $a, b \in \mathbb{Z}_q$, to compute g^{ab}. We say that an algorithm $\mathcal{B}$ has advantage ϵ in solving CDH problem in group $\mathbb{G}$, if $\Pr[\mathcal{B}(g, g^a, g^b) = g^{ab} | g \in_R \mathbb{G}, a, b \in_R \mathbb{Z}_q] \geqslant \epsilon$.*

Definition 3. *The decisional Diffie-Hellman (DDH) problem is, given g, g^a, g^b, g^c, where $a, b \in_R \mathbb{Z}_q^*$, to tell if $c = ab$ or $c \in_R \mathbb{Z}_q^*$. A tuple (g, g^a, g^b, g^c) is called a "DDH tuple" if $ab = c \mod q$ holds. We say that an algorithm $\mathcal{B}$ has advantage ϵ in solving the DDH problem in $\mathbb{G}$ if $|\Pr[\mathcal{B}(g, g^a, g^b, g^{ab}) = 1 | g \in_R \mathbb{G}, a, b \in_R \mathbb{Z}_q, c = ab] - \Pr[\mathcal{B}(g, g^a, g^b, g^c) = 1 | g \in_R \mathbb{G}, a, b, c \in_R \mathbb{Z}_q]| \geqslant \epsilon$.*

We say that the (t, ϵ)-DL/CDH/DDH assumption holds if no PPT algorithm can solve the respective problem with an advantage at least ϵ within time t.

2.2 Zero-Knowledge Proof for Equality of Discrete Logarithms

The security notions for honest-verifier zero-knowledge protocols for proving the equality of two logarithms are reviewed as follows.

Definition 4. *A protocol for proving the equality of two discrete logarithms is said to be secure, if it simultaneously satisfies the following three properties:*

- ***Completeness.*** *For any honest prover $\mathcal{P}$, an honest verifier $\mathcal{V}$ accepts with overwhelming probability.*
- ***Soundness.*** *An algorithm $\mathcal{A}$ (t, ϵ)-breaks the soundness of the protocol if $\mathcal{A}$ can cheat, without conducting any active attacks, an honest verifier $\mathcal{V}$ (i.e., $\mathcal{A}$ can make $\mathcal{V}$ to accept its proof whereas $\log_g y_1 \neq \log_h y_2$) with at least success probability ϵ within time t, where the probability is taken over the random coins consumed by $\mathcal{A}$ and $\mathcal{V}$.*
- ***(Honest-Verifier) Zero-Knowledge.*** *We say that the protocol is statistical (or computational) zero-knowledge, if for every PPT (honest) verifier $\mathcal{V}$, there exists a PPT machine $\mathcal{S}$ (also called simulator), which is not allowed to interact with the prover $\mathcal{P}$, such that the following two distributions are statistically (or computationally) indistinguishable:*
  - *the output of $\mathcal{V}$ after interacting with a prover $\mathcal{P}$ on the common input of public parameters $(\mathbb{G}, g, h, y_1, y_2)$, and*
  - *the output of $\mathcal{S}$ with only the input of public parameters $(\mathbb{G}, g, h, y_1, y_2)$.*

Remark: In our protocol, the public parameters should also contain the description of the hash function H. Our proof for soundness requires modeling H as a random oracle.

2.3 Digital Signature and Its Existential Unforgeability

A signature scheme is defined by the following triple of algorithms:

- $\mathsf{KeyGen}(1^\ell)$: it takes a security parameter 1^ℓ and outputs a private/public key pair (sk, pk).
- $\mathsf{Sign}(sk, m)$: it outputs a signature σ taking a signing key sk and a message m as inputs.
- $\mathsf{Verify}(pk, \sigma, m)$: this verification algorithm takes as input a public key pk, a signature σ and a message m. It outputs 1 if and only if σ is a valid signature on m under pk, 0 otherwise.

Security is modeled by the existential unforgeability under adaptive chosen message attack (EUF-CMA), defined via the following game between a forger $\mathcal{F}$ and a challenger $\mathcal{C}$:

Setup. The challenger $\mathcal{C}$ runs algorithm $\mathsf{KeyGen}(1^\ell)$ to generate a key pair (pk, sk). It then forwards pk to $\mathcal{F}$, keeping sk to himself.

Signing queries. For each query, $\mathcal{F}$ adaptively issues a message m, the challenger runs $\mathsf{Sign}(sk, m)$ to get a signature σ, which is returned to $\mathcal{F}$.

Forge. $\mathcal{F}$ outputs a message/signature pair (m^*, σ^*) where $\mathsf{Verify}(pk, \sigma^*, m^*) = 1$.

$\mathcal{F}$ wins if m^* did not appear in the signing queries. $\mathcal{F}$'s advantage is defined by $\mathrm{Adv} = \Pr[\mathcal{F} \text{ wins}]$, where the probability is taken over the random coins consumed by $\mathcal{C}$ and $\mathcal{F}$.

Definition 5. *We say that a signature scheme is (t, q_s, ϵ)-EUF-CMA secure, if for any t-time forger $\mathcal{F}$ who asking at most q_s signing queries, his advantage is less than ϵ.*

3 Zero-Knowledge Argument for Simultaneous Discrete Logarithms

3.1 Review of Chaum-Pedersen Protocol

Let $\mathbb{G}$ be a cyclic group of prime order q, and let g, h be two random generators of $\mathbb{G}$. The prover $\mathcal{P}$ picks $x \in \mathbb{Z}_q$, and publishes $y_1 = g^x$ and $y_2 = h^x$. To convince the verifier $\mathcal{V}$ that $\log_g y_1 = \log_h y_2$, the CP protocol [1] proceeds as in Fig. 1. It is well known that the CP protocol is an honest-verifier zero-knowledge protocol [16].

1. $\mathcal{P}$ picks $k \in_R \mathbb{Z}_q$ and sends the pair of commitments $(a_1, a_2) = (g^k, h^k)$ to $\mathcal{V}$.
2. Upon receiving (a_1, a_2), $\mathcal{V}$ sends the challenge $c \in \mathbb{Z}_q^*$ to $\mathcal{P}$.
3. $\mathcal{P}$ sends the response $s = k - cx \mod q$ to $\mathcal{V}$.
4. $\mathcal{V}$ accepts if and only if both $a_1 = g^s y_1^c$ and $a_2 = h^s y_2^c$ hold.

Fig. 1. Chaum-Pedersen Protocol for proving $y_1 = g^x$ and $y_2 = h^x$

3.2 Our Proposed Protocol

Let $H : \mathbb{G}^2 \to \mathbb{Z}_q^*$ be a cryptographic hash function (in the proof of soundness, we shall model it as a random oracle). We define the commitment as $v = (g^z h)^k$ where $z = H(y_1, y_2)$. We assume that the discrete logarithm $\log_g h$ is unknown to all participants. This can be easily ensured in practice by requiring h to be the output of a hash function (which again is modeled as a random oracle) with a public seed and a public input. For example, one may set h to be $PRF_s(g)$ where PRF is a pseudorandom function taking the public seed s. Detailed protocol is shown in Fig. 2.

Use of $H()$. It is not entirely trivial to integrate two elements into a single element. The use of $H()$ is crucial for the security. In particular, if $H()$ is not used (as if it always outputs 1), or if y_2 is not required as part of the input of $H()$, e.g. the commitment is computed as $(g^{z'} h)^k$ where $z' = H(g, h, y_1)$ instead, a malicious prover $\mathcal{P}$ can cheat a verifier $\mathcal{V}$ to accept.

Concretely, the malicious prover $\mathcal{P}$ first picks $x, x' \in_R \mathbb{Z}_q^*$, publishes $y_1 = g^x$ and $y_2 = y_1^{-z'} (g^{z'} h)^{x'}$ where $z' = H(g, h, y_1)$, and then interacts with $\mathcal{V}$ as follows: $\mathcal{P}$ sends the commitment $v = (g^{z'} h)^k$ where $k \in_R \mathbb{Z}_q^*$ to $\mathcal{V}$; then responds with $s = k - x'c \mod q$ upon received a challenge $c \in_R \mathbb{Z}_q^*$. Since $(g^{z'} h)^s (y_1^{z'} y_2)^c = (g^{z'} h)^s (y_1^{z'} \cdot y_1^{-z'} (g^{z'} h)^{x'})^c = (g^{z'} h)^{s + x'c} = (g^{z'} h)^k = v$, but $\log_g y_1 \neq \log_h y_2$ with overwhelming probability. So, it fails to provide soundness.

Finally, the range of $H()$ should be large (exponential in the security parameter) even if y_2 is part of the input. Otherwise, a cheating prover can just try all possible z' in the small range of $H()$, construct $y_2 = y_1^{-z'} (g^{z'} h)^{x'}$ and see if $z' = H(y_1, y_2)$ by coincidence. When the range of $H()$ is small, the last equality happens with non-negligible probability, and this exhaustive search is efficient.

1. $\mathcal{P}$ picks $k \in_R \mathbb{Z}_q^*$ and sends the commitment $v = (g^z h)^k$ to $\mathcal{V}$, where $z = H(y_1, y_2)$.
2. Upon receiving v, $\mathcal{V}$ sends the challenge $c \in_R \mathbb{Z}_q^*$ to $\mathcal{P}$.
3. $\mathcal{P}$ sends the response $s = k - cx \mod q$ to $\mathcal{V}$.
4. $\mathcal{V}$ accepts if and only if $v = (g^z h)^s (y_1^z y_2)^c$ holds, where $z = H(y_1, y_2)$.

Fig. 2. Our Protocol for $ZPK[(x) : y_1 = g^x \wedge y_2 = h^x]$

The security of our protocol can be ensured by the following theorem, its proof can be found in the full version.

Theorem 1. *In the random oracle model, our protocol is secure according to Definition 4, under the DL assumption in group* $\mathbb{G}$.

Extension to n-element. Our protocol presented in Fig. 2 can be naturally extended to show the equality of n discrete logarithms, i.e., $\log_{g_0} y_0 = \log_{g_1} y_1 = \cdots = \log_{g_{n-1}} y_{n-1} = x$. Extended protocol is given in Fig. 3. In the full version, we will discuss how to modify the proof of Theorem 1 to fit for our extended protocol. Our protocol can be seen as borrowing some techniques from the multi-signature scheme in [12].

Efficiency. We exploit the fact that an n-element multi-exponentiation (a computation of the form $a_1^{x_1} \cdots a_n^{x_n}$) can be performed significantly more efficiently than n different simple exponentiations using the windows methods in [17–20]. A multi-exponentiation requires a number of multiplications and squarings. For simplicity, we assume multiplication and squaring have the same complexity, and we use t_{m} to denote the computational cost of one modular multiplication in group $\mathbb{G}$. Let $\ell = |\mathbb{Z}_q|$. Setting $\ell = 160$ and the window size w to be 1, the average costs of one exponentiation, 2-element multi-exponentiation and 4-element multi-exponentiation in $\mathbb{G}$ are about $t_{1\mathrm{xp}} = 1.488\ell t_{\mathrm{m}}$, $t_{2\mathrm{xp}} = 1.739\ell t_{\mathrm{m}}$, and $t_{4\mathrm{xp}} = 1.863\ell t_{\mathrm{m}}$ respectively. For n-element multi-exponentiation, its cost increases with n and converges to $t_{n\mathrm{xp}} = 1.988\ell t_{\mathrm{m}}$ when n reaches 15. We obtain these figures from [20][Theorem 3.4].

1. $\mathcal{P}$ picks $k \in_R \mathbb{Z}_q^*$ and sends the commitment $v = (g_0 g_1^{z_1} \cdots g_{n-1}^{z_{n-1}})^k$ to $\mathcal{V}$, where $z_i = H(i, y_1, \cdots, y_{n-1})$ for $i \in [1, n-1]$.
2. Upon receiving v, $\mathcal{V}$ sends the challenge $c \in_R \mathbb{Z}_q^*$ to $\mathcal{P}$.
3. $\mathcal{P}$ sends the response $s = k - cx \mod q$ to $\mathcal{V}$.
4. $\mathcal{V}$ accepts if and only if $v = (g_0 g_1^{z_1} \cdots g_{n-1}^{z_{n-1}})^s (y_0 y_1^{z_1} \cdots y_{n-1}^{z_{n-1}})^c$ holds, where $z_i = H(i, y_1, \cdots, y_{n-1})$ for $i \in [1, n-1]$.

Fig. 3. Our Protocol for proving $y_0 = g_0^x, \cdots, y_{n-1} = g_{n-1}^x$

Table 1. Comparisons between Our Protocol and Chaum-Pedersen Protocol

Protocol	**Chaum-Pedersen [1]**		**Our Protocol**	
Number of DLs	2	n	2	n
Prover's Operation	$2t_{1\text{xp}} = 2.976\ell t_{\text{m}}$	$n \cdot 1.488\ell t_{\text{m}}$	$t_{2\text{xp}} = 1.739\ell t_{\text{m}}$	$t_{n\text{xp}} = 1.988\ell t_{\text{m}}$
Verifier's Operation	$2t_{2\text{xp}} = 3.478\ell t_{\text{m}}$	$n \cdot 1.739\ell t_{\text{m}}$	$t_{4\text{xp}} = 1.863\ell t_{\text{m}}$	$t_{n\text{xp}} = 1.988\ell t_{\text{m}}$
Bandwidth	$2\lvert\mathbb{G}\rvert + 2\ell$	$n\lvert\mathbb{G}\rvert + 2\ell$	$1\lvert\mathbb{G}\rvert + 2\ell$	
Model	Standard		Random Oracle	
Soundness	Unconditional		Computational	

An efficiency comparison between our protocol and the CP protocol is given in Table 1. For the equality of 2 discrete logarithms, our protocol offers about 40% computational cost saving, and saves one group element in terms of the communication overhead. The savings are more for $n > 2$, which basically reduces the computation and the bandwidth requirements from $O(n)$ to $O(1)$.

4 An Efficient Signature Scheme

The well-known CP protocol has been widely used as a building block in many cryptosystems. For examples, applying the Fiat-Shamir transformation technique [14] on the CP protocol, several signature schemes have been constructed [2, 3]. We use our protocol to construct an efficient signature scheme based on the observation made in [2]. The public key contains (g, h, y_1, y_2) and a signature is simply a proof such that the public key is a DDH tuple.

Let $\mathbb{G}$ be a group with ℓ_q-bit prime order q where ℓ_q is the security parameter. Let g be a random generator of $\mathbb{G}$, and let $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$ be two cryptographic hash functions (which are modeled by random oracles in the security proof). Our DDH-based signature scheme $\mathsf{S2}$ is specified as below:

KeyGen. Pick $x \in_R \mathbb{Z}_q^*, h \in_R \mathbb{G}$, output x as the private key and $(h, y_1 = g^x, y_2 = h^x)$ as the public key.

Sign. Given the private key $x \in \mathbb{Z}_q^*$, the corresponding public key (y_1, y_2), and a message $m \in \{0,1\}^*$, the signer generates a signature as below:

1. compute $z = H_1(y_1, y_2)$; (pre-computable from the public key)
2. compute $u = g^z h$; (pre-computable from the public key too)
3. pick $k \in_R \mathbb{Z}_q^*$, and compute $v = u^k = (g^z h)^k$; (independent of m)
4. output $(e = H_2(m, y_1, y_2, v), s = k - xe \mod q)$.

Sign is derived from our protocol using the Fiat-Shamir heuristic: $v = (g^z h)^k$ is the commitment, e is the challenge, and $s = k - xe$ is the response.

Verify. Given the public key (y_1, y_2), and a signature (e, s) on message m, compute $z = H_1(y_1, y_2)$, and $v' = (g^z h)^s (y_1^z y_2)^e$; return 1 if $e = H_2(m, y_1, y_2, v')$, 0 otherwise.

For a valid signature (e, s) on m, $v' = (g^z h)^s (y_1^z y_2)^e = (g^z)^{s+xe} h^{s+xe} = (g^z h)^k = v$.

Verifier Pre-computation. For a given signer, observe that the values $z, u = g^z h$ and $w = y_1^z y_2$ are invariants for every signature verification, and hence they can be pre-computed. This saves Verify from one 4-element multi-exponentiation to one 2-element multi-exponentiation, and is helpful in the situation where the verifier receives signatures from the same signer more than once.

Signer Pre-computation. The values $z = H_1(y_1, y_2)$ and $u = g^z h$ can be pre-computed without the private key x. Sign only takes one online exponentiation.

Offline/Online Signing. A signing algorithm can be split into offline and online stages. Offline computation refers to computation that may be performed before the message to be signed is known, which possibly includes pre-selecting random numbers for a probabilistic signature scheme. In contrast, online computation must be done after the message is known.

Our Sign algorithm can be naturally split. The offline stage generates the offline token (k, v). The online stage outputs the signature (e, s) by one hash function evaluation and one modular multiplication, which is nearly the least one may expect for a hash-based signature.

Security. The security of our scheme S2 can be ensured by the following theorem. Its detailed proof can be found in the full version.

Theorem 2. *Our signature scheme S2 is EUF-CMA secure in the random oracle model, under the DDH assumption in group $\mathbb{G}$. Concretely, if there exists a t-time adversary $\mathcal{F}$ that has a non-negligible advantage ϵ against the EUF-CMA security of our scheme S2, making at most q_S signing queries, q_{H_1} queries on hash function H_1, and at most q_{H_2} queries on hash function H_2, then there exists an algorithm $\mathcal{B}$ that can solve the DDH problem with*

$$\text{advantage } \epsilon' \geqslant \epsilon - \eta - \frac{\eta + q_{H_2} + q_s}{2^{\ell_q}} \text{ within time } t' \leq t + (q_s + 1)t_{\text{2xp}},$$

where η denotes the success probability to break the soundness of our protocol in Fig. 2, and t_{2xp} denotes the computational cost of a 2-element multi-exponentiation in $\mathbb{G}$.

Efficiency Comparisons. In [2], Katz and Wang presented a signature scheme (denoted by KW-DDH) with tight security reduction to the DDH assumption. Table 2 presents a detailed comparison between KW-DDH and S2. Both the computational cost for signing and verification of our DDH-based scheme are reduced to 50%, where the comparison is made by instantiating both DDH-based schemes using the same security parameter. Even without signer-specific pre-computation by the verifier, verification only takes $1.863\ell t_{\text{m}}$ which remains competitive. We remark that the aim of [2] is to achieve a tight security reduction. To compensate for the tightness, we need to use a larger security parameter, which results in a larger signature size. However, the 50% saving in computational cost means that our scheme can afford a larger security parameter without incurring too much extra computation time.

Table 2. Efficiency Comparisons between Different Signature Schemes

Scheme	Sign	Verify	Signature Size	Public Key Size
KW-DDH [2]	$2t_{1\text{xp}} = 2.976\ell t_{\text{m}}$	$2t_{2\text{xp}} = 3.478\ell t_{\text{m}}$	$2\|\mathbb{Z}_q\|$	$3\|\mathbb{G}\|$
Our Scheme S2	$t_{1\text{xp}} = 1.488\ell t_{\text{m}}$	$t_{2\text{xp}} = 1.739\ell t_{\text{m}}$	$2\|\mathbb{Z}_q\|$	$3\|\mathbb{G}\|$

5 Conclusion

We investigated the protocol for the simultaneous discrete logarithms problem – a fundamental building block appears in many cryptosystems of different flavors. We noted that such a protocol is usually used in a non-interactive form by applying the Fiat-Shamir heuristics which inevitably relies on the random oracle model. The adversary is also assumed to be computationally bounded in these applications. We thus proposed an efficient zero-knowledge argument system in the random oracle model which gives a better performance in terms of both the computational cost and the bandwidth. We believe that our protocol can also be used to improve other cryptographic cryptosystems. In particular, we proposed a signature scheme that only takes one (offline) exponentiation to sign, without pairing, based on the standard decisional Diffie-Hellman assumption.

We remark that our technique can be generalized for proving linear relations other than the simple equality. It may be interesting to see if other zero-knowledge argument systems can be speeded up using a similar technique.

References

1. Chaum, D., Pedersen, T.P.: Wallet Databases with Observers. In: Brickell, E.F. (ed.) CRYPTO 1992. LNCS, vol. 740, pp. 89–105. Springer, Heidelberg (1993)
2. Katz, J., Wang, N.: Efficiency Improvements for Signature Schemes with Tight Security Reductions. In: Jajodia, S., Atluri, V., Jaeger, T. (eds.) Computer and Communications Security, pp. 155–164. ACM, New York (2003)
3. Chevallier-Mames, B.: An Efficient CDH-based Signature Scheme with a Tight Security Reduction. In: Shoup, V. (ed.) CRYPTO 2005. Chevallier-Mames, B, vol. 3621, pp. 511–526. Springer, Heidelberg (2005)
4. Stadler, M.: Publicly Verifiable Secret Sharing. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 190–199. Springer, Heidelberg (1996)
5. Shoup, V.: Practical Threshold Signatures. In: Preneel, B. (ed.) EUROCRYPT 2000. LNCS, vol. 1807, pp. 207–220. Springer, Heidelberg (2000)
6. Shoup, V., Gennaro, R.: Securing Threshold Cryptosystems against Chosen Ciphertext Attack. J. Cryptology 15(2), 75–96 (2002)
7. Ateniese, G.: Verifiable Encryption of Digital Signatures and Applications. ACM Transactions on Information and System Security (TISSEC) 7(1), 1–20 (2004)
8. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym Systems. In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 184–199. Springer, Heidelberg (2000)
9. Camenisch, J., Maurer, U.M., Stadler, M.: Digital Payment Systems with Passive Anonymity-Revoking Trustees. Journal of Computer Security 5(1), 69–90 (1997)

10. Cramer, R., Gennaro, R., Schoenmakers, B.: A Secure and Optimally Efficient Multi-Authority Election Scheme. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 103–118. Springer, Heidelberg (1997)
11. Chow, S.S.M., Liu, J.K., Wong, D.S.: Robust Receipt-Free Election System with Ballot Secrecy and Verifiability. In: NDSS, The Internet Society (2008)
12. Bellare, M., Neven, G.: Multi-Signatures in the Plain Public-Key Model and A General Forking Lemma. In: Juels, A., Wright, R.N., di Vimercati, S.D.C. (eds.) Computer and Communications Security, pp. 390–399. ACM, New York (2006)
13. Ma, C., Weng, J., Li, Y., Deng, R.H.: Efficient Discrete Logarithm based Multi-Signature Scheme in the Plain Public Key Model. Des. Codes Cryptography 54(2), 121–133 (2010)
14. Fiat, A., Shamir, A.: How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987)
15. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. Computer and Communications Security, 62–73 (1993)
16. Goldreich, O.: Foundations of Cryptography. Basic Tools, vol. 1. Cambridge University Press, Cambridge (2001)
17. Dimitrov, V.S., Jullien, G.A., Miller, W.C.: Complexity and Fast Algorithms for Multiexponentiations. IEEE Trans. Computers 49(2), 141–147 (2000)
18. Möller, B.: Algorithms for Multi-exponentiation. In: Vaudenay, S., Youssef, A.M. (eds.) SAC 2001. LNCS, vol. 2259, pp. 165–180. Springer, Heidelberg (2001)
19. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (2001)
20. Avanzi, R.M.: On Multi-Exponentiation in Cryptography. Cryptology ePrint Archive, Report 2002/154 (2002)