Efficient Primitives from Exponentiation in $\mathbb{Z}_p$

Shaoquan Jiang

Department of Computer Science,
University of Electronic Science and Technology of China,
ChengDu, CHINA 610054
Email: jiangshq@calliope.uwaterloo.ca

February 9, 2006

Abstract. Since Diffie-Hellman [14], many secure systems, based on discrete logarithm or Diffie-Hellman assumption in $\mathbb{Z}_p$, were introduced in the literature. In this work, we investigate the possibility to construct efficient primitives from exponentiation techniques over $\mathbb{Z}_p$. Consequently, we propose a new pseudorandom generator, where its security is proven under the decisional Diffie-Hellman assumption. Our generator is the most efficient among all generators from $\mathbb{Z}_p^*$ that are provably secure under standard assumptions. If an appropriate precomputation is allowed, our generator can produce $O(\log \log p)$ bits per modular multiplication. This is the best possible result in the literature (even improved by such a precomputation as well). Interestingly, our generator is the first provably secure under a decisional assumption and might be instructive for discovering potentially more efficient generators in the future. Our second result is a new family of universally collision resistant hash family (CRHF). Our CRHF is provably secure under the discrete log assumption and is more efficient than all previous CRHFs that are provably secure under standard assumptions (especially without a random oracle). This result is important, especially when the unproven hash functions (e.g., MD4, MD5, SHA-1) were broken by Wang et al. [41–43].

1 Introduction

Diffie-Hellman protocol [14] is an exponentiation based key exchange procedure. It is provably secure (against a passive attack) [8] under a now called *decisional Diffie-Hellman* (DDH) assumption. Since then, Diffie-Hellman techniques in $\mathbb{Z}_p^*$ have been largely employed to construct secure systems, see a very partial list of random examples: key exchange [28, 8–10, 26], encryption [16, 12], key escow [3]. As provable security is always related to a mathematically hard problem, the systems above are usually proven secure under an assumption of either discrete log, computational Diffie-Hellman, or the decisional Diffie-Hellman. These assumptions, throughout more than twenty years' tests [1, 7, 32, 38, 39], have become widely accepted standard assumptions. Naturally, exponentiation in $\mathbb{Z}_p^*$ has served as the main technique in embedding a hard problem into such a system. In this work, we investigate the possibility to construct efficient and secure primitives from exponentiations in $\mathbb{Z}_p^*$. Consequently, we obtain a new pseudorandom generator and an efficient family of collision resistant hash function. Before going on, we first review the research status in these two topics.

Pseudorandom Generator A key stream generator is a polynomial time algorithm, which upon a short secret outputs a poly-length binary stream. Encryption of a message is to bit-wise XOR it with the underlying key stream. Decryption works in the obvious way. Key stream generators are widely used in the real world, from ancient military communications to today's cell phone conversations. The notion of pseudorandom generator (or cryptographically secure key stream generator) was formally defined by Blum and Micali [6] and Yao [44]. In these (different but equivalent) definitions, a generator is said to be secure if no PPT algorithm can distinguish the generator's output from a uniformly random stream. Blum and Micali [6] constructively showed that a one-way permutation suffices to construct a pseudorandom generator. Then, they showed that a generator that iterates $g^x \pmod{p}$ for a large prime p and that extracts the most significant bit of x, is secure. This is improved by Long and Wigderson [31] for extracting the most $O(\log\log p)$ significant bits in each iteration. Blum, Blum and Shub [5] showed that the parity sequence with an iteration function $x^2 \pmod{N}$ is secure, where N is a RSA composite [36]. Yao [44] and Goldwasser et al [18] constructed more pseudorandom generators from the intractability of factoring. Alex et al. [2] showed that inverting a RSA function is equivalent to guessing the least significant bit of the input significantly better than 1/2. They further showed that the least significant $O(\log\log N)$ bits of RSA function $x^e \pmod{N}$ are simultaneously secure. This results was also obtained by Vazirani and Vazirani [40]. This implies a pseudorandom generator with each iteration extracting $O(\log\log N)$ bits. Hastad et al. [23] showed that generally a secure pseudorandom generator exists if and only if a one-way function exists. Hastad et al. [24] showed that the most (or least) significant $\lceil \frac{n}{2} \rceil$ bits of exponentiation function modulus a RSA composite N are simultaneously secure, where $|N| = n$. This results in a HSS generator that extracts half of the RSA input in each iteration. Goldreich and Rosen [22] improved the HSS generator with more efficient computation in each iteration. These generators [24, 22] are further improved by Dedic et al. [15] by removing the requirement of an extra extractor or hash. Recently, Patel et al [33] and Gennaro [17] constructed a very efficient generator that output almost n bits for each iteration, while his construction assumes a non-standard Discrete Logarithm Short Exponent Assumption.

Collision Resistant Hash Function Roughly speaking, collision resistant hash function (CRHF) is a function that is hard to find two inputs with an identical function value. CRHF was first proposed by Damgard [13] from claw-free permutations. Their construction requires $O(1/\log r)$ time (r is a fixed integer) to process one bit while none of the concrete schemes in his paper can achieve $O(1/\log k)$ time per bit, where k is the security parameter. CRHF in Pointcheval [35] and Shamir and Tauman [37] require 1.5 modular multiplication per bit. Goldwasser et al. [19] requires one modular squaring per bit. Bellare et al. [4] proposed a very efficient CRHF but it assumes random oracle. Efficient CRHF from non-standard assumptions are proposed in Peikert and Rosen [34] and Contini et al. [11]. To our knowledge, no construction, provably secure under a standard assumption, has achieved $O(1/\log k)$ time per bit.

1.1 Contribution

In this work, through manipulating an exponentiation technique in $\mathbb{Z}_p^*$ (p a prime), we construct a new pseudorandom generator and a new family of collision resistant hash function.

Our generator can output one bit per one modular multiplication and more efficient than the previous generators from $\mathbb{Z}_p^*$ (i.e., [6, 31]) that are probably secure under a standard assumption. Our generator is the first one provably secure under a decisional assumption and might be instructive for discovering potentially more efficient generators. But we point out that, comparing with factoring based construction, our generator is asymptotically the same efficient as HSS generator [24] and BBS generator [5] but less efficient than GR generator [22], ACGS generator [2] generator and DRV generator [15]. We stress that here the comparison assumes $|p| = |N|$, where N is the RSA modulus used in factoring based construction. This is justified by the following: (1) in the current start of art results, factoring and discrete log problems have the same heuristic cryptanalytic result [38, 29]; (1) no known cryptanalytic result can separate the decisional Diffie-Hellman and the discrete log problem in $\mathbb{Z}_p^*$, when p is a safe prime. If an appropriate precomputation is allowed, our generator can output $O(\log\log p)$ bits per modular multiplication, which achieves the current best result as GR generaor, DRV generator and ACGS generator (although DR generator and DRV generator requires less precomputation than ours and ACGS generator needs no precomputation at all).

Our CRHF is provably secure under the discrete log assumption. It requires only $O(\frac{1}{\log \log p})$ time per bit and is more efficient than all previous constructions that are provably secure under standard assumptions (especially without a random oracle). This result is very important, especially when unproven hash constructions (e.g., MD4, MD5, SHA-1) were broken recently [41–43].

2 Notions

In this section, we introduce some notions that will be used in this paper. Denote $\mathbb{Z}$ the set of integers, $\mathbb{R}$ the set of real numbers. We say a function $\nu : \mathbb{Z} \to \mathbb{R}$ is *negligible*, if for any positive polynomial $p(n)$, $\lim_{n\to\infty} \nu(n)p(n) = 0$.

Definition 1. *We say two ensembles $\{X_n\}_n$ and $\{Y_n\}_n$ are* **computationally indistinguishable**, *if for any probabilistic polynomial time algorithm $\mathcal{A}$ and any polynomial $p(n)$, when n large enough,*

$$|\Pr[\mathcal{A}(X_n) = 1] - \Pr[\mathcal{A}(Y_n) = 1]| < 1/p(n).$$

Definition 2. *Let U_l be a random variable uniformly distributed over $\{0,1\}^l$. We say an efficiently computable function $G : \{0,1\}^\kappa \times \mathbb{Z} \to \{0,1\}^*$ is a* **pseudorandom generator**, *if for any polynomially bounded integer $l \in \mathbb{Z}$, $G(U_\kappa, l)$ is computationally indistinguishable from U_l.*

In the above, we only consider the case of a binary generator. We can also generalize it to the setting where the generator output is from an arbitrary domain D (instead of $\{0,1\}$ only). In this case, the above definition is modified such that U_t is uniformly random in D^t and G is a function from $D' \times \mathbb{Z}$ to D^* for some domain D'. We call a generator satisfying the modified definition **a pseudorandom number sequence generator**.

A pseudorandom function is a cryptographic approximation of a random function. Loosely speaking, pseudorandom functions are functions that are indistinguishable from random functions.

Definition 3. *Let $\{F_n\}$ be an ensemble of functions, where $F_n : \{0,1\}^* \to \{0,1\}^{l(n)}$ is a random variable uniformly distributed over some set of functions Ω_n, where l is a fixed integer. If Ω_n consists of all possible functions from $\{0,1\}^*$ to $\{0,1\}^{l(n)}$, then $\{F_n\}$ is called a* **random function ensemble**.

We use $\mathcal{M}^f$ to denote the algorithm $\mathcal{M}$ with oracle access to the function f (i.e., he can adaptively issue a query x, and in return he will receive the function value $f(x)$). We call such an algorithm $\mathcal{M}$ an *oracle machine.*

Definition 4. *Let $\{H_n\}_n$ with $H_n : \{0,1\}^* \to \{0,1\}^{l(n)}$ be a random function ensemble. Assume $\{F_n\}_n$ with $F_n : \{0,1\}^* \to \{0,1\}^{l(n)}$ is an efficiently sampleable and efficiently computable function ensemble. $\{F_n\}_n$ is said to be* **pseudorandom** *if for any PPT oracle machine $\mathcal{M}$,*

$$|\Pr[\mathcal{M}^{F_n}(1^n) = 1] - \Pr[\mathcal{M}^{H_n}(1^n) = 1]| \tag{1}$$

is negligible.

Definition 5. *A family of efficiently computable functions $\{H_s\}_{s\in\{0,1\}^*}$ from $\{0,1\}^*$ to $\{0,1\}^{l(|s|)}$ is said to be* **Collision Resistant** *if for any PPT algorithm $\mathcal{A}$,*

$$\Pr[(x', x) \leftarrow \mathcal{A}(s) \textit{ for } s \leftarrow I(1^n) \textit{ s.t. } H_s(x') = H_s(x) \ \&\ x' \neq x] \tag{2}$$

is negligible, where $I(1^n)$ is the index generation for the function family $\{H_s\}$, and the probability is taken over internal coin flips in both $I(1^n)$ and $\mathcal{A}$.

3 Our New Pseudorandom Generator

In this section, we will introduce our new pseudorandom generator. Our generator is provably secure under the decisional Diffie-Hellman.

Let $p = 2q + 1$ and q be two large primes. Assume G_q is the subgroup of $\mathbb{Z}_p^*$ of order q and g is a generator of G_q. Let function $|\cdot|_p : G_q \to \mathbb{Z}_q$ be defined as follows:

$$|x|_p = \begin{cases} x & \text{if } 1 \leq x < q, \\ p - x & \text{if } q + 2 \leq x < p, \\ 0 & \text{otherwise.} \end{cases}$$

Note that G_q is exactly the set of quadratic residues in $\mathbb{Z}_p^*$. In addition, $\left(\frac{q(q+1)}{p}\right) = \left(\frac{1+q^{-1}}{p}\right) = \left(\frac{-1}{p}\right) = -1$ as $p \equiv 3 \pmod 4$. It follows that either $q \in G_q$ or $q + 1 \in G_q$ but not both. More precisely, by *Law of Quadratic Reciprocity* [25], we have that $q \in G_q$ if $q \equiv 1 \pmod 4$ and $q + 1 \in G_q$ if $q \equiv 3 \pmod 4$. Further notice that $\left(\frac{p-x}{p}\right) = -1, \forall x \in G_q$.

Therefore, we have that $|\cdot|_p$ is a 1-1 and onto mapping from G_q to $\mathbb{Z}_q$.

Construction 1. We define a number sequence generator NSG as follows. Let $(A_0, A_1) \in G_q \times G_q$ is the initial secret. Starting from $t = 2$, iteratively define $A_t = g^{|A_{t-1}|_p |A_{t-2}|_p}$. Let $A'_t = g^{|A_t|_p}$. The output sequence is $A'_0, A'_1, A'_2, \cdots$.

Now we show that our number sequence generator is secure under the decisional Diffie-Hellman (DDH) assumption.

Theorem 1. *Under the* decisional Diffie-Hellman *assumption, NSG is a pseudorandom number sequence generator.*

Proof. We need to show that $S_t = A'_0, A'_1, \ldots, A'_t$ is indistinguishable from $U_{t+1} \leftarrow G_q^{t+1}$ for any polynomially bounded t. We use a hybrid argument. Let $S_t^{(v)}$ be S_t, except $A'_0, \ldots, A'_v$ being taken uniformly random from G_q. Thus, $S_t^{(t)}$ is uniformly random in G_q^{t+1}. If there exists an adversary $\mathcal{M}$ distinguishing S_t from U_{t+1}, then $\mathcal{M}$ can distinguish $S_t^{(w)}, S_t^{(w+1)}$ for some $w \geq 1$ (Note the distributions of $S_1^{(0)}$ and $S_t^{(1)}$ are identical). Without loss of generality, assume w is known (otherwise, one can guess it correctly with probability $1/t$). We construct a PPT adversary $\mathcal{D}$ to break the DDH assumption. Upon receiving input (α, β, γ), $\mathcal{D}$ takes $A'_0 \leftarrow G_q, \ldots, A'_{w-2} \leftarrow G_q$, and defines $A'_{w-1} = \alpha, A'_w = \beta, A'_{w+1} = g^{|\gamma|_p}$. For $v = w+2, \ldots, t$, iteratively and normally computes $A_v = g^{|A_{v-1}|_p |A_{v-2}|_p}$ and $A'_v = g^{|A_v|_p}$. Note here $A_{w+2} = \beta^{|\gamma|_p}$. Finally, $\mathcal{D}$ feeds $A'_0, \ldots, A'_t$ to machine $\mathcal{M}$, and outputs whatever he does. When (α, β, γ) is a DH tuple, then the input to $\mathcal{M}$ is distributed exactly as $S_t^{(w)}$; if (α, β, γ) is a random tuple, then the input to $\mathcal{M}$ is distributed exactly as $S_t^{(w+1)}$. Thus, $\mathcal{D}$ has a non-negligible advantage, contradiction. ■

We have showed that Construction 1 is a pseudorandom number sequence generator. But in real applications, we are more interested in a binary generator. A naive idea is to encode the output of NSG into a binary form. However, one can easily show that it does not work. In the following, we construct a simple function to convert an NSG sequence into a binary pseudorandom sequence.

Construction 2. Let $A'_0, A'_1, \ldots$ be the output sequence of NSG in Construction 1. Let $L_k(x)$ be the k least significant bits of x, where k is a positive integer. Define $B_i = L_k(|A'_i|_p)$ for all $i \geq 0$. Then the output stream of the new generator is set to $B_0, B_1, \cdots$. Denote this binary generator by PSG_2.

Theorem 2. *If* $k = |q| - \omega(\log|q|)$, *then* PSG_2 *is a pseudorandom generator, where* $\omega(x)$ *means* $\lim_{x\to\infty} \frac{x}{\omega(x)} = 0$.

Proof. Denote $Z_t = B_0, B_1, \ldots, B_t$. Let U_t be a random variable uniform in $\{0,1\}^{kt}$. We need to show that Z_t is indistinguishable from U_{t+1} for any polynomially bounded t. Consider a random variable $\tilde{Z}_t = \tilde{B}_0, \tilde{B}_1, \ldots, \tilde{B}_t$, where $\tilde{B}_i = L_k(|C_i|_p)$ and $C_i \leftarrow G_q$ for all $i = 0, 1, \ldots, t$. To prove the theorem, it suffices to show that (1) Z_t and $\tilde{Z}_t$ are indistinguishable, and that (2) $\tilde{Z}_t$ and U_{t+1} are indistinguishable.

- *Z_t and $\tilde{Z}_t$ are indistinguishable.* If this is not true, there exists a PPT algorithm $\mathcal{D}_1$ to distinguish them. Then, one can constructs an algorithm $\mathcal{D}_1'$ to distinguish S_t (in Construction 1) from $V_{t+1} \leftarrow G_q^{t+1}$ as follows. $\mathcal{D}_1'$ first applies operator $L_k()$ to the received sequence, then feeds the produced sequence to $\mathcal{D}_1$ and outputs whatever he does. When the input to $\mathcal{D}_1'$ is S_t, then the input to $\mathcal{D}_1$ is distributed exactly as Z_t; otherwise, it is distributed as $\tilde{Z}_t$. Thus, a non-negligible advantage of $\mathcal{D}_1$ implies a non-negligible advantage of $\mathcal{D}_1'$, contradicting Theorem 1.
- *$\tilde{Z}_t$ and U_{t+1} are indistinguishable.* Note that $|\cdot|_p$ is 1-1 and onto mapping from G_q to $\mathbb{Z}_q$. Thus, if C_i is uniform in G_q, $|C_i|_p$ is uniform in $\mathbb{Z}_q$. Let $\tilde{C}_i = |C_i|_p$. Thus, defining $\tilde{B}_i = L_k(|C_i|_p)$ with $C_i \leftarrow G_q$ is equivalent to defining $\tilde{B}_i = L_k(\tilde{C}_i)$ with $\tilde{C}_i \leftarrow \mathbb{Z}_q$. We consider the latter when defining $\tilde{Z}_t$. Consider equation $X \equiv w \pmod{2^k}$ with an unknown X over $\mathbb{Z}_q$, where $w \in \{0, 1, \ldots, 2^k - 1\}$. For any $w \in \{0, 1, \ldots, 2^k - 1\}$, there are either $\lfloor \frac{q}{2^k} \rfloor$ or $\lfloor \frac{q}{2^k} \rfloor + 1$ solutions in $\mathbb{Z}_q$ for X. Thus, $\tilde{B}_i = w$ with probability $2^{-k} + \frac{\delta_w}{q}$ for some $\delta_w \in [-1, 1]$. Thus, the statistic distance between $\tilde{Z}_t$ and U_{t+1} (denoted by $\mathsf{dist}[\tilde{Z}_t, U_{t+1}]$) is at most $\sum_{i=0}^{t} \mathsf{dist}[\tilde{B}_i, U_{t+1}^{(i)}] \le \frac{(t+1)2^k}{q} \le \frac{t+1}{2^{\omega(\log|q|)}}$, negligible, where $U_{t+1}^{(i)}$ is the ith k-bit component of U_{t+1}. For any distinguisher $\mathcal{D}_2$, we have

$$\begin{aligned}
&|\Pr[\mathcal{D}_2(\tilde{Z}_t) = 1] - \Pr[\mathcal{D}_2(U_{t+1}) = 1]| \\
&= \textstyle\sum_{w \in \mathbb{Z}_q^{t+1}} |\Pr[\tilde{Z}_t = w]\Pr[\mathcal{D}_2(w) = 1] - \Pr[\mathcal{D}_2(w) = 1]\Pr[U_{t+1} = w]| \\
&= \textstyle\sum_{w \in \mathbb{Z}_q^{t+1}} \Pr[\mathcal{D}_2(w) = 1] |\Pr[\tilde{Z}_t = w] - \Pr[U_{t+1} = w]| \\
&\le \textstyle\sum_{w \in \mathbb{Z}_q^{t+1}} |\Pr[\tilde{Z}_t = w] - \Pr[U_{t+1} = w]| \\
&= \mathsf{dist}[\tilde{Z}_t, U_{t+1}],
\end{aligned}$$

negligible. ■

From Theorem 2, we immediately have the following corollary.

Corollary 1. *When* $k = |q| - \left\lfloor \frac{|q|}{c} \right\rfloor$ *for a constant* $c > 1$*, the resulting* PSG_2 *is a cryptographically secure pseudorandom generator.*

Corollary 2. *When* $k = |p| - \log^2 |p|$*, then the resulting* PSG_2 *is a pseudorandom generator.*

Remark 1. In each iteration, our generator involves two modular exponentiations. By Lim and Lee [30], one modular exponentiation can be done in $|p|/2$ modular multiplications, assuming pre-computation of $g^{2^i}, i = 0, \ldots, |p|$ and $g^{2^j + 2^{j+[|p|/2]}}, j = 0, \ldots, [|p|/2]$. So our generator can asymptotically output one bit for each multiplication modular a prime p. This result is better than other generators from $\mathbb{Z}_p^*$ [6, 31] that are provably secure under a standard assumption. More interestingly, in each iteration, our generator can output bits of length almost $|p|$, while no previous generator (including factoring based generator) proven secure in the standard assumption, has achieved this. Thus, the construction here might be interesting for motivating more efficient generators in the future. Note if a more complex version of pre-computation in [30] is adopted, then our generator can output $O(\log \log p)$ bits per modular multiplication. This is the best result in the literature. Specifically, it has been achieved by GR generator and DRV generator with even less precomputation than ours, and by ACGS generator [2] with no precomputation at all.

4 A New Family of Collision Resistant Hash Function

In this section, we construct a family of collision resistant hash function. We start with the following construction. This construction is essentially a realization of the framework [13] but waived of the extra requirement of making the input prefix-free. Later we will show how to obtain more efficient constructions.

Construction 3. Let $p = 2q + 1$ and q be two large primes, G_q be the subgroup of order q in $\mathbb{Z}_p^*$. Our hash family $\mathcal{H}_1$ is indexed by (g_0, g_1, s), where $g_0, g_1, s \leftarrow G_q$. Let H be a hash function in $\mathcal{H}_1$ with index (g_0, g_1, s). Upon input $x = x_1 x_2 \cdots x_t \in \{0, 1\}^*$, $H(x)$ is computed as follows. First set $Y_{t+1} = s$. For $i = t, t-1, \ldots, 1$, iteratively compute $Y_i = g_{x_i}^{|Y_{i+1}|_p}$. Finally, define $H(x) = Y_1$.

Theorem 3. *Assuming discrete log problem in G_q is hard, $\mathcal{H}_1$ is a collision resistant hash family. In addition, if the input is r bits, then assuming a pre-computation, one hash requires a cost of at most $r|p|/2$ modular multiplications.*

Proof. The second argument follows from the precomputation of

$$g_j^{2^i}, g_j^{2^{i+\lfloor\frac{|q|}{2}\rfloor}}, g_j^{2^i+2^{i+\lfloor\frac{|q|}{2}\rfloor}}, j=0,1, i=1,\cdots,\left\lceil\frac{|q|}{2}\right\rceil.$$

We thus concentrate on the first argument. Assume $H(x) = H(x')$ for some binary string $x = x_1x_2\cdots x_t$ and $x' = x'_1x'_2\cdots x'_l$ s.t. $x \neq x'$, for some integer l, l'. Without loss of generality, assume $l \geq t$. Then, there must exist a unique index j with $0 \leq j \leq t$ such that $x_1 = x'_1, \ldots, x_j = x'_j$ but $x'_{j+1} \neq x'_{j+1}$, where by default $x_{t+1} = \lambda$ (meaning empty). Let Y_i and Y'_i be the intermediate term with index i when computing $H(x)$ and $H(x')$, respectively. Since $|\cdot|_p$ is a 1-1 and onto mapping, we have the following result.

- Case $j < t$: We immediately have $Y_{j+1} = Y'_{j+1}$. However, $x_{j+1} \neq x'_{j+1}$ and $j+1 \leq t$, $g_{x_{j+1}}^{|Y_{j+2}|_p} = g_{x'_{j+1}}^{|Y'_{j+2}|_p}$. Thus, the discrete $\log_{g_0} g_1$ is obtained. If this event happens with non-negligible probability, we can transform the adversary to break the discrete log assumption, contradiction!
- Case $j = t$: In this case, since $x \neq x'$, it follows that $l > t$. Thus, $s = g_{x'_{t+1}}^{|Y'_{t+2}|_p}$. Therefore, we can obtain the discrete log either $\log_{g_1} s$ or $\log_{g_0} s$. If this happens with non-negligible probability, we can easily transform the adversary to break the discrete log assumption. ■

In the following, we present a more efficient construction of collision resistant hash family.

Construction 4 Let $p = 2q + 1$ and q be two large primes, G_q be the subgroup of order q in $\mathbb{Z}_p^*$. $|q| = k + 1$. Our hash family $\mathcal{H}_2$ is indexed by $(g_{00}, g_{01}, g_{10}, g_{11}, \ldots, g_{(k-1)0}, g_{(k-1)1})$, where $g_{ij} \leftarrow G_q$, $i = 0, \ldots, k-1, j = 0, 1$. Let H be the hash function in $\mathcal{H}_2$ with index $\{g_{ij} : 0 \leq i \leq k-1, j = 0, 1\}$. Upon input $x = x_1x_2\cdots x_t$ for $x_i \in \{0,1\}^k$ $(i < t)$ and $|x_t| \leq k$, $H(x)$ is computed as follows. Let $x_j = x_{j0}x_{j1}\cdots x_{j(k-1)}$ for $x_{jl} \in \{0,1\}$. For a l-bit string $z = z_0z_1\cdots z_{l-1}$ $(l \leq k)$, denote $g_z = \prod_{j=0}^{l-1} g_{jz_j}$. $Y_{t+1} = s$. For $i = t, t-1, \ldots, 1$, iteratively compute $Y_i = g_{x_i}^{|Y_{i+1}|_p}$. Finally, define $H(x) = Y_1$.

Theorem 4. *Under the discrete log assumption, $\mathcal{H}_2$ is a collision resistant hash family. In addition, if the input is r bits, one function evaluation costs no more than $3r$ modular multiplications.*

Proof. The second argument follows from the facts: an r-bit input is uniformly divided into $t = r/k$ segments; for each segment x_i, g_{x_i} is computed in k modular multiplications; each exponentiation in $\mathbb{Z}_p^*$ costs at most $2|p|$ modular multiplications. We thus focus on the first argument. We show that if the conclusion is wrong, we construct an algorithm $\mathcal{S}$ to solve the discrete log problem over G_q. Assume $\mathcal{H}_2$ is broken by an adversary $\mathcal{A}$. Then $\mathcal{S}$ is constructed as follows. Upon input $g, h \in G_q$, take $w_{ij}^0, w_{ij}^1 \leftarrow Z_q$, and define $g_{ij} = g^{w_{ij}^0} h^{w_{ij}^1}$, for $i = 0, 1, \ldots, k-1, j = 0, 1$. $\mathcal{S}$ provides $p, (g_{10}, g_{11}, \cdots, g_{(k-1)0}, g_{(k-1)1})$ to $\mathcal{A}$ and in turn receives a collision pair x, x' ($x \neq x'$) from $\mathcal{A}$. Assume $x = x_1 x_2 \cdots x_t$ and $x' = x'_1 x'_2 \cdots x'_{t'}$ for some $t, t' > 0$. W.L.O.G, assume $t \leq t'$. Let J be the smallest index such that $x_J \neq x'_J$. Let $Y_{t+1} = Y'_{t'+1} = s$, iteratively define $Y_i = g_{x_i}^{|Y_{i+1}|_p}$ and $Y'_j = g_{x'_j}^{|Y'_{j+1}|_p}$. Thus, we have $Y_J = Y'_J$. So

$$\begin{aligned} & g^{|Y_{J+1}|_p \sum_{i:x_{Ji}=1} w_{Ji}^0} h^{|Y_{J+1}|_p \sum_{i:x_{Ji}=1} w_{Ji}^1} \\ = {} & g^{|Y'_{J+1}|_p \sum_{i:x'_{Ji}=1} w_{Ji}^0} h^{|Y'_{J+1}|_p \sum_{i:x'_{Ji}=1} w_{Ji}^1} \end{aligned}$$

If $|Y'_{J+1}|_p \sum_{i:x'_{Ji}=1} w_{Ji}^1 \neq |Y_{J+1}|_p \sum_{i:x_{Ji}=1} w_{Ji}^1 \pmod q$, then discrete log $\log_g h$ can be efficiently obtained from the above relation. On the other hand, we show the probability that this condition is violated for some J is negligible. Indeed, let $v = \log_g h, z_{ij} = \log_g g_{ij}$. Then given $z_{ij} = w_{ij}^0 + v w_{ij}^1, i = 0, 1, \ldots, k-1, j = 0, 1$. Thus, given $\{g_{ij} : i = 0, 1, \ldots, k-1, j = 0, 1\}$, $\{w_{ij}^1 : i = 0, \ldots, k-1, j = 0, 1\}$ is independent of the view of $\mathcal{A}$. Thus, $|Y'_{J+1}|_p \sum_{i:x'_{Ji}=1} w_{Ji}^1 = |Y_{J+1}|_p \sum_{i:x_{Ji}=1} w_{Ji}^1$ (mod q) holds for a particular J with probability $1/q$. So the probability that there exists a J violating the condition is no more than k/q. ■

Now we further improve Construction 4 to reduce the computation cost by factor $\log k$ but increase the storage by a factor k.

Construction 5 Denote the new hash family by $\mathcal{H}_3$. The construction is similar to $\mathcal{H}_2$. But $\mathcal{H}_3$ is indexed by $(g_{00}, g_{01}, g_{0(k-1)}, g_{10}, g_{11}, \cdots, g_{(k-1)(k-1)})$, where $g_{ij} \leftarrow G_q$, $i = 0, \ldots, k-1, j = 0, \ldots, k-1$. Let H be the hash function in $\mathcal{H}_3$ with index $\{g_{ij} : 0 \leq i \leq k-1, 0 \leq j \leq k-1\}$. Upon input $x = x_1 x_2 \cdots x_t$ for $x_i \in \{0, \ldots, k-1\}^k$ ($i < t$) and $|x_t| \leq k$, $H(x)$ is computed as follows. Let $x_j = x_{j0} x_{j1} \cdots x_{j(k-1)}$ for $x_{jl} \in \{0, \ldots, k-1\}$. For a k-ary string $z = z_0 z_1 \cdots z_{l-1}$ ($l \leq k$), denote $g_z = \prod_{j=0}^{l-1} g_{jz_j}$. $Y_{t+1} = s$. For $i = t, t-1, \ldots, 1$, iteratively compute $Y_i = g_{x_i}^{|Y_{i+1}|_p}$. Finally, define $H(x) = Y_1$.

Theorem 5. *Under the discrete log assumption, $\mathcal{H}_3$ is a collision resistant hash family. In addition, if the input is r bits, one function costs no more than $3r/\log k$ modular multiplications.*

The proof of the theorem is almost identical to Theorem 4. Construction 5 is computationally more efficient than Construction 4 but it requires more storage. Thus, these two constructions have their own merit of existence.

Remark 2. Our CRHF $\mathcal{H}_3$ is the first construction provably secure in the standard assumption and only requires $O(1/\log k)$ time for each bit. Bellare et al. [4] proposed a notion of *incrementality* for CRHF, which means one can evaluate $H(x|y)$ from $H(x)|y$ instead of from the scratch. Our constructions satisfy this property too.

5 Application to Pseudorandom Function

In this section, we unite our pseudorandom generator and collision resistant hash function by employing them to construct (universal) pseudorandom functions. To do this, we first show that a collision resistant hash function can be used to extend an input-restricted pseudorandom function to a universal pseudorandom function. To be specific, we just use state this result in term of an input-restricted GGM construction [21].

Construction 6. Let $G : \mathcal{D} \to \mathcal{D}^2$ be a pseudorandom generator. Assume $G(x) = G_0(x)|G_1(x)$, where $G_i(x) \in \mathcal{D}, i = 0, 1$. H is a collision resistant hash function. A family of function $F : \mathcal{K} \times \{0,1\}^* \leftarrow \mathcal{D}$ is defined as follows. Given a private index $k \in \mathcal{K}$ and input $x \in \{0,1\}^*$, compute $u_0 u_1 \cdots u_{t-1} = H(x)$, $F_k(x)$ is defined to be $G_{u_{t-1}} \circ G_{u_{t-2}} \circ \cdots \circ G_{u_0}(k)$.

Theorem 6. *Let H be a collision resistant hash function, G is a pseudorandom generator. Then $\mathcal{F}$ is a family of pseudorandom function.*

Proof. Let X be the input queried by adversary. If a collision in X under $H()$ (i.e., $\exists x_1, x_2 \in X$ with $H(x_1) = H(x_2)$) happens with non-negligible probability, then $H()$ is not collision resistant. Otherwise, the proof is identical to the proof of Theorem 3.6.5 in [20]. The details are omitted. ■

Corollary 3. *Let $\mathcal{H}$ be the collision resistant hash family in Construction 5, and G be the pseudorandom generator in Construction 2. Then for l-bit input x, our pseudorandom function can be computed in roughly $(\frac{3l}{\log k} + 2k^2)$ modular multiplications.*

Remark 3. If we do not apply $\mathcal{H}$ to the input first, then the underlying pseudorandom function (by using construction 3.6.6 in [20]) requires more than $2lk$ modular multiplications, inefficient!

References

1. L. M. Adleman, A Subexponential Algorithm for the Discrete Logarithm Problem with Applications to Cryptography (Abstract), *FOCS 1979*, pp. 55-60, 1979.
2. W. Alexi, B. Chor, O. Goldreich, and C. Schnorr, RSA/Rabin Bits are 1/2 + 1/poly(log N) Secure, *FOCS 1984*: 449-457.
3. M. Bellare and S. Goldwasser, Verifiable Partial Key Escrow, *ACM CCS'97*, pp. 78-91, 1997.
4. M. Bellare, D. Micciancio, A New Paradigm for Collision-Free Hashing: Incrementality at Reduced Cost, *Advances in Cryptology-EUROCRYPT 1997*, pp. 163-192, 1997.
5. L. Blum, M. Blum, M. Shub, A Simple Unpredictable Pseudo-Random Number Generator, *SIAM J. Comput.* 15(2): 364-383 (1986).
6. M. Blum, S. Micali, How to Generate Cryptographically Strong Sequences of Pseudo Random Bits, *FOCS 1982*: 112-117.
7. D. Coppersmith, A. M. Odlyzko, and R. Schroeppel, Discrete Logarithms in GF(p), *Algorithmica* 1(1): 1-15 (1986).
8. R. Canetti and H. Krawczyk, analysis of key-exchange protocols and their use for building secure channels, *Advances in Cryptology-EUROCRYPT 2001*, B. Pfitzmann (Ed.), LNCS 2045, Springer-Verlag, pp. 453-474, 2001.
9. R. Canetti and H. Krawczyk, universally composable notions of key exchange and secure channels, *Advances in Cryptology-EUROCRYPT 2002*, L. R. Knudsen (Ed.), LNCS 2332, Springer-Verlag, pp. 337-351, 2002.
10. R. Canetti and H. Krawczyk, security analysis of IKE's signature-based key-exchange protocol, *Advances in Cryptology-CRYPTO 2002*, M. Yung (Ed.), LNCS 2442, Springer-Verlag, pp. 143-161, 2002.
11. S. Contini and A.K. Lenstra and R. Steinfeld, VSH: an Efficient and Provable Collision Resistant Hash Function, *NIST Cryptographic Hash Workshop 2005*, Maryland, USA, 2005.
12. R. Cramer and V. Shoup, a practical public-key cryptosystem provably secure against adaptive chosen ciphertext attack, *Advances in Cryptology-CRYPTO 1998*, H. Krawczyk (Ed.), LNCS 1462, Springer-Verlag, pp. 13-25, 1998.
13. I. Damgard, Collision Free Hash Functions and Public Key Signature Schemes, *EUROCRYPT 1987*: 203-216.
14. W. Diffie and M. E. Hellman, New Directions in Cryptography, *IEEE Transactions on Information Theory*, vol. IT-22, Nov. 1976, pp: 644-654.
15. N. Dedic, L. Reyzin and S. Vadhan, An Improved Pseudorandom Generator Based on Hardness of Factoring, *Security in Communication Networks 2002*, S. Cimato et al. (Eds.), LNCS 2576, Springer-Verlag, pp. 55-73, 2003.

16. T. El Gamal, a public-key cryptosystem and a signature scheme based on discrete logarithms, *IEEE Transactions on Information Theory*, Vol. 31, No. 4, pp. 469-472, 1985.
17. R. Gennaro, An Improved Pseudo-random Generator Based on the Discrete Logarithm Problem , *Journal of Cryptology*, 18(2), pp.91-110, Spring 2005. Early version appeared in CRYPTO'2000.
18. S. Goldwasser, S. Micali and P. Tong, Why and how to establish a private code on a public network, *FOCS'82*, pp. 134-144.
19. S. Goldwasser, S. Micali, and R. L. Rivest, A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks, *SIAM J. Comput.*, 17(2): 281-308 (1988).
20. O. Goldreich, *Foundations of Cryptography: Basic Tools*, Cambridge University Press, 2001.
21. O. Goldreich, S. Goldwasser amd S. Micali, How to Construct Random Functions, *Journal of the ACM*, Vol 33, No. 4, pp. 792-807, 1986.
22. O. Goldreich, V. Rosen, On the Security of Modular Exponentiation with Application to the Construction of Pseudorandom Generators, *J. Cryptology*, 16(2): 71-93 (2003).
23. J. Hastad, R. Impagliazzo, L. A. Levin, Michael Luby, A Pseudorandom Generator from any One-way Function, *SIAM J. Comput.* 28(4): 1364-1396 (1999). Early verision is in STOC'89.
24. J. Hastad, A. Schrift and A. Shamir, The Discrete Logarithm Modulo a Composite Hides $O(n)$ Bits, *JCSS*, 47: 376-404, 1993.
25. L. Hua, *Introduction to Number Theory*, Berlin: Springer-verlag, 1982.
26. H. Krawczyk, HMQV: A High-Performance Secure Diffie-Hellman Protocol, *Advances in Cryptology-CRYPTO 2005*, V. Shoup (Ed), Springer-Verlag, LNCS 3621, pp. 546-566, 2005.
27. S. Jiang and G. Gong, password based key exchange with mutual authentication, *Selected Area in Cryptography 2004*, H. Handschuh and A. Hasan (Eds.), LNCS 3357, Springer-Verlag, pp. 271-283, 2005.
28. J. Katz, R. Ostrovsky and M. Yung, efficient password-authenticated key exchange using human-memorable passwords, *Advances in Cryptology-EUROCRYPT 2001*, B. Pfitzmann (Ed.), LNCS 2045, Springer-Verlag, pp. 475-494, 2001.
29. A. K. Lenstra and H. W. Lenstra, Jr. (Eds.), the Developement of the Number Field Sieve, LNM 1554, Springer-Verlag, 1993.
30. C. Lim, P. Lee, More Flexible Exponentiation with Precomputation, *Advances in Cryptology-CRYPTO 1994*, Y. Desmedt (Ed.), LNCS 839, Springer-Verlag, pp. 95-107, 1994.
31. D. L. Long, A. Wigderson, How Discreet is the Discrete Log, *STOC 1983*, pp. 413-420, 1983.
32. A. M. Odlyzko, Discrete Logarithms: The Past and the Future, *Des. Codes Cryptography* 19(2/3): 129-145 (2000).
33. S. Patel and G. S. Sundaram, An Efficient Discrete Log Pseudo Random Generator, *Advances in Cryptology-CRYPTO 1998*, H. Krawczyk (Ed.), LNCS 1462, Springer-Verlag, pp. 304-317, 1998.
34. C. Peikert and A. Rosen, Efficient Collision-Resistant Hashing From Worst-Case Assumptions on Cyclic Lattices, *TCC 2006.*
35. D. Pointcheval, The Composite Discrete Logarithm and Secure Authentication, *Public Key Cryptography 2000*, H. Imai and Y. Zheng (Eds.), LNCS 1751, Springer-Verlag, pp. 113-128, 2000.

36. R. Rivest, A. Shamir and L. Adleman, A Method for Obtaining Digital Signatures and Public-key Cryptosystems, *Communications of ACM*, Vol. 2, pp. 120-126, February 1978.
37. A. Shamir and Y. Tauman, Improved Online/Offline Signature Schemes, *Advances in Cryptology-CRYPTO 2001*, J. Kilian (Ed.), LNCS 2139, Springer-Verlag, pp. 355-367, 2001.
38. O. Schirokauer, Discrete Logarithm and Local Units, *Philosophical Transactions: Physical Science and Engineering*, Vol. 345, No. 1676, pp. 409-423, 1993.
39. Victor Shoup: Lower Bounds for Discrete Logarithms and Related Problems, *Advances in Cryptology-EUROCRYPT 1997*, W. Fumy (Ed.), LNCS 1233, Springer-Verlag, pp. 256-266, 1997.
40. U. Vazirani and V. Vazirani, Efficient and Secure Pseudo-random number generation, *FOCS'84*, pp. 458-463.
41. X. Wang, X. Lai, D. Feng, H. Chen and X. Yu, Cryptanalysis of the Hash Functions MD4 and RIPEMD, *Advances in Cryptology-EUROCRYPT 2005*, R. Cramer (Ed.), LNCS 3494, Springer-Verlag, pp. 1-18, 2005.
42. X. Wang and H. Yu, How to Break MD5 and Other Hash Functions, *Advances in Cryptology-EUROCRYPT 2005*, R. Cramer (Ed.), LNCS 3494, Springer-Verlag, pp. 19-35, 2005.
43. X. Wang, Y. L. Yin and H. Yu, Finding Collisions in Full SHA-1, *Advances in Cryptology-CRYPTO 2005*, V. Shoup (Ed.), LNCS 3621, Springer-Verlag, pp. 17-36, 2005.
44. A. Yao, Theory and Applications of Trapdoor Functions (Extended Abstract), *FOCS 1982*: 80-91.