Direct Zero Knowledge Proofs of Computational Power in Five Rounds

Tatsuaki Okamoto† *David Chaum*‡ *Kazuo Ohta*†

†NTT Laboratories
Nippon Telegraph and Telephone Corporation
1-2356, Take, Yokosuka-shi, Kanagawa-ken, 238-03 Japan

‡Centre for Mathematics and Computer Science
Kruislaan 413, 1098SJ, Amsterdam, Netherlands

Abstract

Zero-knowledge proofs of *computational power* have been proposed by Yung and others. In this paper, we propose an *efficient (direct) and constant round (five round)* construction of zero knowledge proofs of computational power. To formulate the classes that can be applied to these efficient protocols, we introduce a class of invulnerable problems, $FewPR$ and $FewPR_U$. We show that any invulnerable problem in $FewPR$ and $FewPR_U$ has an efficient and constant round zero knowledge proof of computational power, assuming the existence of a one-way function. We discuss some applications of these zero-knowledge proofs of computational power.

1. Introduction

Zero knowledge interactive proofs that were originally introduced by Goldwasser, Micali, and Rackoff [GMR] were defined for "membership" problems, in which the membership of an instance in language L is demonstrated. On the other hand, two other types of (zero-knowledge) interactive proofs have been proposed; one is proofs of "knowledge" [FFS, TW], in which prover's possession of knowledge is demonstrated, and the other is proofs of "computational power"[Y, K, OkOh, BDLP], in which prover's computational power or ability of solving a problem is demonstrated.

In many cases, a protocol constructed for a zero-knowledge proof of "membership" is also a protocol for a zero-knowledge proof of "knowledge", if the infinite power prover in the former one can be replaced by a poly-time bounded prover with an auxiliary input in the latter one. On the other hand, zero-knowledge proofs of "computational power" are quite different from zero-knowledge proofs of "membership". For example, a protocol of a zero-knowledge proof of "membership" usually cannot be a protocol for a zero-knowledge proof of "computational power", although the former can be used for constructing the latter as a subprotocol.

Similar to the relationship between zero-knowledge proofs of "membership" and "knowledge", there is also a relationship between zero-knowledge proofs of "computational power" and "knowledge". Note that, therefore, we have two types of zero-knowledge proofs of "knowledge"; one is related to "membership" zero-knowledge proofs

D.W. Davies (Ed.): Advances in Cryptology - EUROCRYPT '91, LNCS 547, pp. 96-105, 1991.

(where knowledge is a "witness of the related language") and the other is related to "computational power" zero-knowledge proofs (where knowledge is a "key of the related trapdoor function").

Regarding zero-knowledge proofs of "membership" and its related proofs of "knowledge", some efficient and constant round (five or four round)[1] constructions have been proposed [BMO, FFS, FeS, TW]. However, these results do not imply that we can also construct an efficient zero-knowledge proofs of "computational power", since, as mentioned above, zero-knowledge proofs of "computational power" are quite different from those of "membership". Actually, the previously proposed zero-knowledge proofs of "computational power" [Y, K] are very inefficient, because these proofs use some zero-knowledge proofs of knowledge as subprotocols repeated polynomially many times. Moreover, no direct construction of the zero-knowledge proof of knowledge used as subprotocol in these zero-knowledge proofs is given, since its construction is only guaranteed by an inefficient (even though polynomial-time) reduction to a zero-knowledge proof for an NP-complete predicate. On the other hand, zero-knowledge proofs of "computational power" by [OkOh, BDLP] need specific and strong cryptographic assumptions, although they are efficient ([OkOh] is three rounds, and [BDLP] is four rounds).

In this paper, we propose *efficient (direct) and constant round (five round)* zero knowledge interactive proofs of "computational power" assuming the existence of a secure bit-commitment (or of a one-way function [H, ILL, N]). In order to formulate the classes that can be applied to these efficient proofs, we introduce the class of invulnerable problems, $FewPR$ and $FewPR_U$. We show that any invulnerable problem in $FewPR$ and $FewPR_U$ has an efficient (direct) and constant round (five round) zero knowledge proof of computational power, assuming the existence of a secure bit-commitment.

As an application of this efficient zero knowledge proof of "computational power", we show an efficient zero knowledge proof of "knowledge", in which prover's possession of knowledge (a key of the related trapdoor function) is demonstrated. As a typical example of this application, we show a zero-knowledge proof of possessing prime factors of a composite number, which is much more efficient than the previously proposed protocol in [TW]. We also discuss another application for an identification scheme.

2. Notations

$\nu(n)$ denotes a function vanishing faster than the inverse of any polynomial. Formally,

$$\forall c > 0\ \exists d > 0\ \forall n > d \qquad \nu(n) < 1/n^c.$$

Negligible probability is the probability that behaves as $\nu(n)$, and overwhelming probability is probablity behaves as $1 - \nu(n)$. When S is a set, $\| S \|$ denotes the number of elements in the set S. $\oplus$ denotes the exclusive-or.

(P, V) is an interactive pair of Turing machines, where P is the prover, and V is the verifier [GMR, TM]. Let $T \in \{P, V\}$. $T(s)$ denotes T begun with s on its input work tape. $(P, V)(x)$ refers to the probability space that assigns to the string σ the probability that (P, V), on input x, outputs σ.

[1] Here, one "round" means one message transmission. Note that "round" is used as a couple of message transmissions (send and return) in [FeS].

3. Invulnerable Problems

In this section, we define the problems whose instance-witness pairs are efficiently generated. These are variants of the *invulnerable* problems introduced by Abadi, Allender, Broder, Feigenbaum, and Hemachandra [AABFH].

Definition 1: R denotes a *predicate* that can be computed in polynomial time by a deterministic algorithm.

$$S_R = \{(x, w) \mid R(x, w)\},$$

$$\mathcal{X}_R = \{x \mid \exists w\ R(x, w)\},$$

$$\mathcal{W}_R(x) = \{w \mid R(x, w)\},$$

$$FewPR = \{R \mid \exists c > 0\ \forall x \in \mathcal{X}_R \ \| \mathcal{W}_R(x) \| \leq |x|^c\},$$

R_U denotes a *predicate with an auxiliary witness* that can be computed in polynomial time by a deterministic algorithm with an auxiliary witness.

$$S_{R_U} = \{(x, w, u) \mid R_u(x, w)\},$$

$$\mathcal{X}_{R_U} = \{x \mid \exists w\ \exists u\ R_u(x, w)\},$$

$$\mathcal{W}_{R_U}(x) = \{w \mid R_u(x, w)\},$$

$$FewPR_U = \{R_u \mid \exists c > 0\ \forall x \in \mathcal{X}_{R_U} \ \| \mathcal{W}_{R_U}(x) \| \leq |x|^c\},$$

Note: In our definition, when $R \in FewPR$ (or $R_U \in FewPR_U$) is determined, the size of $x \in \mathcal{X}_R$ (or $x \in \mathcal{X}_{R_U}$) is fixed. In other words, R is defined for each size of $x \in \mathcal{X}_R$. We write $|R|$ as $|x|$ $(x \in \mathcal{X}_R)$.

Definition 2: The *uniform generation scheme* for $R \in FewPR$, which we denote G_R, is a polynomial-time algorithm that, on R, obtains an output string y. If y is $(x, w) \in S_R$ with uniform probability, where $R(x, w)$, then G_R outputs (x, w); otherwise, it outputs Λ. G_R is *α-invulnerable*, if the probability that there exists a polynomial-time algorithm computing w from x for nonnegligible fraction of $x \in \mathcal{X}_R$ or G_R outputs Λ is at most $1 - \alpha$.

The *uniform generation scheme* for $R_U \in FewPR_U$, which we denote G_{R_U}, is a polynomial time algorithm that, on input R_U, obtains an output string y. If y is $(x, w, u) \in S_{R_U}$ with uniform probability, where $R_u(x, w)$, then G_{R_U} outputs (x, w, u); otherwise, it outputs Λ. G_{R_U} is *α-invulnerable*, if the probability that there exists a polynomial-time algorithm computing w from x and R_U for nonnegligible fraction of $x \in \mathcal{X}_{R_U}$ or G_{R_U} outputs Λ is at most $1 - \alpha$.

R or R_U is α-invulnerable, if there exists an α-invulnerable uniform generation scheme. A subset $\mathcal{C} \subset FewPR$ is α-invulnerable, if any $R \in \mathcal{C}$ is α-invulnerable. A subset $\mathcal{C} \subset FewPR_U$ is α-invulnerable, if any $R_U \in \mathcal{C}$ is α-invulnerable. When $1 - \alpha$ is negligible, *α-invulnerable* is refered to simply as *invulnerable*.

Example 1: (Invulnerable problem in $FewPR$)

$$R(x, w) : x = E(w),$$

where E is a one-way function (e.g., an encryption function).

Example 2: (Invulnerable problem in $FewPR$)

$$R(x, w) : x = w^e \bmod n,$$

where $n = p \cdot q$ (p, q: prime), e is coprime to $p-1$ and $q-1$. If breaking the RSA scheme is hard, then this is an invulnerable problem in $FewPR$.

Example 3: (Invulnerable problem in $FewPR_U$)

$$R_u(x, w) : x = g^u \bmod p, \quad w = b^u \bmod p,$$

where p is prime, g is a primitive element of $GF(p)$, and b is an element in $GF(p)$ ($b = g^a \bmod p$). If breaking the DH scheme is hard, then this is an invulnerable problem in $FewPR_U$.

Remark: Hereafter, we will only talk about $FewPR$ and omit $FewPR_U$, because they are almost same.

4. Interactive Proofs of Computational Power

In this section, we introduce the formal definition of interactive proofs of computational power.

Definition 3: (P, V) is an interactive proof that the prover P has the computational power to solve the invulnerable problem $\mathcal{C} \subset FewPR$, if and only if it satisfies the following two conditions

- Completeness:

  For any $R \in \mathcal{C}$,

$$\Pr\{(P, V) \text{ accepts } R\} > 1 - \nu(|R|).$$

The probability is taken over the coin tosses of P and V.

- Soundness:

  For any $c > 0$, there exists a polynomial-time probabilistic algorithm M (with complete control over P^*) such that, for any machine P^* acting as a prover, and any sufficiently large $R \in \mathcal{C}$,

$$\Pr\{(P^*, V) \text{ accepts } R\} > 1/|R|^c \quad \rightarrow \quad \Pr\{M(x) = \mathcal{W}_R(x),\ x \in \mathcal{D}_R\} > 1 - \nu(|R|),$$

where $\mathcal{D}_R \subset \mathcal{X}_R$, $\| \mathcal{D}_R \| < |R|^{c'}$ for a constant c', and $\mathcal{D}_R$ is randomly selected from $\mathcal{X}_R$ with the same distribution as generated by verifier V. The probability is taken over the coin tosses of P^*, M, V, and the distribution of $\mathcal{D}_R$.

5. Efficient Zero Knowledge Interactive Proof of Computational Power to Solve an Invulnerable Problem in FewPR

In this section, we show that any invulnerable problem in $FewPR$ (and $FewPR_U$) has an efficient (direct) and constant round (five round) zero-knowledge interactive

proof of computational power assuming the existence of a secure bit-commitment (or of a one-way function).

First, before describing the construction of zero-knowledge proofs of computational power, we roughly introduce the notion of the bit-commitment (see the formal definition in [N]). The bit-commitment protocol between Alice (committer) and Bob (verifier) consists of two stages; the *commit stage* and the *revealing stage*. In the commit stage, after their exchanging messages, Bob has some information that represents Alice's secret bit b. In the revealing stage, Bob knows b. Roughly, after the commit stage, Bob cannot guess b, and Alice can reveal only one possible value. Here, when the protocol needs k rounds in the commit stage, we call it *k round* bit-commitment. We can construct *one round* bit-commitment using probabilistic encryption [GM], and *two round* bit-commitment using any one-way function [H, ILL, N]. In the commit stage of one round bit-commiment, Alice generates a random number r and sends $BC(b, r)$ to Bob, where BC is a bit-commit function. When Alice wishes to commit l bits, $b_1, b_2, \ldots, b_l$, she sends $BC(b_1, r_1)$, ..., $BC(b_l, r_l)$ to Bob. Hereafter, we will simply write $BC(\overline{b}, \overline{r})$ as $(BC(b_1, r_1), \ldots, BC(b_l, r_l))$, where $\overline{b} = (b_1, \ldots, b_l)$ and $\overline{r} = (r_1, \ldots, r_l)$.

Next, we show the protocol of an efficient (direct) and constant round (five round) zero-knowledge interactive proof of computational power.

Protocol A

Let $\mathcal{C} \subset FewPR$ be an invulnerable problem. The following (P, V) protocol on input $R \in \mathcal{C}$ should be repeated $m = O(|R|)$ times.

(i) P generates random bit strings r_i and t_i $(i = 1, 2, \ldots, I)$, and calculates bit-commitment $v_i = BC(r_i, t_i)$ $(i = 1, 2, \ldots, I)$, where BC is a bit-commitment function, and t_i is a random bit string to commit r_i. Here, I is the maximum number of witnesses of R with $|R|$.

(ii) V randomly generates $(x, w) \in S_R$ using G_R, and sends x to P. If G_R does not output anything, V sends *terminate* to P and halts.

(iii) P generates random bit strings s_i, and calculates $u_i = s_i \oplus r_i$ $(i = 1, 2, \ldots, I)$. P computes $\{w_i \mid R(x, w_i), i = 1, 2, \ldots, I\}$ from x using P's computational power, and calculates $z_i = BC(w_i, s_i)$ $(i = 1, 2, \ldots, I)$. (When the number of the witnesses of x is less than I, P set any values as the dummy witnesses.) If P cannot find any witness w_i, then P sends *terminate* to V and halts. Otherwise, P sends u_i, z_i $(i = 1, 2, \ldots, I)$ to V.

(iv) V sends w to P.

(v) P checks whether there exists j such that $w = w_j$. If j exists, V sends j, r_j and t_j, otherwise P halts.

(vi) V checks whether $v_j = BC(r_j, t_j)$ and $z_j = BC(w, (u_j \oplus r_j))$ hold or not. If they hold, V continues the protocol, otherwise rejects and halts.

After m repetitions of this protocol, if V has not rejected it, V accepts and halts.

Remark: If Naor's bit-commitment scheme [N] is used, V also sends random bits for P's bit-commitment before steps (i) and (iii). Note that if probabilistic encryption or blob is used as a bit-commitment, V does not need to send any information for P's bit-commitment.

Theorem 1: If there exists a secure (one round) bit-commitment, then Protocol A is a computational zero knowledge interactive proof that the prover has the computational power to solve $R \in \mathcal{C}$.

(Proof)

(Completeness)

Since P has the power of computing $\{w_i \mid R(x, w_i), i = 1, 2, \ldots, I\}$, then clearly P can be proven valid with probability 1.

(Soundness)

Here, for any $c > 0$, we construct a polynomial time algorithm M such that when for any sufficiently large $R \in \mathcal{C}$

$$\Pr\{(P^*, V) \text{ accepts } R\} > 1/|R|^c,$$

then

$$\Pr\{M(x) = \mathcal{W}_R(x),\ x \in \mathcal{D}_R\} > 1 - \nu(|R|).$$

This algorithm consists of two phases. In the first phase, M executes protocol A with P^* by simulating the honest verifier, and obtains the values of random numbers t_i, and r_i generated by P^* with nonnegligible probability. By repeating polynomially many times this procedure with fixing the random tape of P^*, M obtains all t_i, and r_i that are used for commiting true witnesses but dummy witnesses.

In the second phase, M resets P^*, and also executes protocol A, where M does not simulate the honest verifier, but sends $x \in \mathcal{D}_R$ to P^* in step (ii). In step (iii), P^* outputs z_i. Because in step (iv) M cannot send a valid w, then P^* halts in step (v). However, since, in the first phase, M knows all t_i and r_i, then M can obtain all w_i from u_i, z_i, t_i, and r_i with nonnegligible probability, because the same random tape of P^* is used in the first and second phases.

Thus, we can extract all w_i from $x \in \mathcal{D}_R$ with overwhelming probability by repeating the above procedure polynomially many times.

(Zero knowledgeness)

Let V^* be any polynomial time algorithm for the verifier.

The simulator M can simulate the history, $(P,V^*)(R)$, by using V^* as a black-box as follows:

The following procedure should be repeated $m = O(|R|)$ times.

(1) M generates prover's first message, $\{v_i'; i = 1, 2, \ldots, I\}$, by using the same procedure as honest prover's, where $v_i' = BC(r_i', t_i')$.

(2) M gives $\{v_i'; i = 1, 2, \ldots, I\}$ to V^* as prover's first message and runs V^*, which outputs x' as verifier's first message.

(3) M generates random numbers $\{w_i'\}$, $\{s_i'\}$ and computes $\{u_i' = s_i' \oplus r_i'\}$, $\{z_i' = BC(w_i', s_i'\}$, where $i = 1, 2, \ldots, I$.

(4) M gives $\{u_i', z_i'; i = 1, 2, \ldots, I\}$ to V^* as prover's second message and runs V^*, which outputs w' as verifier's second message.

(5) M checks whether $R(x', w')$ holds or not. If $R(x', w')$ does not hold, then M outputs $(\{v_i'; i = 1, 2, \ldots, I\}, x', \{u_i', z_i'; i = 1, 2, \ldots, I\}, w')$, and halts. If it holds, then go to step (6).

(6) M resets V^*, and repeats the exact same procedure of steps (1) and (2) with the same ramdom tapes.

(7) M randomly generates $j \in \{1, \ldots, I\}$, s_i'' $(i = 1, \ldots, I)$ and w_i'' for $i = 1, \ldots, j-1, j+1, \ldots, I$. M sets $w_j'' = w'$, and computes $u_i'' = s_i'' \oplus r_i'$, $z_i'' = BC(w_i'', s_i'')$ $(i = 1, \ldots, I)$.

(8) M gives $\{u_i'', z_i''; i = 1, \ldots, I\}$ to V^* as prover's second message, and runs V^*, which outputs w'' as verifier's second message.

(9) M checks whether $w'' = w'$ or not. If it holds, M outputs $(\{v_i'; i = 1, \ldots, I\}, x', \{u_i'', z_i''; i = 1, \ldots, I\}, w', (j, t_j', r_j'))$ as the history. Otherwise, M returns to step (6).

First, the real history H_0 of $(\{v_i; i = 1, \ldots, I\}, x, \{u_i, z_i; i = 1, \ldots, I\}, w)$ is computationally indistinguishable from the first-stage simulated history H_1 of $(\{v_i'; i = 1, \ldots, I\}, x', \{u_i', z_i'; i = 1, \ldots, I\}, w')$. This is because if there exists a distinguisher D that distinguishes H_0 and H_1, then we can use D to extract one committed bit of the bit-commitment scheme with nonnegligible probability. This contradicts the definition of bit-commitment, since V^* and M are polynomial-time bounded and D is nonuniformly polynomial-time bounded.

Similarly, the first-stage simulated history H_1 is computationally indistinguishable from the final-stage simulated history H_2 of $(\{v_i'; i = 1, \ldots, I\}, x', \{u_i'', z_i''; i = 1, \ldots, I\}, w')$. This is because H_1 is computationally indistinguishable from $H_1' = (\{v_i'; i = 1, \ldots, I\}, x', \{R_i, z_i'; i = 1, \ldots, I\}, w')$, H_2 is computationally indistinguishable from $H_2' = (\{v_i'; i = 1, \ldots, I\}, x', \{R_i, z_i''; i = 1, \ldots, I\}, w')$, and H_1' and H_2' are computationally indistinguishable, where R_i is a real random string.

Thus, the output of M is computationally indistinguishable from the real history, since H_2 is computationally indistinguishable from H_0.

Then, we show that the running time of M is polynomial time with overwhelming probability. Suppose that p_i is the probability that V^* outputs w_i as w' on input $(\{v_i'; i = 1, \ldots, I\}, x', \{u_i', z_i'; i = 1, \ldots, I\})$ and p_i' is the probability that V^* outputs w_i as w' on input $(\{v_i'; i = 1, \ldots, I\}, x', \{u_i'', z_i''; i = 1, \ldots, I\})$. The expected repetition number from step (6) to (9) is $\sum_i p_i/p_i'$. From the property of the bit-commitment, $\varepsilon = |p_i - p_i'| < \nu(|R|)$. If there exists c_0 for sufficient large $|R|$ such that $p_i > 1/|R|^{c_0}$, then $|1 - p_i'/p_i| \leq \varepsilon/p_i < \nu(|R|)$. Hence, $\sum_i p_i/p_i' < I + \nu(|R|)$. If $p_i < \nu(|R|)$, then we can neglect the history with w_i, since the probability of the history is negligible and the history without w_i is statistically indistinguishable from the real history. Thus, the running time of the simulation is polynomial time with overwhelming probability.

(QED)

Notes:

(1) In the above protocol, the prover shows his power to compute all witnesses, $w_1, \ldots, w_I$, from instance x. Definition 3 and this protocol can be easily modified to those in which the prover shows his power to compute a subset of $\{w_1, \ldots, w_I\}$ from x. In the modified protocol, the verifier accepts the proof if the ratio of the number of the accepted cycles to the total cycle number is greater than a predetermined value (e.g., $1/(I - c)$; c is an arbitrary constant), where a"cycle" means step (i) to (vi) in Protocol A.

(2) The zero-knowledge protocol in the case of example 1 demonstrates that the prover can calculate E^{-1}.
(3) The zero-knowledge protocol in the case of example 2 demonstrates that the prover can decrypt an RSA ciphertext.
(4) The zero-knowledge protocol in the case of example 3 demonstrates that, given p, g, b, the prover can generate $b^u \bmod p$ from $g^u \bmod p$ without knowing u.

Corollary 1: Let $\mathcal{C} \subset FewPR$ or $FewPR_U$ be an invulnerable problem. If there exists a secure (one round) bit-commitment, then there exists a constant round (five round) computational zero knowledge interactive proof that the prover has the computational power to solve $R \in \mathcal{C}$.

(Proof Sketch)

Clearly, the parallel version of Protocol A satisfies the completeness and soundness conditions.

To prove the zero-knowledgeness, first, simulator M of this protocol executes the parallel version of steps (1) and (2) of simulator's procedure in Theorem 1's proof. Here, V^* outputs m instances, $x'_1, \ldots, x'_m$, as verifier's first messages. Then, M obtains witnesses of each instance x_k $(k = 1, \ldots, m)$ that V^* outputs nonnegligibly as verifier's second message, by repeating the parallel version of steps (6) through (9) of simulator's procedure in Theorem 1's proof. (Here, w''_j can be a random string in step (7).) The expected repetition number is polynomial in $|R|$. Then, M puts all obtained witnesses in z''_i (which corresponds to step (7)), and executes the procedure which corresponds to steps (8) and (9) of simulator's procedure in Theorem 1's proof.

Therefore, The running time of M is polynomial time with overwhelming probability, and the output of M is computationally indistinguishable from the real history.

(QED)

Corollary 2: Let $\mathcal{C} \subset FewPR$ or $FewPR_U$ be an invulnerable problem. If there exists a secure blob (chameleon bit-commitment [BCC]), then there exists a *perfect* zero knowledge *argument* that the prover has the computational power to solve $R \in \mathcal{C}$.

(Proof Sketch) If we use the same simulator M as that of Theorem 1, the output of M is perfectly indistinguishable to the real history of (P, V^*). The expected running time of M is $\sum_i p_i/p_i \leq I$.

(QED)

Corollary 3: Let $\mathcal{C} \subset FewPR$ or $FewPR_U$ be an invulnerable problem. If there exists a secure (one round) blob (chameleon bit-commitment [BCC]), then there exists a constant round (five round) *statistical* zero knowledge *argument* that the prover has the computational power to solve $R \in \mathcal{C}$.

(Proof Sketch) If we use the same simulator M as that of Corollary 1, the output of M is statistically indistinguishable to the real history of (P, V^*). The expected running time of M is polynomial time with overwhelming probability.

(QED)

Corollary 4: Let $\mathcal{C} \subset FewPR$ or $FewPR_U$ be an invulnerable problem. If there exists a one-way function, then there exists a constant round (six round) computational zero knowledge interactive proof that the prover has the computational power to solve $R \in \mathcal{C}$.

6. Applications

6.1 Zero-Knowledge Proofs for Possession of Knowledge

In this subsection, we show an efficient zero knowledge proof of "knowledge", based on our zero-knowledge proof of "computational power".

When the power of the prover's ability is bounded in polynomial-time, protocol A for example 2 (Section 3) becomes a zero knowledge proof of "knowledge", which demonstrates that the prover has a secret key or algorithm for decrypting an RSA ciphertext. In the same situation, protocol A for example 3 demonstrates that the prover has a secret key or algorithm for computing $b^u \bmod p$ from $g^u \bmod p$, g, b, and p. Here, note that we need another definition of interactive proofs of "knowledge" than that by [FFS, TW].

Simialrly, when E in example 1 is the Rabin scheme, we can construct a zero-knowledge proof of possessing prime factors of a composite number. Note that this protocol is a zero knowledge proof of "knowledge" in the sense of [FFS, TW]. Clearly, this protocol is much more efficient than the previously proposed protocol in [TW].

6.2 Identification Schemes

The zero-knowledge proofs of computational power can be applied to identification schemes. In the above-mentioned zero-knowledge proofs, we assume that the prover has infinite power. When we apply the zero-knowledge proofs to identification schemes, we must assume that the power of a prover is bounded in polynomial-time, and that the prover possesses a secret auxiliary input. Then, we use a trapdoor one-way function (public-key encryption) (E, d), where E is a function, and d is a secret key for the trapdoor (in other words, the inverse of E can be efficiently computed using d.) By using this trapdoor function, we can construct an invulnerable problem $R \in FewPR$ such that

$$R(x, w) : x = E(w),$$

and there exists a polynomial-time algorithm D_d, where $x = E(D_d(x))$. The identification scheme is as follows:

(Key generation)

First, prover P randomly generates (E, d), and publishes E as his public key (d is his secret key).

(Identification) P proves to a verifier that P has the power to compute the inverse of E through Protocol A.

Thus, we can construct an identification scheme if there exists a trapdoor one-way function.

7. Conclusion

In this paper, we have proposed an *efficient (direct) and constant round (five round)* construction of zero knowledge proofs of computational power. We have shown that any invulnerable problem in $FewPR$ and $FewPR_U$ has an efficient and constant round zero knowledge proof of computational power, assuming the existence of a one-way function.

We have discussed some applications of these zero-knowledge proofs of computational power.

Acknowledgements: We would like to thank Toshiya Itoh and Kouichi Sakurai for their invaluable discussions on the preliminary manuscript. We would also like to thank Ivan Damgård for his pointing out an error in the proof of Theorem 1 in the preliminary manuscript.

References

[AABFH] M.Abadi, E.Allender, A.Broder, F.Feigenbaum, and L.Hemachandra, "On Generating Solved Instances of Computational Problems," the Proceedings of Crypto (1988)

[BDLP] J.Brandt, I.Damgård, P.Landrock, T.Pedersen, "Zero-Knowledge Authentication Scheme with Secret Key Exchange," (Preprint)

[BMO] M.Bellare, S.Micali, and R.Ostrovsky, "Perfect Zero-Knowledge in Constant Rounds," the Proceedings of STOC, pp.482-493 (1990)

[FeS] U.Feige, A.Shamir, "Zero-Knowledge Proofs of Knowledge in Two Rounds," the Proceedings of Crypto'89, pp.526-544 (1989)

[FFS] U.Feige, A.Fiat and A.Shamir, "Zero Knowledge Proofs of Identity," the Proceedings of STOC, pp.210-217 (1987)

[FiS] A.Fiat and A.Shamir, "How to Prove Yourself," the Proceedings of Crypto (1986)

[GM] S.Goldwasser, S.Micali, "Probabilistic Encryption," Journal of Computer and System Science, pp270-299 (1984)

[GMR] S.Goldwasser, S.Micali, and C.Rackoff, "Knowledge Complexity of Interactive Proofs," the Proceedings of STOC, pp291-304 (1985)

[GMW] O.Goldreich, S.Micali, and A.Wigderson, "Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design," the Proceedings of FOCS, pp.174-187 (1986)

[H] J.Håstad, "Pseudo-Random Generators under Uniform Assumptions," the Proceedings of STOC, pp.395-404 (1990)

[ILL] R.Impagliazzo, L.Levin, M.Luby "Pseudo-Random Number Generation from One-Way Functions," the Proceedings of STOC, pp.12-24 (1989)

[K] K.Kurosawa, "Dual Zero Knowledge Interactive Proof Systems," Technical Report of the IEICE. Japan, ISEC88-33 (1988)

[OkOh] T.Okamoto, and K.Ohta, "Zero Knowledge Proofs for Possession of Blackboxes," SCIS'89 (in Japan) (1989)

[N] M.Naor, "Bit Commitment Using Pseudo-Randomness," the Proceedings of Crypto'89 (1989)

[TW] M.Tompa and H.Woll, "Random Self-Reducibility and Zero Knowledge Interactive Proofs of Possession of Information," the Proceedings of FOCS, pp.472-482 (1987)

[Y] M.Yung, "Zero-Knowledge Profs of Computational Power," the Proceedings of Eurocrypt'89, (1989)