Secure Bit Commitment Function against Divertibility

Kazuo Ohta Tatsuaki Okamoto Atsushi Fujioka

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

Abstract

Some zero-knowledge interactive proofs (*ZKIP*s) have divertibility, that is, evidence of proof issued by a genuine prover, A, can be transferred to plural verifiers, B and then C, where the intermediate verifier, B, acts as A, with A's help, to confound the other verifier C without revealing the relation between the A-B interaction and the B-C interaction. This property is a serious problem in practice, e.g. the mafia fraud attack on identification scheme and the multi-verifier attack against undeniable signatures.

This paper proposes a new concept, *security against divertibility*, and proves that Naor's bit commitment function based on pseudo-random generators is secure against divertibility under the reasonable assumption. Usage of this bit commitment in *ZKIP* can convert a *divertible ZKIP* to a *divertible-free-ZKIP* which is secure against the mafia fraud attack and the multi-verifier attack.

1 Introduction

Zero-knowledge interactive proofs [GMR] are an attractive concept in theory and in practice [GMW, FS, C]. In zero-knowledge proofs, coin flips of the prover are essential for zero-knowledgeness of the proof, while coin flips of the verifier are essential for soundness of the proof.

The usages of randomness has two sides: one positive and the other negative. On the positive side, the prover's randomness can be used to transfer some information in order to achieve positive applications, for example, identity based key distribution, digital signatures, etc [C, OO90]. On the negative side, it can be used to create a subliminal channel [DGB], where a prover can send an authenticated message, which contains a hidden message.

The verifier's randomness can be also used maliciously to realize the *mafia fraud attack* on the Fiat-Shamir scheme [DGB] and the *multi-verifier attack*

R.A. Rueppel (Ed.): Advances in Cryptology - EUROCRYPT '92, LNCS 658, pp. 324-340, 1993.

against Chaum's undeniable signature [DY], where an intermediate B can pass himself off as the genuine prover A to another verifier C, when A proves her identity or her signature to B, and B conceals any evidence that he used A's help. This concept was expressed in the *divertible zero knowledge interactive proof* and it has been proven that the commutative random-self reducible (CRSR) problem satisfies this property [OO89]. It has also been proven that wide classes of language, NP, have *divertible ZKIP* under some assumption [BD]. [1]

Recently, it has been proven that no undeniable signature scheme is secure against a multi-verifier attack provided that half of the verifiers are "honest (for their conspiracy)" in a more general setting than the divertible ZKIP, where the verifiers collaborate with a sub-protocol hidden from the prover, using the concept of secure function evaluations known as "mental games"[DY]. Though this negative result is theoretically exciting, there are two problems in practice (from the attacker's side). First, their collaboration can be detected by anyone who observes the transmitted data among users, since the relationship between the mental game protocol and the undeniable signature protocol can be traced. Second, if the majority of the verifiers are dishonest, the minority can believe a false proof by the majority. Therefore, in their protocol based on the mental game, a malicious verifier cannot be convinced of the correctness of the proof without believing that the majority of the verifiers are honest.

These problems imply the possibility of constructing a secure undeniable signature against a multi-verifier attack under reasonable constraints such as the non-detectivity and the dishonest majority of the verifiers. So, there was an open problem whether such a secure undeniable signature against a multi-verifier attack under reasonable constraints exists or not.

In this paper, we solve this problem: we propose a secure undeniable signature scheme against a multi-verifier attack under a reasonable scenario satisfying the non-detectivity and the dishonest majority of the verifiers. In this scenario, the interface between verifiers are based on the basic protocol between the prover and the verifier, and the relationship among the interactions cannot be traced. Hereafter we call it the *divertible scenario*. Note that the detection of abuses is difficult in this scenario and that this scenario assumes that no verifier is trusted to be honest by the others (or the majority of the verifiers can be dishonest). Therefore, this scenario satisfies the non-detectivity and the dishonest majority of the verifiers.

In order to construct this secure undeniable signature scheme against a multi-verifier attack, we propose a new concept, *secure bit commitment function against divertibility*, and prove that Naor's bit commitment function [N] based on pseudo-random generators is secure against divertibility under the reasonable assumption. Implementation of *divertible ZKIP* using the *secure bit commitment function against divertibility* ensures invulnerability against multi-verifier attacks.

[1] Though [ISS] tried to construct the *divertible zero knowledge interactive proof* for IP, their definition of divertibility was wrong. So the construction of *divertible ZKIP* for IP is still open problem.

In other words, we show the way to convert a *divertible ZKIP* to a *divertible-free-ZKIP* by using a secure bit commitment function against divertibility. Therefore, a divertible ZKIP can be converted to a divertible-free-ZKIP assuming the existence of a one-way function [N, ILL, H]. Then, any negative side of a divertible ZKIP, such as the mafia fraud attack and the multi-verifier attack, can be protected by the usage of a secure bit commitment function against divertibility.

Note that a "non-malleable" bit commitment scheme proposed by Dolev, Dwork and Naor [DDN] can also solve this divertible problem. Although their scheme has broader applications than ours, since their scenario does not require the non-detectivity, their scheme is, however, much less efficient than our solution. Thus there is a tradeoff between the applicable scenario and the efficiency.

2 Problems in Divertible Scenario

First we will explain problems in the divertible scenario in this section. We will define a new security concept in the divertible scenario, and clarify some properties in the next section.

Recently, new digital signature schemes having the following properties were proposed based on *ZKIP* [C, OO90]:

(1) Only signer A can prove the validity of a message to any verifier B by using A's public key or A's identity.

(2) Verifier B cannot prove the validity of the message to another verifier C (non-transitivity).

Though it was hoped that the non-transitivity property was useful in many applications, for example, undeniable signature is suitable to software distribution [C], where only paying customers are able to verify the signature of software supplier with undeniable signature procedure, its weakness was pointed out in [DY, OOF1].

We will explain the problems in the divertible scenario using Chaum's undeniable signature in more detail.

2.1 Undeniable Signature

Although an undeniable signature is similar to a digital signature in that it is a number issued by a signer that is related to the signer's public key and his message, it cannot be verified without the signer's cooperation. In order to check the validity or invalidity of the signature, this scheme consists of two parts, confirmation and disavowal protocols. Hereafter, we will explain only the confirmation protocol.

Center generates a large prime number p and selects a primitive element g of field GF(p) as common information in the system. Signer A generates his secret key x, and computes y ($= g^x \pmod p$). He publishes y as his public key.

Signer A generates signature s ($= m^x \pmod p$) corresponding to a message m from p and his secret key x, and sends (m, s) to verifier B.

Verifier B verifies the validity of signature s for a message m by cooperating with signer A using the following procedures.

Protocol 1 (Confirmation Protocol)

Step 1: Verifier B generates two random integers a and b, calculates

$$X = m^a \cdot g^b \pmod p$$

and sends X with m to signer A.

Step 2: Signer A generates a random integer q, computes

$$\begin{aligned} Y &= X \cdot g^q \pmod p \\ Z &= Y^x \pmod p \end{aligned}$$

and sends (Y, Z) to verifier B.

Step 3: Verifier B sends a and b to signer A.

Step 4: Signer A checks the following equation

$$X \stackrel{?}{=} m^a \cdot g^b \pmod p.$$

If the check succeeds, A sends q to verifier B. If the check fails, the procedure halts.

Step 5: Verifier B checks the following equations

$$\begin{aligned} Y &\stackrel{?}{=} m^a \cdot g^{b+q} \pmod p \\ Z &\stackrel{?}{=} s^a \cdot y^{b+q} \pmod p. \end{aligned}$$

If both checks succeed, B accepts the validity of (m, s). Otherwise, B does not accept the validity of (m, s).

This protocol satisfies non-transitivity, because it is *ZKIP* and its view of communication between a prover and a verifier can be simulated easily. Thus the view is not regarded as evidence of a signature.

2.2 Abuse of Undeniable Signature

We will describe an attack that allows plural verifiers to check the validity of a signature simultaneously, in which if a malicious person takes part as one verifier, the non-transitivity of a signature is suspect.

Suppose software supplier A believes that there is only paying verifier B_1, but unfortunately B_1 is malicious. Since B_1 can convince another verifier B_2 of the software's validity using the following attack, B_1 can be paid by B_2. As

a result, B_1 can use the genuine software without paying his own money. Note that supplier A does not know that there are plural verifiers in this case.

Hereafter, we consider the simple case, where only two verifiers, B_1 and B_2, use the protocol. Two verifiers, B_1 and B_2, can verify the validity of (m, s) in cooperation with signer A using the following procedures in the confirmation protocol. Since A and B_2 act in the same way as **Protocol 1**, we will describe only the procedure of B_1.

Protocol 2

Step 1: Verifier B_1 generates two random integers a and b, calculates

$$X_1 = X_2 \cdot m^a \cdot g^b \pmod{p}$$

and sends X_1 with m to signer A, where X_2 is calculated by B_2 using **step 1** of **Protocol 1**.

Step 2: Verifier B_1 calculates

$$\begin{aligned} Y_2 &= Y_1/(m^a \cdot g^b) \pmod{p} \\ Z_2 &= Z_1/(s^a \cdot y^b) \pmod{p} \end{aligned}$$

and sends (Y_2, Z_2) to verifier B_2, where (Y_1, Z_1) is calculated by A using **step 2** of **Protocol 1**.

Step 3: Verifier B_1 checks the following equation

$$X_2 \stackrel{?}{=} m^{a_2} \cdot g^{b_2} \pmod{p}.$$

If the check succeeds, B_1 sends $a_1 = a_2 + a \pmod{p-1}$, $b_1 = b_2 + b \pmod{p-1}$ to signer A, where (a_2, b_2) are sent by B_2 at **step 3** of **Protocol 1**. If the check fails, the procedure halts.

Step 4: Verifier B_1 checks the following equations receiving q from A

$$\begin{aligned} Y_1 &\stackrel{?}{=} m^{a_1} \cdot g^{b_1+q} \pmod{p} \\ Z_1 &\stackrel{?}{=} s^{a_1} \cdot y^{b_1+q} \pmod{p}. \end{aligned}$$

If both checks succeed, B_1 accepts the validity of (m, s) and sends q to verifier B_2. Otherwise, B_1 does not accept the validity of (m, s) and sends q to B_2.

Remark

Strictly speaking, the above attack is detectable, since the same value, q, is transferred between A-B_1 and B_1-B_2. The typical attacks proposed in [OO89, BD, OOF2] are based on the divertible property, which is not detectable.

2.3 Discussion

Why does the above attack succeed? Because plural verifiers can verify the validity of a signature using the same challenges, X, and the same responses, Y and Z, simultaneously.

Though we will explain the challenge case, a similar discussion is true for in the response case. Denote the generation of X as $X = f(a,b) = m^a \cdot g^b \pmod p$. Then $X_1 = X_2 \cdot f(a,b) = f(a_2,b_2) \cdot f(a,b) = f(a_2+a, b_2+b)$ holds. Therefore, B_1 calculates the value of $X_1 = X_2 \cdot f(a,b)$ $(= f(a_1,b_1))$, where X_2 is issued by B_2 and (a,b) are generated by B_1, without knowing the values of a_1 and b_1 at **Step 1** of **Protocol 2**, and calculates the values of a_1 $(= a_2 + a)$ and b_1 $(= b_2 + b)$ using (a_2, b_2) issued by B_2 at **Step 3**.

The above attack depends on the homomorphism of functions which are used to generate a challenge, X, by a verifier B, and responses, Y and Z, by a prover A. Recently, it was proven that if probabilistic encryption homomorphism exists, then all language in NP have divertible $ZKIP$ based on 'swapping techniques' [BD]. Since a *public* coin type $ZKIP$ is used in their construction, where the committed bit by a verifier is sent publicly. the homomorphism of the bit commitment from a verifier also holds.

Therefore it is important for a protocol designer to provably overcome the homomorphism of either the challenge generating function or the response generating function.

Remark

This attack is applicable even if verifiers cannot trust each other. Since B_1 engages A with a protocol similar to **protocol 1**, B_1 can trust the result of **Step 4** of **protocol 2**. Note that the usage of randomness, a and b, by verifier B_1 is necessary in order to prevent the conspiracy of false A and B_2, because the communication between A and B_2 is simulated easily if B_1 doesn't use randomness (zero-knowledgeness of **protocol 1**).

2.4 Divertible Scenario and Divertible-freeness

Hereafter, we discuss a $ZKIP$ protocol against a multi-verifier attack under a reasonable scenario, which we call the *divertible scenario*, where the interface between verifiers are based on the basic protocol between the prover and the verifier and the relationship among the interactions cannot be traced. Note that the detection of abuses is difficult in this scenario and that this scenario assumes that no verifier is trusted to be honest by the others (or the majority of the verifiers can be dishonest). Therefore, this scenario satisfies the non-detectivity and the dishonest majority of the verifiers.

Moreover, we call a ZKIP *divertible-free* if the ZKIP is secure against multi-verifier attacks in the *divertible scenario*.

3 Secure Bit Commitment in Divertible Scenario

3.1 Definition of Security

We can counter the attack based on homomorphism described in the previous section by using the following concept, *secure bit commitment function against divertibility*, to generate the challenges. Intuitively it satisfies the following property: A committer who doesn't know the value of a and b cannot calculate the value of $f(a,b)$. Remember that B_1 can calculate the value of $f(a_1,b_1)$ without knowing the value of a_1 and b_1 at **Step 1** of **protocol 2** in the above example.

Definition 3.1 (Secure Bit Commitment Function)
Let $\mathcal{Y}(x,b)$ be a bit commitment function, where $b \in \{0,1\}$ is a committed bit and $x \in \{0,1\}^*$ is a random string [N]. *We say that $\mathcal{Y}$ is secure against divertibility, if in the case where $t = O(poly(|\mathcal{Y}(x,b)|)) > 1$, there is no triple of expected polynomial time probabilistic Turing machines $(\mathcal{M},\mathcal{B},\mathcal{X})$ such that, $\mathcal{M}(y_1,\ldots,y_t) = \mathcal{Y}(x^*,b^*)$ holds with non-negligible probability, where $y_i = \mathcal{Y}(x_i,b_i)$ $(i = 1,\ldots,t)$, $b^* = \mathcal{B}(b_1,\ldots,b_t,y_1,\ldots,y_t)$ and $x^* = \mathcal{X}(x_1,\ldots,x_t,y_1,\ldots,y_t)$, and in the case where $t = 1$, for any x $(|x| \leq O(poly(|\mathcal{Y}(x,b)|)))$ there is no triple $(\mathcal{M},\mathcal{B},\mathcal{X})$ such that $\mathcal{M}(y_1,x) = \mathcal{Y}(x^*,b^*)$ holds with non-negligible probability, where $y_1 = \mathcal{Y}(x_1,b_1)$, $b^* = \mathcal{B}(b_1,x,y_1)$ and $x^* = \mathcal{X}(x_1,x,y_1)$. Here $\mathcal{B}$ and $\mathcal{X}$ satisfy the following* **properties** *in* **subsection 3.3**.

Remark
The reason why the function $\mathcal{Y}$ is applied to all (x_i,b_i) is that verifier B_i engages prover A or verifier B_{i-1} with a basic protocol. Note that verifiers, $(B_1,\ldots,B_t)$, cannot trust each other in the divertible scenario.

Definition 3.2 (Secure Bit Commitment Function for multiple bits)
Let $\vec{b}$ be a k-bit string, $b_1||b_2||\ldots||b_k$, where $||$ means concatenation. We call $\vec{\mathcal{Y}}$ a secure bit commitment function for multiple bits if $\vec{\mathcal{Y}}(\vec{b},\vec{x}) = \mathcal{Y}(b_1,x_1)||\cdots||\mathcal{Y}(b_k,x_k)$, where $\mathcal{Y}$ is a secure bit commitment function, and $\vec{x} = x_1||\ldots||x_k$.

3.2 An Application of Secure Bit Commitment Function

Chaum's scheme becomes secure against multi-verifier attack when verifier B sends $w(=\vec{\mathcal{Y}}(a||b,x))$ with (X,m) to signer A at **Step 1**, B sends x with (a,b) to A at **Step 3** and A checks $w \stackrel{?}{=} \vec{\mathcal{Y}}(a||b,x)$ at **Step 4** during the confirmation protocol (**Protocol 1**).

Why is this modification secure in the divertible scenario? Let us consider the following situation: signer A convinces verifier B of the validity of A's signature in the confirmation protocol using the above protocol, and plural verifiers,

B_i $(i = 1, \ldots, t)$, try to share the validity of A's signature through B issuing $y_i = \vec{\mathcal{Y}}(a_i||b_i, x_i)$ to B.

Assume that verifier B, which is a polynomial time probabilistic Turing machine, succeeds the multi-verifier attack, that is, B could commit the value w calculating from $y_1, \ldots, y_t$ at **Step 1** of **Protocol 1**, where we denote this function as $\mathcal{M}$, i.e., $w = \mathcal{M}(y_1, \ldots, y_t)$, and could open the values $a||b$ and x calculating from $a_1||b_1, \ldots, a_t||b_t, x_1, \ldots, x_t$ such that $w = \vec{\mathcal{Y}}(a||b, x)$ at **Step 3** of **protocol 1**, where we denote these functions as $\vec{\mathcal{B}}$ and $\vec{\mathcal{X}}$, i.e., $a||b = \vec{\mathcal{B}}(a_1||b_1, \ldots, a_t||b_t, y_1, \ldots, y_t)$, $x = \vec{\mathcal{X}}(x_1, \ldots, x_t, y_1, \ldots, y_t)$.

Since B is a polynomial time probabilistic Turing machine, the triple $(\mathcal{M}, \vec{\mathcal{B}}, \vec{\mathcal{X}})$ is so. This is a contradiction of the definition of $\vec{\mathcal{Y}}$. Thus the multi-verifier attack fails if we use a secure bit commitment function, $\vec{\mathcal{Y}}$, in this modification.

More generally, we can prove the following theorem.

Theorem 3.3 (Conversion of divertible ZKIP to divertible-free ZKIP)
Let (A, B) be a divertible ZKIP repeating the following procedure:
step 1: *A sends a message x' to B.*
step 2: *B selects a random bit $b \in \{0, 1\}$ and sends it to A.*
step 3: *A sends a message z to B.*
step 4: *B checks whether (x', b, z) satisfies a relation.*

Case a) If we construct $(\widetilde{A}, \widetilde{B})$ using a secure bit commitment function $\mathcal{Y}$ as follows:
step 1a: *$\widetilde{B}$ selects a random bit $b \in \{0, 1\}$ and a random string $x \in \{0, 1\}^*$, calculates $y = \mathcal{Y}(x, b)$ and sends it to $\widetilde{A}$.*
step 2a: *$\widetilde{A}$ sends a message x' to $\widetilde{B}$.*
step 3a: *$\widetilde{B}$ sends the bit b and the string x to $\widetilde{A}$.*
step 4a: *$\widetilde{A}$ checks whether $y = \mathcal{Y}(x, b)$ holds. If the check succeeds, $\widetilde{A}$ sends a message z to $\widetilde{B}$.*
step 5a: *$\widetilde{B}$ checks whether (x', b, z) satisfies a relation.*
Then $(\widetilde{A}, \widetilde{B})$ is a divertible-free ZKIP.

Case b) If we construct $(\hat{A}, \hat{B})$ using a secure bit commitment function $\vec{\mathcal{Y}}$ as follows:
step 1b: *$\hat{A}$ calculates two random values $\vec{z_0}, \vec{z_1}$, where $\vec{z_b}$ corresponds to the message z at step 3 when it receives a bit b at step2, selects two random strings $\vec{x_0}, \vec{x_1} \in \{0, 1\}^*$, calculates $\vec{y_i} = \vec{\mathcal{Y}}(\vec{x_i}, \vec{z_i})(i = 0, 1)$ and sends them with x' to $\hat{B}$.*
step 2b: *$\hat{B}$ sends a bit b to $\hat{A}$.*
step 3b: *$\hat{A}$ sends the message $\vec{z_b}$ and the string $\vec{x_b}$ to $\hat{B}$.*
step 4b: *$\hat{B}$ checks whether $\vec{y_b} = \vec{\mathcal{Y}}(\vec{x_b}, \vec{z_b})$ holds. If the check succeeds, $\hat{B}$ checks whether $(x', b, \vec{z_b})$ satisfies a relation.*
Then $(\hat{A}, \hat{B})$ is a divertible-free ZKIP.

Sketch of Proof:

We will discuss Case a) ($t > 1$) only. The similar discussion holds for Case a) ($t = 1$) and Case b).

Assume that $(\widetilde{A}, \widetilde{B})$ is not *divertible-free*, that is, there exists verifier $\widetilde{B}$ succeeds the multi-verifier attack in the divertible scenario, where signer $\widetilde{A}$ convinces verifier $\widetilde{B}$ of the validity of $\widetilde{A}$'s proof using the above protocol, and plural verifiers, $\widetilde{B}_i$ $(i = 1, \ldots, t)$, try to share the validity of $\widetilde{A}$'s proof through $\widetilde{B}$, where $\widetilde{B}_i$ issues $y_i = \mathcal{Y}(x_i, b_i)$ to $\widetilde{B}$.

$\widetilde{B}$ could commit the value y^* calculating from $y_1, \ldots, y_t$ at **step 1a**, where we denote this function as $\mathcal{M}$, i.e., $y^* = \mathcal{M}(y_1, \ldots, y_t)$, and could open the values b^* and x^* calculating from $b_1, \ldots, b_t$, and $x_1, \ldots, x_t$, such that $y^* = \mathcal{Y}(b^*, x^*)$ at **step 3a**, where we denote these functions as $\mathcal{B}$ and $\mathcal{X}$, i.e., $b^* = \mathcal{B}(b_1, \ldots, b_t, y_1, \ldots, y_t)$ and $x^* = \mathcal{X}(x_1, \ldots, x_t, y_1, \ldots, y_t)$. [2]

Since $\widetilde{B}$ is a polynomial time probabilistic Turing machine, the triple $(\mathcal{M}, \mathcal{B}, \mathcal{X})$ is so. This is a contradiction of the definition of $\mathcal{Y}$.

(Q.E.D. Theorem)

3.3 Properties of Functions

We will clarify some properties of $\mathcal{X}, \mathcal{B}$ introduced in the above definition. The following properties come from the divertible scenario, where verifiers cannot trust each other. That is, verifier B_i is afraid of the conspiracy of false A and other verifiers B_j $(j \neq i)$. So, the values of b^* and x^* should depend on all b_i and x_i components equally.

Properties

1) $\mathcal{B}_i^{b_1, \ldots, b_{(i-1)}, b_{(i+1)}, \ldots, b_t} : b_i \longrightarrow \mathcal{B}(b_1, \ldots, b_{(i-1)}, b_i, b_{(i+1)}, \ldots, b_t)$: bijective for any i $(1 \leq i \leq t)$. We call this the bijective property of $\mathcal{B}$.

2) $\mathcal{X}_i^{x_1, \ldots, x_{(i-1)}, x_{(i+1)}, \ldots, x_t} : x_i \longrightarrow \mathcal{X}(x_1, \ldots, x_{(i-1)}, x_i, x_{(i+1)}, \ldots, x_t)$: bijective for any i $(1 \leq i \leq t)$. We call this the bijective property of $\mathcal{X}$.

Remark

These assumptions imply that $(t-1)$ verifiers can neither guess the hidden bit, b_i, nor control the value b^* in a conspiracy.

4 Naor's Bit Commitment is Secure against Divertibility

We will discuss the security of the bit commitment function based on Naor's idea [N].

---

[2] Note that our framework, where $x^* = \mathcal{X}(x_1, \ldots, x_t, y_1, \ldots, y_t)$ and $b^* = \mathcal{B}(b_1, \ldots, b_t, y_1, \ldots, y_t)$, is slightly restricted than the case, where $x^* = \mathcal{X}((x_1, b_1), \ldots, (x_t, b_t))$ and $b^* = \mathcal{B}((x_1, b_1), \ldots, (x_t, b_t))$, since $y_i = \mathcal{Y}(x_i, b_i)$ $(i = 1, \ldots, t)$ hold. So the security in the latter case is an open problem.

4.1 Naor's Bit Commitment Function

Let $\mathcal{G}$ be a pseudo-random generator, and $\mathcal{G}_i(x)$ be the i-th bit of the output of $\mathcal{G}(x)$.

Commit stage:
step 1: Verifier B selects a random vector $r = (r_1, \ldots, r_{3n})$, where $r_i \in \{0,1\}$ for $1 \leq i \leq 3n$, and sends it to committer A.
step 2: A selects a seed $x \in \{0,1\}^n$ and sends to B the vector $y = (d_1, \ldots, d_{3n})$ where

$$d_i = \begin{cases} \mathcal{G}_i(x) & \text{if } r_i = 0 \\ \mathcal{G}_i(x) \oplus b & \text{if } r_i = 1 \end{cases}$$

and $b \in \{0,1\}$ is the bit A is committed to, where $\oplus$ denotes the exclusive-or operation.

Reveal stage:
A sends x and B verifies that for all $1 \leq i \leq 3n$

$$\begin{aligned} &\text{if } r_i = 0 \quad \text{then } d_i \stackrel{?}{=} \mathcal{G}_i(x) \\ &\text{if } r_i = 1 \quad \text{then } d_i \stackrel{?}{=} \mathcal{G}_i(x) \ominus b \end{aligned}$$

Notation: Hereafter, we denote $y = \mathcal{Y}^{<r>}(x,b) = \mathcal{G}^{(3n)}(x) \oplus br$, where $\mathcal{G}^{(3n)}$ means the first $3n$-bits output of $\mathcal{G}(x)$.

Assumption 4.1 *There exists a pseudo-random generator $\mathcal{G}^{(3n)} : \{0,1\}^n \longrightarrow \{0,1\}^{3n}$ satisfying the following property: for any polynomial time relation $\mathcal{R} : \{0,1\}^n \times \{0,1\}^n \longrightarrow \{0,1\}$, there is no expected polynomial time probabilistic Turing machine $\mathcal{D} : \{0,1\}^{3n} \times \{0,1\}^{3n} \longrightarrow \{0,1\}$ such that $\mathcal{D}(g,g') = \mathcal{R}(x,x')$ holds with non-negligible probability, where $g = \mathcal{G}^{(3n)}(x)$ and $g' = \mathcal{G}^{(3n)}(x')$. We call this assumption the independence of $\mathcal{G}$.*

Remark
Generally PSRGs do not always satisfy this assumption. There are two directions to avoid this assumption, which are open problems:

(1) The construction of PSRG which satisfies the independence of PSRG: Note that it was proven that if there exists a one-way function, then there exists a pseudo-random generator (PSRG) [ILL, H]. So, there might be some construction technique based on a one-way function.

(2) Clarify the condition of $\mathcal{R}$ which is sufficient for the main theorem: The relation $\mathcal{R}$ used in the proof of main theorem is defined using a function $\mathcal{X}$. If we could prove the sufficient condition of $\mathcal{R}$ for the theorem based on the properties of $\mathcal{X}$, the above assumption is avoidable. [3]

[3] We conjecture that a sufficient condition of $\mathcal{R}$ as follows: for any x, $\#\{x' | \mathcal{R}(x,x') = 0\}/\#\{x'\}$ is negligible, where $\#S$ means the number of elements of a set S.

4.2 Main Theorem

Theorem 4.2 *If there exists a pseudo-random generator $\mathcal{G}$, then Naor's bit commitment function is secure against divertibility, assuming the independence of $\mathcal{G}$ and the bijective properties of $\mathcal{B}$ and $\mathcal{X}$, where $t = O(poly(|\mathcal{Y}(x,b)|))$. The random vector r is independently selected for each bit commitment in the case of $t = 1$.*

Sketch of Proof:

We will discuss the case where $t \geq 2$ at first, then discuss the case where $t = 1$.

Case of $t \geq 2$

The proof is by contradiction. Hereafter we prove the situation where the random vector r is fixed. Since a scheme with the independently selected vector r is more intractable than the one with a fixed r, this restriction to r component is not considered essential.

Assume that expected polynomial time probabilistic Turing machines, $\mathcal{M}(y_1, \ldots, y_t)$, $\mathcal{Y}^{<r>}(x_j, b_j)$ $(1 \leq j \leq t)$, $\mathcal{X}(x_1, \ldots, x_t, y_1, \ldots, y_t)$ and $\mathcal{B}(b_1, \ldots, b_t, y_1, \ldots, y_t)$ exist such that $\mathcal{M}(y_1, \ldots, y_t) = \mathcal{Y}^{<r>}(x^*, b^*)$ holds with non-negligible probability(ϵ), where $y_j = \mathcal{Y}^{<r>}(x_j, b_j)$, $x^* = \mathcal{X}(x_1, \ldots, x_t, y_1, \ldots, y_t)$, and $b^* = \mathcal{B}(b_1, \ldots, b_t, y_1, \ldots, y_t)$.

We will consider two cases with regard to the output of $\mathcal{M}$:

Case 1) The output y^* of $\mathcal{M}(y_1, \ldots, y_t)$ is coincident with some y_α or $y_\alpha \oplus r$, where $1 \leq \alpha \leq t$, (hereafter we denote the probability that **Case 1)** occurs under the existence of $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ as δ.) and

Case 2) All other cases than **Case 1)**.

We will construct an expected polynomial time probabilistic algorithm $\mathcal{A}$ which guesses b from the input y $(= \mathcal{G}^{(3n)}(x) \oplus br)$, where b is unknown to $\mathcal{A}$, for **Case 1)** using $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$. This is a contradiction for the difficulty of guessing a committed bit in Naor's protocol, when δ is non-negligible.

For **Case 2)**, we will construct an expected polynomial time probabilistic algorithm $\mathcal{D}$, which, given g and g', decides whether some relation, $\mathcal{R}(x, x')$, holds or not using $\mathcal{M}$, where $\mathcal{R}$ is defined with $\mathcal{X}$, $g = \mathcal{G}^{(3n)}(x)$ and $g' = \mathcal{G}^{(3n)}(x')$. This is a contradiction of the independence of $\mathcal{G}$, when δ is not non-negligible.

Case 1)

At first, we will construct $\mathcal{A}$, whose input is y $(= \mathcal{G}^{(3n)}(x) \oplus br)$ and output is a guessed bit of b, as follows:

[Algorithm $\mathcal{A}$]
For $i = 1$ to t do
Repeat N times

Step 1: Put $y_i = y$.

Step 2: Select $x_j \in \{0,1\}^n$ and $b_j \in \{0,1\}$ randomly ($j \neq i$).

Step 3: Calculate $y_{jb_j} = \mathcal{G}^{(3n)}(x_j) \oplus b_j r$ ($j \neq i$) and

$$y^* = \mathcal{M}(\vec{y}) \quad \text{where} \quad \vec{y} = (y_{1b_1}, \ldots, y_{i-1b_{i-1}}, y_i, y_{i+1b_{i+1}}, \ldots, y_{tb_t}).$$

Step 4: If $\mathcal{M}$ succeeds, that is, there exist $\alpha \in \{1, \ldots, i-1, i+1, \ldots, t\}$ and $\beta \in \{0,1\}$ such that $y^* = y_{\alpha\beta}$, then output b' which satisfies $\mathcal{B}(b_1, \ldots, b_{i-1}, b', b_{i+1}, \ldots, b_t, \vec{y}) = \beta$ as a guessed bit of b such that $y = \mathcal{G}^{(3n)}(x) \oplus br$ and halt. Otherwise go to the next iteration.

End repeat
End do

[**end of Algorithm** $\mathcal{A}$]

Claim 1: Algorithm $\mathcal{A}$ is polynomial time computable, and it can guess the value of bit b such that $y = \mathcal{G}^{(3n)}(x) \oplus br$ with probability significantly better than $\frac{1}{2}$.

(Proof of Claim 1)

When there exist $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$, $y^* = \mathcal{G}^{(3n)}(x^*) \oplus \mathcal{B}(b_1, \ldots, b_{i-1}, b, b_{i+1}, \ldots, b_t, \vec{y})r$ always holds from their definitions, where $y = \mathcal{G}^{(3n)}(x_i) \oplus br$ and $x^* = \mathcal{X}(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_t, \vec{y})$.

Generally, the probability that x and x' ($x \neq x'$) exist and satisfy $\mathcal{G}^{(3n)}(x') - \mathcal{G}^{(3n)}(x) \oplus r$ is at most 2^{-n}, since $r \in \{0,1\}^{3n}$ is selected randomly, and both x and x' are elements of $\{0,1\}^n$. On the other hand, $\mathcal{B}(b_1, \ldots, b_{i-1}, 0, b_{i+1}, \ldots, b_t, \vec{y}) \neq \mathcal{B}(b_1, \ldots, b_{i-1}, 1, b_{i+1}, \ldots, b_t, \vec{y})$ holds (**Property 1**)). So the probability that $y^* = y_{\alpha 0} = y_{\alpha' 1}$ is at most 2^{-n}. Therefore, if $y^* = y_{\alpha\beta}$ holds, then $x^* = x_\alpha$ and b satisfies $\mathcal{B}(b_1, \ldots, b_{i-1}, b, b_{i+1}, \ldots, b_t, \vec{y}) = \beta$ with overwhelming probability ($\geq 1 - \frac{1}{2^n}$). Let us call the case, where $\mathcal{M}$ finds $y^* = y_{\alpha\beta}$ including the case where $\alpha = i$ at **Step 4**, **Case 1**), and denote the probability that **Case 1**) occurs under the existence of $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ as δ. In the case where $\alpha = i$, since we don't know the value of β, **Step 4** does not decide the value of b. Since the probability that $\alpha = i$ holds at **Step 4** is $\frac{1}{t}$, so the probability $\mathcal{A}$ outputs b is $(1 - \frac{1}{t})\delta\epsilon$. Note that there is an error probability at most $\frac{1}{2^n}$ in the output of b at **Step 4**.

Since we assume that $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ exist with non-negligible probability(ϵ), the algorithm $\mathcal{A}$ can guess the value of bit b such that $y = \mathcal{G}^{(3n)}(x) \oplus br$ with success probability significantly better than $\frac{1}{2}$ if δ is non-negligible. This is because for all sufficiently large n,

$$\begin{aligned}
Prob[\mathcal{A}(y) &= b] \\
&= 1 - Prob[\mathcal{A} \text{ fails at all round}] \\
&= 1 - \{Prob[\mathcal{A} \text{ fails at round } i]\}^t \\
&> 1 - \{(1-\epsilon) + (1-\delta)\epsilon + \tfrac{1}{t}\delta\epsilon + \tfrac{1}{2^n}(1 - \tfrac{1}{t})\delta\epsilon\}^{Nt} \\
&> 1 - \{1 - (1 - \tfrac{2}{t})\delta\epsilon\}^{Nt} \\
&\to 1 - e^{-1} \qquad (N = \tfrac{1}{(t-2)\delta\epsilon}),
\end{aligned}$$

where N is selected as $O(\frac{1}{(t-2)\delta\epsilon})$ such that this probability is significantly better than $\frac{1}{2}$.

Note that $\mathcal{A}(y)$ is polynomial time computable, since $t = O(poly(|\mathcal{Y}(x,b)|))$ and $N = O(\frac{1}{(t-2)\delta\epsilon})$, where both ϵ and δ are non-negligible probabilities.

(q.e.d. Claim 1)

Remark

1) If the bijective property of $\mathcal{B}$ does not hold, then it might hold that $\mathcal{B}(b_1, \ldots, b_{i-1}, 0, b_{i+1}, \ldots, b_t, \vec{y}) = \mathcal{B}(b_1, \ldots, b_{i-1}, 1, b_{i+1}, \ldots, b_t, \vec{y}) = \beta$ or $\overline{\beta}$. Here the value of b is not decided uniquely or there is no value of b that satisfies this relation.

2) We can not guess the value of b given in both **Case 2)** and $\alpha = i$, where $y^* \neq y_{\alpha\beta}$, that is, $x^* \neq x_\alpha$ ($\alpha \in \{1, \ldots, i-1, i+1, \ldots, t\}$ in each round i). Thus, the following discussion is necessary for **Case 2)**, where δ is not non-negligible.

Case 2)

Let denote $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*) = 0$ iff $\mathcal{X}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_t, \vec{y}) = x^*$ holds, where $\vec{y} = (y_{1b_1}, \ldots, y_{i-1b_{i-1}}, y_{ib_i}, y_{i+1b_{i+1}}, \ldots, y_{tb_t})$.

Next we will construct $\mathcal{D}$ and the relation $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}$, on input of g ($= \mathcal{G}^{(3n)}(x)$) and g^* ($= \mathcal{G}^{(3n)}(x^*)$), where x and x^* are not known to $\mathcal{D}$, as follows:

[Algorithm $\mathcal{D}$]

For $i = 1$ to t do

Repeat N times

Step 1: Put $y_i = g$.

Step 2: Select $b_i \in \{0,1\}$, $x_j \in \{0,1\}^n$ and $b_j \in \{0,1\}$ randomly ($j \neq i$).

Step 3: Calculate $y_{ib_i} = y_i \oplus b_i$, $y_{jb_j} = \mathcal{G}^{(3n)}(x_j) \oplus b_j r$ ($j \neq i$) and

$$y^* = \mathcal{M}(\vec{y}) \quad \text{where} \quad \vec{y} = (y_{1b_1}, \ldots, y_{i-1b_{i-1}}, y_{ib_i}, y_{i+1b_{i+1}}, \ldots, y_{tb_t}).$$

Step 4: If the following equation holds, then output 0. Otherwise output 1.

$$y^* = g^* \oplus \mathcal{B}(b_1, \ldots, b_{i-1}, b_i, b_{i+1}, \ldots, b_t, \vec{y})r$$

End repeat

End do

[end of Algorithm $\mathcal{D}$]

We will prove that $\mathcal{D}(g, g^*) = \mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*)$ holds with non-negligible probability, where g ($= \mathcal{G}^{(3n)}(x)$) and g^* ($= \mathcal{G}^{(3n)}(x^*)$).

Claim 2: If there exist $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$, $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x') = 0$ implies $\mathcal{D}(g, g') = 0$, where $g = \mathcal{G}^{(3n)}(x)$ and $g' = \mathcal{G}^{(3n)}(x')$.

(Proof of Claim 2)

By the definition of $\mathcal{R}$, $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x') = 0$ means $x' = \mathcal{X}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_t, \vec{y})$. The existence of $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ implies that

$$y^* = \mathcal{G}^{(3n)}(x') \oplus \mathcal{B}(b_1, \ldots, b_t, \vec{y})r.$$

Since **Step 4** of algorithm $\mathcal{D}$ holds, $\mathcal{D}(g, g') = 0$ holds, where $g = \mathcal{G}^{(3n)}(x)$ and $g' = \mathcal{G}^{(3n)}(x')$.

(q.e.d. Claim 2)

Claim 3: If there exist $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$, $\mathcal{D}(g, g^*) = 0$ implies $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*) = 0$, where $g = \mathcal{G}^{(3n)}(x)$ and $g^* = \mathcal{G}^{(3n)}(x^*)$ with overwhelming probability.

(Proof of Claim 3)

By the definition of $\mathcal{D}$, $\mathcal{D}(g, g^*) = 0$ means

$$y^* = g^* \oplus \mathcal{B}(b_1, \ldots, b_{i-1}, b_i, b_{i+1}, \ldots, b_t, \vec{y})r.$$

The existense of $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ implies that

$$y^* = \mathcal{G}^{(3n)}(x') \oplus \mathcal{B}(b_1, \ldots, b_t, \vec{y})r,$$

where x is defined as $g = \mathcal{G}^{(3n)}(x)$ and $x' = \mathcal{X}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_t, \vec{y})$.

These equations imply $g^* = \mathcal{G}^{(3n)}(x')$ with overwhelming probability. The reason is as follows: Generally, the probability that x and x' $(x \neq x')$ satisfy $\mathcal{G}^{(3n)}(x') = \mathcal{G}^{(3n)}(x)$ is at most 2^{-3n}, since pseudo-random generators pass the next bit test [Y].

Since x^* satisfies $g^* = \mathcal{G}^{(3n)}(x^*)$, the probability that $x^* \neq x'$ holds is negligible $(\leq \frac{1}{2^{3n}})$. Therefore, $x^* = x' = \mathcal{X}(x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_t, \vec{y})$, that is, $\mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*) = 0$ with overwhelming probability $(\geq 1 - \frac{1}{2^{3n}})$.

(q.e.d. Claim 3)

It is clear that if there exist $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$, then $\mathcal{D}(g, g^*) = \mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*)$ holds by combining **Claim 2** and **Claim 3** with overwhelming probability (γ). Since we assume that $(\mathcal{M}, \mathcal{Y}, \mathcal{X}, \mathcal{B})$ exist with non-negligible probability (ϵ) and **Case 2)** occurs with probability $(1-\delta)$, $\mathcal{D}(g_1, g^*) = \mathcal{R}_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_t,\vec{y}}(x, x^*)$ holds with non-negligible probability $(\gamma\epsilon(1-\delta))$, where δ is not non-negligible. This is a contradiction of the independence of $\mathcal{G}$.

Case of $t = 1$

Finally we will prove the case where $t = 1$. The proof is also by contradiction. Differently from the case that $t > 1$, in this case we assume an additional condition that random vector r for Naor's bit commitment scheme is independently selected for each bit commitment.

Assume that there exists $x_0 \in \{0,1\}^*$ and expected polynomial time probabilistic Turing machines, $\mathcal{M}$, $\mathcal{X}$ and $\mathcal{B}$, such that $\mathcal{M}(y_1, x_0) = \mathcal{Y}^{<r^*>}(x^*, b^*)$ holds with non-negligible probability(ϵ), where $y_1 = \mathcal{Y}^{<r_1>}(x_1, b_1)$, $x^* = \mathcal{X}(x_1, x_0, y_1)$ and $b^* = \mathcal{B}(b_1, x_0, y_1)$.

We will also consider two cases with regard to the output of $\mathcal{M}$:

Case 1) The output y^* of $\mathcal{M}(y_1, x_0)$ is coincident with y_1 or $y_1 \oplus r^*$, and

Case 2) All other cases than **Case 1)**.

We can construct an expected polynomial time probabilistic algorithm $\mathcal{A}'$ which guesses b from input y $(= \mathcal{G}^{(3n)}(x) \oplus br)$. The algorithm $\mathcal{A}'$ is as follows:

Step 1: Determine x_0 and calculate $y^* = \mathcal{M}(y, x_0)$.

Step 2: Determine b^* by checking whether y^* is coincident with y or $y \oplus r^*$. Then determine b by using $b^* = \mathcal{B}(b, x_0, y^*)$.

Similarly to Claim 1, we can show that algorithm $\mathcal{A}'$ can guess b correctly with non-negligible probability. This is a contradiction for the difficulty of guessing a committed bit in Naor's protocol.

For **Case 2)**, we can construct an expected polynomial time probabilistic algorithm $\mathcal{D}'$, which, given g and g', decides whether some relation, $\mathcal{R}(x, x')$, holds or not using $\mathcal{M}$, where $\mathcal{R}$ is defined with $\mathcal{X}$, $g = \mathcal{G}^{(3n)}(x)$ and $g' = \mathcal{G}^{(3n)}(x')$. Algorithm $\mathcal{D}'$ can be constructed in a manner similar to algorithm $\mathcal{D}$, and it can be shown similarly that $\mathcal{D}'$ works correctly with non-negligible probability. Hence, this is a contradiction of the independence of $\mathcal{G}$.

These results can be easily extended to any multiple bit case ($k > 1$), because if a multiple bit relation $\mathcal{B}$ holds such that $\vec{b}^* = \mathcal{B}(\vec{b}_1, \vec{b}_2, \vec{y}_1, \vec{y}_2)$, then the first elements of $\vec{b}^*$, $\vec{b}_1$ and $\vec{b}_2$ satisfy a relation with the parameters of the remaining elements of $\vec{b}^*$, $\vec{b}_1$, $\vec{b}_2$, $\vec{y}_1$ and $\vec{y}_2$. Therefore, the result that there is no relation among single bit variables implies that there is no relation among multiple bit variables.

(Q.E.D. Theorem)

5 Conclusion and Remarks

This paper has proposed a new security concept, the *secure bit commitment function against divertibility*. We has shown that Naor's bit commitment function

based on a pseudo-random generator satisfies this property under the independence of PSRG. Implementation of *divertible ZKIP* using the *secure bit commitment function against divertibility* ensures invulnerability against multi-verifier attacks in the *divertible scenario* where the *non-detectivity* and the *dishonest majority* of the verifiers are satisfied. Thus, any negative attributes of a divertible ZKIP, such as the mafia fraud attack and the multi-verifier attack, can be removed by using the secure bit commitment function against divertibility.

Note that this conversion from a *divertible ZKIP* to a *divertible-free-ZKIP* by using the secure bit commitment function against divertibility is also effective against the *meddler attack* described in [DY] since the intermediate node uses the homomorphic property of challenge generating function.

However, there are several open problems:

(1) Security in a more general situation, e.g. where $x^* = \mathcal{X}((x_1, b_1), \ldots, (x_t, b_t))$, and $b^* = \mathcal{B}((x_1, b_1), \ldots, (x_t, b_t))$.

(2) The construction of PSRG which satisfies the independence of PSRG.

(3) Clarify the condition of $\mathcal{R}$ which is sufficient for the main theorem.

Acknowledgments

This research was conducted while the first author was visiting the MIT Laboratory for Computer Science. He would like to acknowledge the generous support provided by MIT. The authors would like to thank anonymous referees for useful comments on our preliminary manuscript.

References

[BD] M.Burmester and Y.Desmedt, "All Languages in NP Have Divertible Zero-Knowledge Proofs and Arguments under Cryptographic Assumptions," EUROCRYPT'90

[C] D.Chaum, "Zero-Knowledge Undeniable Signatures," EUROCRYPT'90

[D] Y.Desmedt, "Subliminal-Free Authentication and Signature," EUROCRYPT'88

[DDN] D.Dolev, C.Dwork and M.Naor, "Non-Malleable Cryptography," STOC'91

[DGB] Y.Desmedt, C.Goutier and S.Bengio, "Special Uses and Abuses of the Fiat-Shamir Passport Protocol," CRYPTO'87

[DY] Y.Desmedt and M.Yung, "Weaknesses of Undeniable Signature Schemes," EUROCRYPT'91

[FS] A.Fiat and A.Shamir, "How to Prove Yourself," CRYPTO'86

[GMR] S.Goldwasser, S.Micali and C.Rackoff, "The Knowledge Complexity of Interactive Proof Systems," STOC'85

[GMW] O.Goldreich, S.Micali and A.Wigderson, "Proofs that Yield Nothing But their Validity and a Methodology of Cryptographic Protocol Design," FOCS'86

[H] J.Håstad, "Pseudo-Random Generators under Uniform Assumptions," STOC'90

[ILL] R.Impagliazzo, L.Levin and M.Ludy, "Pseudo-random generation from one-way functions," STOC'89

[ISS] T.Itoh, K.Sakurai and H.Shizuya, "Any Language in IP has a Divertible ZKIP," ASIACRYPTO'91

[N] M. Naor, "Bit Commitment Using Pseudo-Randomness," CRYPTO'89

[OO89] T.Okamoto and K.Ohta, "Divertible Zero-Knowledge Interactive Proofs and Commutative Random Self-Reducible," EUROCRYPT'89

[OO90] T.Okamoto and K.Ohta, "How to Utilize the Randomness of Zero-Knowledge Proofs," CRYPTO'90

[OOF1] K.Ohta, T.Okamoto and A.Fujioka, "Abuses of Undeniable Signature and Their Countermeasures," IEICE Transactions, Vol. E-74, No. 8, pp. 2109–2113, 1991

[OOF2] K.Ohta, T.Okamoto and A.Fujioka, "Multi-Verifier Digital Signature Scheme," (in Japanese) Japanese Patent, File No. Toku-gan-hei 3-24856 (Feb. 19, 1991)

[Y] A.Yao, "Theory and Applications of Trapdoor Functions," FOCS'82