Additive Proofs of Knowledge - A New Notion For Non-Interactive Proofs

Amitabh Saxena
Dept. of Computer Science and Computer Engineering
La Trobe University, Bundoora, VIC, Australia 3086

February 23, 2007

Abstract

In this paper, we study the opacity property of *verifiably encrypted signatures* (VES) of Boneh et al. (proposed in Eurocrypt 2003). Informally, opacity implies that although some given aggregate signatures can verified, no useful information about the individual signatures is leaked. However, the very fact that an aggregate signature can be verified leaks certain information - that the individual signature is indeed well-formed. Apart from this, is there any other information leaked? In this paper, we show that there is *absolutely no other information leaked* about the individual signatures when the aggregation contains only two signatures. In more formal terms, we show that VES are *Zero-Knowledge* (ZK). We then extend the ZK property of VES to propose efficient Additive Non-Interactive Witness-Indistinguishable (A-NIWI) proofs. Intuitively an A-NIWI proof can be considered as a Proof of Knowledge (PoK) of another A-NIWI proof.

1 Introduction

In this work, we propose a new construction of Non-Interactive Witness Indistinguishable (NIWI) proofs (of Knowledge), namely Additive-Non-Interactive Witness Indistinguishable (A-NIWI) proofs - a given NIWI proof π_1 of statement m_1 can be combined with another NIWI proof π_2 of m_2 to yield a new NIWI proof $\pi_{1,2}$ of $m_1 \wedge m_2$ such that that given $\pi_{1,2}, m_1, m_2$, it is no longer possible to obtain π_1 (or π_2). We term this property *additiveness* and formally give a construction of a NIWI proof that satisfies this property. We call any NIWI proof system satisfying this property an Additive NIWI (A-NIWI) proof system. Although this property can be achieved using conventional constructions of NIZK proofs (i.e., by reducing it to some NP-complete language and then creating a NIZK of the NP-complete language), such constructions are extremely inefficient and therefore useless in practice. Furthermore, these generic NIZK constructions are only defined for proofs of membership, while A-NIWI proofs are defined using proofs of knowledge. Our constructions are based on the opacity property of Verifiably Encrypted Signatures (VES) [1] and the truncation resilience property of chain signatures [2].

The rest of this paper is organized as follows. In Section 2, we give some background on zero-knowledge. In Section 3, we give the intuition behind our idea of additive zero-knowledge by showing that aggregate signatures are zero-knowledge. We then formally define NIZK-PoKs (actually NIZK-PoDPs) and give a construction of a NIZK-PoK for the solution of a computational Diffie-Hellman (CDH) problem instance in Section 4. Finally, in Section 5, we present our examples of additive NIWI proofs.

2 Preliminaries

Zero-Knowledge Proofs. Zero-knowledge proofs (introduced by Goldwasser, Micali and Rackoff [3]) are proofs which convince a verifier that a given statement is indeed true without giving any information as to *why* it is true. This concept can be intuitively captured by saying that whatever

the verifier knows after seeing the proof was already known to the verifier before seeing the proof. The authors of [3] formalized this concept by requiring that there exist a PPT simulator outputting a transcript that is identical to the transcript produced by the real prover. Zero-knowledge is captured by the fact that the simulator generates the identical transcript without knowing the prover's secret. Therefore, there must not be any knowledge "leaked" about the secret. In fact, the entire notion of zero-knowledge proofs can be summarized as follows: *To prove that something is zero-knowledge, simply exhibit a simulator that generates a transcript that cannot be distinguished from the real thing* [4].

Witness Indistinguishability (WI). Another intuitive way to restrict knowledge leakage is using *witness indistinguishable* proofs [5, 6]. However, unlike ZK proofs, a WI proof cannot be simulated. Therefore, there is certainly some information leaked (if it cannot be simulated then more than zero knowledge has leaked). Informally, a WI proof can be defined as follows. Let $x \in L$ for some $L \in$ NP such that x has two witness for L. A proof is WI if it convinces a verifier that indeed $x \in L$ but does not reveal which witness was used to construct the proof (even if the verifier knows both witnesses). In the literature, WI proofs are generally used to prove a statement like $x_1 \in L_1 \vee x_2 \in L_2$ without revealing which witness (of $x_1 \in L_1$ or $x_2 \in L_2$) was used to construct the proof. In this work we focus on WI proofs of statements of the type $x_1 \in L_1 \wedge x_2 \in L_2$ such that it is infeasible to decide if the proof was constructed from the individual witnesses of $x_1 \in L_1$ and $x_2 \in L_2$ or from another WI proof of $x_1 \in L_1 \wedge x_2 \in L_2$.

Proofs Of Knowledge (PoKs). Till now we restricted ourselves to proofs of statements of the type $x \in L$ for some $L \in$ NP. These are called *proofs of membership* (PoMs). However, a more useful notion is of proofs of statements of the type *I know the witness of* $x \in L$. That is, the prover not only proves that $x \in L$ but also proves *knowledge* of a witness to the fact. Such proofs are called *proofs of knowledge* (PoKs) and are formally defined in [7]. Informally, a PoK requires that there be a *knowledge extractor* that uses the prover in a black-box manner and extracts the witness for the statement to be proved [7]. However, this general definition of PoKs cannot be zero-knowledge, even if a simulator exists because of the simple fact that we are no longer trying to prove that $x \in L$ but just knowledge of a witness to $x \in L$. Unfortunately, along the way we also reveal that $x \in L$.

Proofs Of Decision Power (PoDPs). Let $L \in$ NP $\cap$ co-NP. A zero-knowledge (or WI) *proof of decision power* (PoDP) is a PoK for some $x \in L \cup$ co-L that convinces a verifier about the knowledge of a witness for x but does not reveal whether $x \in L$ or $x \in$ co-L. Thus, in effect the proof is proof-of-knowledge of the-ability-to-decide-membership, rather than a proof-of-knowledge-of-membership. See [8, 9] for a discussion on this concept. PoDPs are more powerful than PoKs because they reveal even less information (the verifier is convinced of the knowledge of a witness but still cannot decide membership). All our proofs presented in this paper (whether WI or ZK) will be PoDPs. Consequently, we only focus on the class NP $\cap$ co-NP.

Non-Interactive (NI) ZK and WI Proofs. Zero-knowledge (and WI) proofs come in two flavors: *interactive* and *non-interactive* (NI). In the interactive variants, there are many exchanges of messages (called rounds) before the proof is completed. On the other hand, in the non-interactive variants, the verifier's role is played by a hash function or some other random source of information (such as a random oracle) [10, 11, 4, 12]. Depending on whether the proof is ZK of WI, we call it a NIZK or NIWI proof. Similar to interactive proofs, NI proofs can also be classified as PoMs or PoKs. In this work, we only focus on NIZK-PoKs and NIWI-PoKs. NIZK-PoKs have many applications.[1] Note that WI proofs with two or less rounds are generally called *zaps* [6]. However, we will use the term NIWI to specifically denote that the zap is non-interactive.

Additive NIWI Proofs. Suppose given some NIWI proof π_1 of $x_1 \in L_2$, we can "add" to it another NIWI proof π_2 of $x_2 \in L_2$ to obtain a NIWI proof $\pi_{(1,2)}$ of $x_1 \in L_1 \wedge x_2 \in L_2$, then we call

[1] For instance, in constructing CCA2 secure schemes [11]. The idea is that decryption queries on a ciphertext are only answered if the adversary can prove (using a NIZK-PoK) the knowledge of the corresponding plaintext. Hence decryption queries do not help the adversary.

the NIWI proof system *additive*. The WI property of NIWI proofs implicitly implies that it is no longer possible to extract the proof π_1 just given π_2, x_1, x_2. Therefore, although π_2 is convincing of the truth of both statements $\{x_1, x_2\}$, it cannot be used to prove any one of the statements separately from the other. Additive NIWI proofs arise naturally from the aggregate signatures of [1] as described next. In the above example we considered PoMs. However, our real examples will be based on PoKs. The following discussion is intended to give an idea of this.

3 Aggregate Signatures Are Zero-Knowledge

The aggregate signatures of [1] can be briefly described (with some simplifications) as follows. Let G_1 and G_2 be two cyclic multiplicative groups both of prime order q such that computing discrete logarithms in G_1 and G_2 is intractable. A bilinear pairing is a map $\hat{e} : G_1 \times G_1 \mapsto G_2$ that satisfies the following properties [13, 14, 1].

1. *Bilinearity*: $\hat{e}(a^x, b^y) = \hat{e}(a, b)^{xy}$ $\forall a, b \in G_1$ and $x, y \in \mathbb{Z}_q$.
2. *Non-degeneracy*: If g is a generator of G_1 then $\hat{e}(g, g)$ is a generator of G_2.
3. *Computability*: The map $\hat{e}$ is efficiently computable.

For the rest of this paper we will assume that $g \in G_1$ is some fixed generator and all problem instances are with respect to g. For completeness, we define the CDH problem below.

Definition 3.1. ***Computational Diffie-Hellman (CDH) problem*** *Given* $(X, Y) \in {G_1}^2$*, compute the value* $Z \in G_1$ *satisfying* $\hat{e}(X, Y) = \hat{e}(Z, g)$*.*

The aggregate signature scheme also uses a hash function $\mathcal{H} : \{0,1\} \mapsto G_1$. Let the public keys of two users be $X_1 = g^{x_1}, X_2 = g^{x_2}$ respectively. Let the hashes of the messages to be signed be $Y_1 = g^{y_1}$ and $Y_1 = g^{y_2}$ respectively (for unknown y_1, y_2). Then the the aggregate signature of [1] under public keys Y_1, Y_2 corresponds to the value $Z_2 = g^{x_1 y_1 + x_2 y_2}$. Additionally, the corresponding individual signature under the public key $X_1 = g^{x_1}$ turns out to be $g^{x_1 y_1}$, the extraction of which will correspond to the solution of the CDH instance $(X_1, Y_1) = (g^{x_1}, g^{y_1})$. Call this the signature extraction problem for the tuple $(X_1, Y_1, X_2, Y_2, Z_2)$. Without the extra inputs X_2, Y_2, Z_2, this reduces to the ordinary CDH problem for (X_1, Y_1). We will prove that these extra inputs give absolutely no information about the solution of the corresponding CDH instance (X_1, Y_1).

Observe that given just the CDH instance $(X_1, Y_1) = (g^{x_1}, g^{y_1})$, we can straightaway transform it into an instance of the signature extraction problem without knowing either x_1 or y_1 as follows. Generate two random integers r, u. Then compute $X_2 = X_1 \cdot g^r = g^{r+x_1}$, $Y_2 = g^u / Y_1 = g^{u-y_1}$, $Z_2 = {X_1}^u \cdot {Y_2}^r = g^{x_1 u + r u - r y_1}$. The tuple $(X_1, Y_1, X_2, Y_2, Z_2)$ forms a valid instance of the signature extraction problem.[2] In other words, the aggregate signature leaks absolutely no knowledge about the individual signature!

How is a verifier convinced if there is no knowledge transferred through a signature? The catch here is that in real signature schemes (such as aggregate signatures), we do not have the freedom to choose Y_2 due to the one-way-ness of the hash function. It is proved in [1, Theorem 3.2] that as long as the hash function is indistinguishable from a random oracle, and $Y_1 \neq Y_2$, the aggregate signature scheme is secure against existential forgery. This motivates some interesting applications:

Zero-Knowledge Signatures. Let Alice have the public key X_1 in the above discussion. To sign a message m_1 with resulting hash Y_1, Alice first generates a random public key X_2 (possibly with $X_2 = X_1$) and a message $m_2 \neq m_1$. Let Y_2 be the hash of m_2. Alice computes Z_2 as above and sends $(X_1, m_1, X_2, m_2, Z_2)$ as her signature on m_1. Any verifier can verify that Z_2 is indeed Alice's signature on m_1 by first computing Y_1 and Y_2. The signature is definitely convincing because with a high probability $Y_2 \neq Y_1$, and so the result of [1, Theorem 3.2] stands. However, the signature is also zero knowledge due to the above argument.

[2] This was proved in [15]. Note that it is possible to keep $r = 0$ but we need $X_1 \neq X_2$ for later use.

Suppose Alice is a government official authorized to issue electronic identity cards. The cards contain two pieces of information: (1) The age, and (2) The state of residence. Bob is a person who obtained a card from Alice stating his age to be a and state to be s. Bob needs to enter a club with condition of entry ($age \geq a \wedge state \neq s$). If Alice uses the above technique for signing cards (i.e., by keeping m_1 = "Bob's age is a", m_2 = "Bob's state is s", $X_1 = X_2$, and handing Z_2 to Bob), she can be sure that computing her signature on the statement m_1 given her signature on the statement $m_1 \wedge m_2$ is as hard as computing the signature on m_1 given nothing. On the other hand, Bob must separate the signature if he wants to enter the club using the card issued by Alice, which he cannot do. Of course, the same thing can be achieved if Alice signs the message $m_1 \wedge m_2$ using any standard signature scheme. However, consider the case when the signatures on m_1 and m_2 are computed by different officials (as is common).

Identification. Finally consider the need of Bob when using this card. Bob would like to identify himself with this card and at the same time ensure that the verifier cannot impersonate him later. This can be done as follows. Suppose Bob is identifying himself to Carol using Alice's card. First both Bob and Carol agree on a common random string $Y_3 \in G_1$. It is necessary for both Bob and Carol to ensure that the string is indeed random. Next Bob generates a random private key $x_3 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes the public key $X_3 = g^{x_3}$. Finally he computes the value $Z_3 = Z_2 \cdot Y_3^{x_3} \in G_1$ and gives (Z_3, X_3) to Carol. We prove in the next section that Carol is convinced about Bob's identity but cannot get any useful information about Z_2 from (Z_3, X_3).

4 NIZK Proofs of Knowledge

We now give a formal discussion of the above zero-knowledge property. Our examples deal with NIZK-PoKs (see Section 2). We will use the common-random-string (CRS) model [10]. In this model, both prover and verifier share a common random string that is decided beforehand. It could even be generated by a random oracle. The quality of a NIZK proof is determined by the length of the random string it uses [16]. In our NIZK proofs, the lengths of the statement and the random string are the same.

Let $L \in$ NP $\cap$ co-NP be some language and let $\mathcal{W}(.)$ be an oracle that on input $x \in \Sigma^*$ outputs the witness to either $x \in L$ or $x \notin L$. Define the following protocol between prover P and verifier V. Let k be a security parameter.

Protocol (P, V)

1. **Common Input:** Some string $x \stackrel{R}{\leftarrow} \{0,1\}^k$ is given as common input to both P and V.
2. **Prover's Auxiliary Input:** P is given as auxiliary input $w \leftarrow \mathcal{W}(x)$.
3. **Common Random String:** P and V agree on a common random string (crs) $r \stackrel{R}{\leftarrow} \{0,1\}^k$.
4. **Proof Generation:** P uses (r, w, x) to compute and outputs a proof π.
5. **Proof Verification:** V uses a deterministic procedure on input (x, r, π) and outputs either 0 or 1.

Definition 4.1. *(P, V) is a NIZK-PoK (and a PoDP) for some $L \in NP \cap$ co-NP if the following hold.*

1. *Completeness*: For all $x \in \Sigma^*$ and all honest provers P

$$Pr\Big[r, x \stackrel{R}{\leftarrow} \Sigma^*; w \leftarrow \mathcal{W}(x); \pi \leftarrow P(x, w, r) : V(x, r, \pi) = 1\Big] = 1 \tag{1}$$

2. *Zero-Knowledge*: There exists a universal PPT simulator machine M that on input some random string x (the problem instance) outputs a tuple (r_m, π_m) such that $V(x, r_m, \pi_m) = 1$ and the two distributions $\{\mathcal{X}\}$ and $\{\mathcal{X}\}_m$ defined below are statistically indistinguishable.

$$\{\mathcal{X}\}_m \stackrel{\text{def}}{=} \{r_m, x, \pi_m\} \stackrel{\text{def}}{=} \Big[x \stackrel{R}{\leftarrow} \Sigma^*; (r_m, \pi_m) \leftarrow S : V(x, r_m, \pi_m) = 1\Big]$$

$$\{\mathcal{X}\} \stackrel{\text{def}}{=} \{r, x, \pi\} \stackrel{\text{def}}{=} \left[r, x \stackrel{R}{\leftarrow} \Sigma^*; w \leftarrow \mathcal{W}(x); \pi \leftarrow P(x, w, r) : V(x, r, \pi) = 1\right]$$

3. *Proof-of-Knowledge*: There exists a universal PPT (knowledge) extractor machine E that functions as follows. E gives a "random looking" string r_e to the prover P^*, who outputs a pair (x, π). If $V(x, r_e, \pi) = 1$ then E takes in as input (x, r_e, π) and outputs a string w_e. We require that for all P^*, the strings r_e are indistinguishable from truly random strings, and

$$\Pr\left[\begin{array}{c|l} w_e = w & \begin{array}{l} r_e \leftarrow E(x), (x, \pi) \leftarrow P^*(r_e), V(x, r_e, \pi) = 1, \\ w_e \leftarrow E(x, r_e, \pi), \quad w \leftarrow \mathcal{W}(x) \end{array} \end{array}\right] \approx 1 \tag{2}$$

4.1 NIZK-PoK For A Diffie-Hellman Solution

Let $\hat{e} : G_1 \times G_1 \mapsto G_2$ be a bilinear map as defined in Section 3 such that $|G_1| = |G_2| = q$ (prime). Assume that the computational Diffie-Hellman (CDH) problem is hard in G_1. Therefore, due to the Goldreich-Levin Theorem [17], there must exist a hard-core predicate (say $\delta()$) for the solution of the CDH instance. Let g be some fixed generator of G_1 (this can be fixed once-and-for-all by a trusted third party). Consider the language consisting of pairs of the form $(g^x, g^y) \in G^2$:

$$L = \{(g^x, g^y) | \text{hard-core predicate } \delta(g^{xy}) = 1\}$$

Clearly, $L \in$ NP $\cap$ co-NP and the element g^{xy}, the solution to the CDH instance (g^x, g^y) forms the witness to both the "yes" and "no" instances. We describe a NIZK-PoK for knowledge of this witness. Our non-interactive PoKs can be considered as stating the following: "*Someone* knows the witness to this NP statement". Additionally, our PoKs are actually proofs of decision power (PoDP). That is, a prover proves knowledge of the witness $w = g^{xy}$ to some CCDH instance (g^x, g^y) without revealing whether $(g^x, g^y) \in L$ or not. First we define the following problem.

Definition 4.2. ***Class-CDH (CCDH) problem.*** *Given $X, Y \in G_1$, output 1 if $(X, Y) \in L$, otherwise output 0.*

Define the following protocol between P and V.

Protocol (P, V).

1. **Common input:** CCDH instance $(X_1, Y_1) = (g^{x_1}, g^{y_1}) \in {G_1}^2$.
2. **Provers auxiliary input:** Witness to the CCDH instance $W = g^{x_1 y_1} \in G_1$.
3. **Common reference string:** An element $Y_2 \stackrel{R}{\leftarrow} G_1$ s.t. $Y_1 \neq Y_2$. Let $Y_2 = g^{y_2}$ for unknown y_2.
4. **Proof generation:** P generates $x_1 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $Z_2 = W \cdot {Y_2}^{x_2} \in G_1$. It outputs $(X_2, Z_2) \in {G_1}^2$. The complete proof is the tuple $(X_1, Y_1, X_2, Y_2, Z_2, g)$.
5. **Proof verification:** Accept the above proof if the following holds:

$$\hat{e}(X_1, Y_1) \cdot \hat{e}(X_2, Y_2) \stackrel{?}{=} \hat{e}(Z_2, g) \tag{3}$$

Theorem 4.3. *The above non-interactive protocol (P, V) is a NIZK proof of knowledge of the witness to the CCDH decision problem instance (X_1, Y_1).*

Proof. The proof is fairly straightforward. First note that completeness is trivial to verify:

$$LHS = \hat{e}(X_1, Y_1) \cdot \hat{e}(X_2, Y_2) = \hat{e}(g^{x_1}, g^{y_1}) \cdot \hat{e}(g^{x_2}, g^{y_2}) = \hat{e}(g^{x_1 y_1 + x_2 y_2}, g) = RHS$$

Zero Knowledge: We first prove that the protocol is zero-knowledge. The input is again the CCDH instance (X_1, Y_1). Simulator M generates two random elements $r, u \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$. It then computes

$X_2 = X_1 \cdot g^r$, $Y_2 = g^u / Y_1$ and $Z_2 = {X_1}^u \cdot {Y_2}^r$. It outputs X_2, Y_2, Z_2 as the simulated transcript. The tuple $(X_1, Y_1, X_2, Y_2, Z_2)$ is identical to a real transcript.

Proof of Knowledge: To show that the protocol is a proof of knowledge of the CDH solution, we construct the extractor E as follows. E is given the CCDH instance (X_1, Y_1) and black-box access to a prover that computes the above proof. E generates a random element $y_2 \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and sets $Y_2 = g^{y_2}$. It gives Y_2 as the random string to the prover P, who will output a proof of the form (X_2, Z_2) such that tuple $(X_1, Y_1, X_2, Y_2, Z_2, g)$ satisfies Equation 3. In this case E computes $W = Z_2/(X_2)^{y_2}$ and outputs W as the witness to the CCDH instance. □

5 Additive Non-Interactive Proofs

Another observation in the above protocol is that given the PoK $(X_1, Y_1, X_2, Y_2, Z_2, g)$, we can generate a new CCDH instance $(X_3, Y_3) = (g^{x_3}, g^{y_3})$ and form the tuple $(X_1, Y_1, X_2, Y_2, X_3, Y_3, Z_3, g)$, such that $Z_3 = Z_2 \cdot g^{x_3 y_3}$ behaves like a PoK of Z_2. We call this property "additiveness" - whenever a non-interactive PoK Z_i can be converted into a new non-interactive PoK Z_{i+1} of Z_i. This is formalized in this section. First we define the following problem.

5.1 The Composite Class-CDH problem

Let $S_i = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_i, Y_i)\}$ be a set containing i CCDH instances. Define $Z_i \in G_1$ to be the value such that

$$\prod_{(X_j, Y_j) \in S_i} \hat{e}(X_j, Y_j) = \hat{e}(Z_i, g) \tag{4}$$

Definition 5.1. ***Composite Class-CDH (CCCDH) problem****. Given S_i, compute Z_i.*

We say that Z_i is the CCCDH solution of the set S_i. It is easy to see that the the CCCDH problem is as hard as the CDH problem.

Lemma 5.1. *The CCCDH problem is hard if and only if the CDH problem is hard.*

Proof. The "only if" part is trivial and we will not prove it. For the "if" part, consider an adversary $\mathcal{A}$ who can always output the CCCDH solution of any set S_i. We can use $\mathcal{A}$ to solve any CDH instance (X, Y) of the CDH problem as follows. Generate random $x', y' \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and compute $X' = g^{x'}$; $Y' = g^{y'}$. The set $S_i = \{(X, Y), (X', Y')\}$ is given to $\mathcal{A}$, who outputs the CCCDH solution Z_i of S_i. In this case $Z/g^{x'y'}$ is the solution of our CDH instance. □

5.2 Additive Witness Indistinguishable Proofs

We now present a construction of an *Additive Non-Interactive Witness-Indistinguishable Proof of Knowledge* (A-NIWI-PoK). An A-NIWI-PoK can be instantly transferred into another another A-NIWI-PoK such that the new proof behaves like a PoK of the older PoK (and may include additional statements). Define the following protocol between a prover P and some verifier V.

Protocol (P, V)

1. **Common Input:** A set $S_n = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_n, Y_n)\}$ containing n CCDH instances with respect to a common generator g such that $Y_i \neq Y_j$ if $i \neq j$.
2. **Prover's Auxiliary Input:** Z_n, the CCCDH solution of S_n. P will prove knowledge of Z_n.
3. **Common Random String:** An element $Y_{n+1} \stackrel{R}{\leftarrow} G_1$ such that $Y_{n+1} \neq Y_j$ for $1 \leq j \leq n$,
4. **Proof Generation:** P generates $x_{n+1} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and sets $(X_{n+1}, Z_{n+1}) \leftarrow (g^{x_{n+1}}, Z_n \cdot Y_{n+1}^{x_{n+1}}) \in {G_1}^2$. It outputs (X_{n+1}, Z_{n+1}). Observe that Z_{n+1} is the CCCDH solution of $S_{n+1} = S_n \cup \{(X_{n+1}, Y_{n+1})\}$.

5. **Proof Verification:** V verifies that Z_{n+1} is indeed the CCCDH solution of S_{n+1}.

Theorem 5.2. *The pair (Z_{n+1}, S_{n+1}) is a NIWI-PoK of the CCCDH solution Z_n of S_n for all $n \geq 1$.*

Proof. Similar to ZK proofs, a WI proof has completeness, witness-indistinguishability and knowledge extractor requirements [5, 6]. Completeness is trivial.

Witness-Indistinguishability: The claim is true for $n = 1$ (because ZK implies WI). For any $n > 1$, given the set S_n and random string Y_{n+1}, we can construct a pair (X_{n+1}, Z_{n+1}) such that Z_{n+1} is the CCCDH solution of $S_{n+1} = S_n \cup \{(X_{n+1}, Y_{n+1})\}$. This can be done in at least two different ways: (1) Using the CCCDH solution Z_n of S_n and the witness for CCDH instance (X_{n+1}, Y_{n+1}). (2) Using the CCCDH solution of $S_{n+1} \backslash \{(X_1, Y_1)\}$ and the witness for CCDH instance (X_1, Y_1). Clearly, it is infeasible to distinguish which strategy was used.

Proof of Knowledge: We must construct an extractor E_{n+1} that that works as follows. First E_{n+1} outputs a random string Y_{n+1}, which is given to the prover. The prover then outputs a tuple (S_n, X_{n+1}, Z_{n+1}) such that S_n is a set containing n CCDH instances and Z_{n+1} is the CCCDH solution of $S_n \cup \{(X_{n+1}, Y_{n+1})\}$. Finally, E_{n+1} takes as input (S_n, X_{n+1}, Z_{n+1}) and outputs Z_n, the CCCDH solution of S_n. We construct this extractor as follows.

E_{n+1} generates $y_{n+1} \stackrel{R}{\leftarrow} \mathbb{Z}_q^*$ and computes $Y_{n+1} = g^{y_{n+1}} \in G_1$. E_{n+1} gives Y_{n+1} to some prover P who outputs a tuple (S_n, X_{n+1}, Z_{n+1}) such that S_n contains n CCDH instances and Z_{n+1} is the CCCDH solution of $S_{n+1} = S_n \cup \{(X_{n+1}, Y_{n+1})\}$. From this E_{n+1} computes $Z_n = Z_{n+1} \cdot X_{n+1}^{-y_{n+1}}$ and outputs Z_n as the CCCDH solution of S_n. □

5.2.1 Additiveness

Observe that any given NIWI-PoK (Z_n, S_n) can be instantly transferred into a new NIWI-PoK (Z_{n+1}, S_{n+1}) of (Z_n, S_n) (in other words, (Z_{n+1}, S_{n+1}) proves knowledge of (Z_n, S_n)). We call this property *additiveness* and any NIWI-PoK exhibiting this property an Additive NIWI-PoK (A-NIWI-PoK).

5.2.2 Is It Zero-Knowledge?

The witness indistinguishability property of above NIWI-PoK (combined with the intractability of the CCCDH problem) ensures that Z_{n+1} does not leak any "useful" information about the secret Z_n. However, we have been unable to construct a simulator and it is quite likely that the above proof is not zero-knowledge. To see why it may not be zero-knowledge (and still be witness hiding), observe that given the pair (Z_3, S_3) with $|S_3| = 3$, an adversary may be able to obtain some information about all the CCCDH solutions Z_2^* for the 3 sets $S_2^* \subsetneq S_3$ with $|S_2^*| = 2$ without getting any information about the witnesses of the individual CCDH instances of S_3.

6 Summary

In this paper we introduced the concept of *additive* proofs of knowledge using the underlying properties of aggregate signatures of [1]. There is one major difference between the additive proofs we described here and other ZK or WI proofs studied in the literature. Current models of NIZK and NIWI are based on proving a disjunction of statements (i.e., $x_1 \in L_1 \vee x_2 \in L_2$) such that information about individual statements cannot be obtained. In contrast, A-NIWI proofs are essentially based on proving a conjunction of statements (i.e., $x_1 \in L_1 \wedge x_2 \in L_2$) such that information about individual statements cannot be obtained. Additionally, our model of additive-NIWI-PoKs should not be confused with the idea of *multiple non-interactive zero-knowledge proofs under general assumptions* of Feige, Lapidot and Shamir (FLS) [16], where the authors propose the use of a single random string to prove multiple statements. Their proofs are proofs of membership, while ours are proofs of knowledge (in fact proofs of decision power).

Although the WI property of the protocol of Section 5.2 is sufficient to guarantee that nothing significant about the constituent NIWIs is revealed, zero-knowledge would be desirable. We can use the

technique of the simulator in the proof of Theorem 4.3 and achieve additive NIZK property at the cost of increasing the size of the proof to 2^n at n levels. As an open question, we would like to ask whether constant-size efficient additive NIZK PoKs exist.

In summary, we feel that this "additive" WI property of non-interactive proofs is intriguing and should be further investigated, especially because it may have applications in e-commerce.

References

[1] Dan Boneh, Craig Gentry, Ben Lynn, and Hovav Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 416–432. Springer, 2003.

[2] Amitabh Saxena and Ben Soh. One-way signature chaining: A new paradigm for group cryptosystems and e-commerce. Cryptology ePrint Archive, Report 2005/335, 2005.

[3] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM J. Comput.*, 18(1):186–208, 1989.

[4] Oded Goldreich. *Foundations of Cryptography I*, volume Basic Tools. Cambridge University Press, 2001.

[5] U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *STOC '90: Proceedings of the twenty-second annual ACM symposium on Theory of computing*, pages 416–426, New York, NY, USA, 1990. ACM Press.

[6] Cynthia Dwork and Moni Naor. Zaps and their applications. In *FOCS '00: Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 283–293, Washington, DC, USA, 2000. IEEE Computer Society.

[7] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. *Lecture Notes in Computer Science*, 740:390–420, 1993.

[8] Giovanni Di Crescenzo, Kouichi Sakurai, and Moti Yung. Zero-knowledge proofs of decision power: new protocols and optimal round-complexity. In *ICICS '97: Proceedings of the First International Conference on Information and Communication Security*, pages 17–27, London, UK, 1997. Springer-Verlag.

[9] Giovanni Di Crescenzo, Kouichi Sakurai, and Moti Yung. On zero-knowledge proofs (extended abstract): "from membership to decision". In *STOC '00: Proceedings of the thirty-second annual ACM symposium on Theory of computing*, pages 255–264, New York, NY, USA, 2000. ACM Press.

[10] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *STOC '88: Proceedings of the twentieth annual ACM symposium on Theory of computing*, pages 103–112. ACM Press, 1988.

[11] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO '91: Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, pages 433–444, London, UK, 1992. Springer-Verlag.

[12] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for np. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358. Springer, 2006.

[13] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[14] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[15] Jean-Sébastien Coron and David Naccache. Boneh et al.'s k-element aggregate extraction assumption is equivalent to the Diffie-Hellman assumption. In Chi-Sung Laih, editor, *ASIACRYPT*, volume 2894 of *Lecture Notes in Computer Science*, pages 392–397. Springer, 2003.

[16] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM J. Comput.*, 29(1):1–28, 2000.

[17] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In *STOC '89: Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 25–32, New York, NY, USA, 1989. ACM Press.