The related version of this paper appeared in *Proceedings of the 2nd ACM ASIA Public-Key Cryptography Workshop - ASIAPKC 2014*, pp. 49-58, under the title "Attribute-Based Signatures without Pairings via the Fiat-Shamir Paradigm".

A Concrete Procedure of Σ-protocol on Monotone Predicate*

Hiroaki Anada[1], Seiko Arita[2], and Kouichi Sakurai[3]

[1] Department of Information Security, University of Nagasaki
W408, 1-1-1, Manabino, Nagayo-cho, Nishisonogi-gun, Nagasaki, 851-2195 Japan
anada@sun.ac.jp
[2] Institute of Information Security
509, 2-14-1, Tsuruya-cho, Kanagawa-ku, Yokohama, 221-0835 Japan
arita@iisec.ac.jp
[3] Department of Informatics, Kyushu University
W2-712, 744, Motooka, Nishi-ku, Fukuoka, 819-0395 Japan
sakurai@inf.kyushu-u.ac.jp

Jan 8, 2018

Abstract. We propose a concrete procedure of a Σ-protocol proving knowledge that a set of witnesses satisfies a monotone predicate in the witness-indistinguishable manner. Inspired by the high-level work proposed by Cramer, Damgård and Schoenmakers at CRYPTO '94, we provide a concrete procedure by extending the so-called OR-proof.

Keywords: proof system, sigma-protocol, OR-proof.

1 Introduction

A Σ-protocol formalized in the doctoral thesis of Cramer [Cra96] is a protocol of a 3-move public-coin interactive proof system with the completeness, the special soundness and the honest-verifier zero-knowledge. It is one of the simplest protocols of zero-knowledge interactive proof systems with an easy simulator. Also, it is one of the most typical proof of knowledge systems [BG92]; witness-extraction property by the special soundness enables us to prove that an identification scheme by a Σ-protocol is secure against active and concurrent attacks via a reduction to a number-theoretic assumption [BP02]. Instantiations of the Σ-protocol have been known as the Schnorr protocol [Sch89] and the Guillou-Quisquater protocol [GQ88] of identification schemes. They can be converted into digital signature schemes by the Fiat-Shamir heuristic [FS86]. The signature scheme can be proved secure against chosen-message attacks in the random oracle model [PS96] based on the security of the identification scheme against passive attacks [AABN02]. By virtue of these features, a Σ-protocol can be adopted into building blocks of various cryptographic primitives such as anonymous credential systems [CL02] and group signature schemes [BBS04].

The OR-proof proposed by Cramer, Damgård and Schoenmakers at CRYPTO '94 [CDS94] is a Σ-protocol derived from an original Σ-protocol [Dam10]. It is a perfectly witness-indistinguishable protocol [FS90] by which a prover can convince a verifier that a prover knows one of two (or both) witnesses while even an unbounded distinguisher cannot tell which witness is used. The OR-proof is essentially applied in, for example, the construction of a non-malleable proof of plaintext knowledge [Kat03]. In the paper [CDS94], a more general protocol was

* The first and the second authors are partially supported by *kakenhi* Grant-in-Aid for Scientific Research (C) JP15K00029 from Japan Society for the Promotion of Science.

proposed[4]; suppose a prover and a verifier are given a monotone predicate f over boolean variables. Here a monotone predicate means a boolean predicate without negation; that is, boolean variables connected by AND-gates and OR-gates, but no NOT-gate is used. '1' (True) is substituted into every variable in f at which the prover knows the corresponding witness, and '0' (False) is substituted into every remaining variable. The protocol attains perfect witness-indistinguishability in the sense that the prover knows a satisfying set of witnesses while even an unbounded distinguisher cannot tell which satisfying set is used. This protocol is an extension of the OR-proof to any monotone predicate, and in [CDS94] a high-level construction that employed a "semi-smooth" secret-sharing scheme was given. (As is stated in [CDS94], to remove the restriction of the monotonicity of f looks impossible.)

1.1 Our Contribution

In this paper, we provide a concrete procedure of the protocol. We start with a given Σ-protocol Σ, and derive a Σ-protocol Σ_f for any monotone predicate f. Then we show that our Σ_f is actually a Σ-protocol with witness-indistinguishability.

Herranz [Her14] provided the first attribute-based signature scheme (ABS) with both collusion resistance (against collecting private secret keys) and computational attribute privacy *without pairings (pairing-free)* in the RSA setting. Recently, Herranz [Her16a] provided an ABS scheme without pairings in the discrete-logarithm setting with a constraint that the number of private secret keys is bounded in the set-up phase. In the both ABS schemes [Her14,Her16a], the concrete procedures were described in details for threshold-type access formulas. Our concrete procedure Σ_f of the Σ-protocol in [CDS94] for any monotone predicate serves as building blocks of those Σ-protocols in the pairing-free ABS schemes [Her14,Her16a].

1.2 Our Construction Idea

To provide a concrete procedure for the above protocol Σ_f with witness-indistinguishability, we look into the technique employed in the OR-proof [CDS94] and expand it so that it can treat any monotone predicate, as follows. First express the boolean predicate f as a binary tree $\mathcal{T}_f$. That is, we put leaves each of which corresponds to each position of a variable in f. We connect two leaves by an $\wedge$-node or an $\vee$-node according to an AND-gate or an OR-gate which is between two corresponding positions in f. Then we connect the resulting nodes by an $\wedge$-node or an $\vee$-node in the same way, until we reach to the root node (which is also an $\wedge$-node or an $\vee$-node). A verification equation of the Σ-protocol Σ is assigned to every leaf. If a challenge string $\textsc{Cha}$ of Σ is given, then the prover assigns the string $\textsc{Cha}$ to the root node. If the root node is an $\wedge$-node, then the prover assigns the same string $\textsc{Cha}$ to two children. Else if the root node is an $\vee$-node, then the prover divides $\textsc{Cha}$ into two random strings $\textsc{Cha}_{\mathrm{L}}$ and $\textsc{Cha}_{\mathrm{R}}$ under the constraint that $\textsc{Cha} = \textsc{Cha}_{\mathrm{L}} \oplus \textsc{Cha}_{\mathrm{R}}$, and assigns $\textsc{Cha}_{\mathrm{L}}$ and $\textsc{Cha}_{\mathrm{R}}$ to the left child and the right child, respectively. Here $\oplus$ means a bitwise exclusive-OR operation. Then the prover continues to apply this rule at each height, step by step, until he reaches to every leaf. Basically, the OR-proof technique assures that we can either honestly execute the Σ-protocol Σ or execute the simulator of Σ. Only when a set of witnesses satisfies the binary tree $\mathcal{T}_f$, the above procedure succeeds in satisfying verification equations for all leaves.

1.3 Organization of this Paper

In Section 2, we prepare for required tools and notions. In Section 3, we describe a concrete procedure of the Σ-protocol Σ_f. In Section 4, we conclude our work in this paper.

2 Preliminaries

The security parameter is denoted by λ. The bit length of a string a is denoted by $|a|$. The concatenation of a string a with a string b is denoted by $a \parallel b$. A uniform random sampling of an element a from a set S is denoted by $a \in_R S$. The expression $a =_? b$ returns a value 1 (True) when $a = b$ and 0 (False) otherwise. The expression $a \in_? S$ returns a value 1 when $a \in S$ and 0 otherwise. When an algorithm A with input a outputs z, we denote it as $z \leftarrow A(a)$, or, because of space limitation, $A(a) \rightarrow z$. When a probabilistic polynomial-time (ppt, for short)

[4] In the related version [AAS14] of this paper, the authors could not refer to this previous work. Now we would like to refer.

algorithm A with a random tape R and input a outputs z, we denote it as $z \leftarrow A(a; R)$ When A with input a and B with input b interact with each other and B outputs z, we denote it as $z \leftarrow \langle A(a), B(b)\rangle$. When A has oracle-access to $\mathbf{O}$, we denote it as $A^{\mathbf{O}}$. When A has concurrent oracle-access to n oracles $\mathbf{O}_1, \ldots, \mathbf{O}_n$, we denote it as $A^{\mathbf{O}_i|_{i=1}^n}$. Here "concurrent" means that A accesses oracles in arbitrarily interleaved order of messages. A probability of an event E is denoted by Pr[E]. A probability of an event E on condition that events $\mathrm{E}_1, \ldots, \mathrm{E}_m$ occur in this order is denoted as $\Pr[\mathrm{E}_1, \ldots, \mathrm{E}_m : \mathrm{E}]$.

2.1 Witness-Indistinguishable Proof System and Σ-protocol

Let $R = \{(x, w)\} \subset \{0,1\}^* \times \{0,1\}^*$ be a binary relation. We say that R is polynomially bounded if there exists a polynomial $\ell(\cdot)$ such that $|w| \le \ell(|x|)$ for any $(x, w) \in R$. In this paper, suppose that $|x|$ is bounded by $\ell_x(\lambda)$ and $|w|$ is bounded by $\ell_w(\lambda)$. We say that R is an NP relation if it is polynomially bounded and, in addition, there exists a polynomial-time algorithm for deciding membership of (x, w) in R. For a pair $(x, w) \in R$ we call x a statement and w a witness of x. An NP language for a NP relation R is defined as: $L \stackrel{\text{def}}{=} \{x \in \{0,1\}^*; \exists w \in \{0,1\}^*, (x, w) \in R\}$. We introduce a *relation-function* $R(\cdot,\cdot)$ associated with the relation R by: $R(\cdot,\cdot) : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$, $(x, w) \mapsto 1$ if $(x, w) \in R$, and 0 otherwise. Let us denote by $W(x)$ the set of witnesses of a statement x $\{w \in \{0,1\}^*; R(x, w) = 1\}$.

Witness-Indistinguishable Proof System [Bab85,GMR85,FS90,CDS94] Informally, a proof system [Bab85,GMR85] is witness indistinguishable if the verifier cannot tell which witness $w \in W(x)$ the prover is using. Let R be an NP relation. Suppose that a proof system $\Pi = (\mathbf{P}, \mathbf{V})$ on the relation R with the following property is given.
Witness-Indistinguishability. For any PPT algorithm $\mathbf{A}$, and for $x \in L$ and $w_0, w_1 \in W(x)$ of $\mathbf{A}$'s choice, we have the following computational indistinguishability.

$$\Pr[(x, w_0, w_1, st) \leftarrow \mathbf{A}(1^\lambda) : 1 \leftarrow \langle \mathbf{P}(x, w_0), \mathbf{A}(st)\rangle]$$
$$\approx_{\text{comp.}} \Pr[(x, w_0, w_1, st) \leftarrow \mathbf{A}(1^\lambda) : 1 \leftarrow \langle \mathbf{P}(x, w_1), \mathbf{A}(st)\rangle].$$

The proof system Π with the above property is said to be a *witness-indistinguishable* proof system (WI, for short). If the above indistinguishability holds for any unbounded algorithm $\mathbf{A}$, then the proof system Π is said to be a *perfectly witness-indistinguishable* proof system.

Σ-protocol [Cra96,Dam10] Let R be an NP relation. A *Σ-protocol* Σ on a relation R is a 3-move public-coin protocol of a proof system $\Pi = (\mathbf{P}, \mathbf{V})$ [Dam10]. $\mathbf{P}$ sends the first message called a commitment $\textsc{Cmt}$ to $\mathbf{V}$, then $\mathbf{V}$ sends the second message that is a public random string called a challenge $\textsc{Cha}$ to $\mathbf{P}$, and then $\mathbf{P}$ answers with the third message called a response $\textsc{Res}$ to $\mathbf{V}$. Then $\mathbf{V}$ applies a decision test on $(x, \textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ to return 1 (accept) or 0 (reject). If $\mathbf{V}$ accepts, then the triple $(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ is said to be an *accepting conversation on* x. Here $\textsc{Cha}$ is chosen uniformly at random from $\textsc{ChaSp}(1^\lambda) := \{0,1\}^{l(\lambda)}$ with $l(\cdot)$ being a super-log function. Moreover, Σ is detailed by the following six PPT algorithms, $\Sigma = (\Sigma^1, \Sigma^2, \Sigma^3, \Sigma^{\text{vrfy}}, \Sigma^{\text{ke}}, \Sigma^{\text{sim}})$. $\textsc{Cmt} \leftarrow \Sigma^1(x, w)$. This is the process of generating the first message $\textsc{Cmt}$ according to the protocol Σ on input $(x, w) \in R$. Similarly we denote $\textsc{Cha} \leftarrow \Sigma^2(1^\lambda)$, $\textsc{Res} \leftarrow \Sigma^3(x, w, \textsc{Cmt}, \textsc{Cha})$ and $b \leftarrow \Sigma^{\text{vrfy}}(x, \textsc{Cmt}, \textsc{Cha}, \textsc{Res})$. Furthermore, Σ must satisfy the following three requirements.
Completeness. A prover $\mathbf{P}$ with a witness w makes $\mathbf{V}$ accept with probability 1.
Special Soundness. There is a PPT algorithm called a *knowledge extractor* Σ^{ke}, which, given a statement x, computes a witness $\hat{w}$ satisfying $(x, \hat{w}) \in R$ from two accepting conversations of a common commitment message $\textsc{Cmt}$ and with different challenge messages $\textsc{Cha} \neq \textsc{Cha}'$: $(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ and $(\textsc{Cmt}, \textsc{Cha}', \textsc{Res}')$.

$$\hat{w} \leftarrow \Sigma^{\text{ke}}(x, \textsc{Cmt}, \textsc{Cha}, \textsc{Res}, \textsc{Cha}', \textsc{Res}').$$

Honest-Verifier Zero-Knowledge. There is a PPT algorithm called a *simulator* Σ^{sim} such that $(\tilde{\textsc{Cha}}, \tilde{\textsc{Cmt}}, \tilde{\textsc{Res}}) \leftarrow \Sigma^{\text{sim}}(x)$, where the distribution of $\{(\tilde{\textsc{Cmt}}, \tilde{\textsc{Cha}}, \tilde{\textsc{Res}})\}$ is the same as the distribution $\{(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})\}$ generated as real transcripts of accepting conversations between $\mathbf{P}(x, w)$ and $\mathbf{V}(x)$. Another equivalent variant of the simulator is constructed by using the fact that the challenge message $\textsc{Cha}$ is a public-coin. That is, $\tilde{\textsc{Cha}}$ is generated by running $\Sigma^2(1^\lambda)$ (i.e. uniform random sampling from $\textsc{ChaSp}(1^\lambda)$), then it is input to (another variant of) the simulator to generate the commitment message $\tilde{\textsc{Cmt}}$ and the response message $\tilde{\textsc{Res}}$:

$$(\tilde{\textsc{Cmt}}, \tilde{\textsc{Res}}) \leftarrow \Sigma^{\text{sim}}(x, \tilde{\textsc{Cha}}).$$

A proof system $\Pi = (\mathbf{P}, \mathbf{V})$ with a Σ-protocol is known to be a proof of knowledge system [BG92].

The OR-proof [Dam10] For a boolean predicate $f(X_1, X_2) = X_1 \vee X_2$, we consider a Σ-protocol Σ_{OR} on a relation R_{OR} below.

$$R_{\mathrm{OR}} = \{(x = (x_0, x_1), w = (w_0, w_1)) \in \{0,1\}^* \times \{0,1\}^*; \\ f(R(x_0, w_0), R(x_1, w_1)) = 1\}.$$

The corresponding language is

$$L_{\mathrm{OR}} = \{x \in \{0,1\}^*; \exists w, (x, w) \in R_{\mathrm{OR}}\}.$$

Suppose that a Σ-protocol Σ on a relation R is given. Then we can construct the protocol Σ_{OR} on the relation R_{OR} as follows. For instance, suppose $(x_0, w_0) \in R$ holds. $\mathbf{P}$ computes $\mathrm{CMT}_0 \leftarrow \Sigma^1(x_0, w_0), \mathrm{CHA}_1 \leftarrow \Sigma^2(1^\lambda)$, $(\mathrm{CMT}_1, \mathrm{RES}_1) \leftarrow \Sigma^{\mathrm{sim}}(x_1, \mathrm{CHA}_1)$ and sends $(\mathrm{CMT}_0, \mathrm{CMT}_1)$ to $\mathbf{V}$. Then $\mathbf{V}$ sends $\mathrm{CHA} \leftarrow \Sigma^2(1^\lambda)$ to $\mathbf{P}$. Then, $\mathbf{P}$ computes $\mathrm{CHA}_0 := \mathrm{CHA} \oplus \mathrm{CHA}_1, \mathrm{RES}_0 \leftarrow \Sigma^3(x_0, w_0, \mathrm{CMT}_0, \mathrm{CHA}_0)$ answers to $\mathbf{V}$ with $(\mathrm{CHA}_0, \mathrm{CHA}_1)$ and $(\mathrm{RES}_0, \mathrm{RES}_1)$. Here $\oplus$ denotes a bitwise exclusive-OR operation. Then both $(\mathrm{CMT}_0, \mathrm{CHA}_0, \mathrm{RES}_0)$ and $(\mathrm{CMT}_1, \mathrm{CHA}_1, \mathrm{RES}_1)$ are accepting conversations on x and have the same distribution as real accepting conversations. The protocol Σ_{OR} can be proved to be a Σ-protocol [CDS94,Dam10]. We often call Σ_{OR} the *OR-proof*. A proof system Π with the OR-proof is known to be a perfectly witness-indistinguishable proof of knowledge system (WIPoK) [CDS94,Dam10].

2.2 Boolean Predicate and Access Formula

Let $\mathcal{U} = \{1, \ldots, u\}$ be an attribute universe [GPSW06]. We must distinguish two cases: the case that $\mathcal{U}$ is small (that is, $|\mathcal{U}| = u$ is bounded by a polynomial in λ) and the case that $\mathcal{U}$ is large (that is, u is not necessarily bounded). We assume the small case in this paper.

Let $f = f(X_{i_1}, \ldots, X_{i_a})$ be a boolean predicate over boolean variables $U = \{X_1, \ldots, X_u\}$. That is, variables $X_{i_1}, \ldots, X_{i_a}$ are connected by boolean connectives; AND-gate ($\wedge$) and OR-gate ($\vee$). For example, $f = X_{\mathtt{at}_1} \wedge ((X_{\mathtt{at}_2} \wedge X_{\mathtt{at}_3}) \vee X_{\mathtt{at}_4})$ for some $\mathtt{at}_1, \mathtt{at}_2, \mathtt{at}_3, \mathtt{at}_4,\ 1 \leq \mathtt{at}_1 < \mathtt{at}_2 < \mathtt{at}_3 < \mathtt{at}_4 \leq u$. Note that there is a bijective map between boolean variables and attributes:

$$\psi : U \to \mathcal{U},\ \psi(X_i) \stackrel{\mathrm{def}}{=} i.$$

For $f(X_{i_1}, \ldots, X_{i_a})$, we denote the set of indices (that is, attributes) $\{i_1, \ldots, i_a\}$ by $\mathrm{Att}(f)$. We note the arity of f as $a(f)$. Hereafter we use the symbol i_j to mean the following:

$$i_j \stackrel{\mathrm{def}}{=} \text{the index } i \text{ of a boolean variable that is the } j\text{-th argument of } f.$$

Suppose that we are given an access structure as a boolean predicate f. For $S \in 2^{\mathcal{U}}$, we evaluate the boolean value of f at S as follows:

$$f(S) \stackrel{\mathrm{def}}{=} f\big(X_{i_j} \leftarrow [\psi(X_{i_j}) \in_? S]; j = 1, \ldots, a(f)\big) \in \{0,1\}.$$

Under this definition, a boolean predicate f can be seen as a map: $f : 2^{\mathcal{U}} \to \{0,1\}$. We call a boolean predicate f with this map an *access formula* over $\mathcal{U}$. In this paper, we assume that no NOT-gate ($\neg$) appears in f. In other words, we consider only *monotone* predicate and *monotone* access formulas.[5]

Access Tree A monotone access formula f can be represented by a finite binary tree $\mathcal{T}_f$. Each inner node represents a boolean connective, $\wedge$-gate or $\vee$-gate, in f. Each leaf corresponds to a term $X_{\mathtt{at}}$ (not a variable $X_{\mathtt{at}}$) in f in one-to-one way. For a finite binary tree tree $\mathcal{T}$, we denote the set of all nodes, the root node, the set of all leaves, the set of all inner nodes (that is, all nodes excluding leaves) and the set of all tree-nodes (that is, all nodes excluding the root node) as $\mathrm{Node}(\mathcal{T})$, $r(\mathcal{T})$, $\mathrm{Leaf}(\mathcal{T})$, $\mathrm{iNode}(\mathcal{T})$ and $\mathrm{tNode}(\mathcal{T})$, respectively. Then the attribute map $\rho(\cdot)$ is defined as:

$$\rho : \mathrm{Leaf}(\mathcal{T}) \to \mathcal{U},\ \rho(l) \stackrel{\mathrm{def}}{=} (\text{the attribute } i \text{ that corresponds to } l \text{ through } \psi).$$

If ρ is not injective, then we call the case *multi-use* of attributes.

If $\mathcal{T}$ is of height greater than 0, $\mathcal{T}$ has two subtrees whose root nodes are two children of $r(\mathcal{T})$. We denote the two subtrees by $\mathrm{Lsub}(\mathcal{T})$ and $\mathrm{Rsub}(\mathcal{T})$, which mean the left subtree and the right subtree, respectively.

[5] This limitation can be removed by adding *negation attributes* to $\mathcal{U}$ for each attribute in the original $\mathcal{U}$ though the size of the attribute universe $|\mathcal{U}|$ doubles.

$\mathbf{P}(x, w, f)$:		$\mathbf{V}(x, f)$:
$(v_n)_n \leftarrow \Sigma_f^{\mathrm{eval}}(\mathcal{T}_f, S)$		
If $v_{r(\mathcal{T}_f)} \neq 1$, then abort		
else		
$\Sigma_f^1(x, w, \mathcal{T}_f, (v_n)_n, \star)$		
$\to ((\textsc{Cmt}_l)_l, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$	$(\textsc{Cmt}_l)_l$ $\longrightarrow$	
	$\textsc{Cha}$	$\textsc{Cha} \leftarrow \Sigma_f^2(1^\lambda)$
$\Sigma_f^3(x, w, \mathcal{T}_f, (v_n)_n, (\textsc{Cmt}_l)_l,$	$\longleftarrow$	
$\textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$		$\Sigma_f^{\mathrm{vrfy}}(x, \mathcal{T}_f, (\textsc{Cmt}_l)_l,$
$\to ((\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$	$(\textsc{Cha}_n)_n, (\textsc{Res}_l)_l$	$\textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$
	$\longrightarrow$	$\to b$, Return b

Fig. 1. Our WIPoK Σ_f on the relation R_f.

3 Our Procedure of Σ-protocol on Monotone Predicate

In this section, we construct a Σ-protocol Σ_f of a witness-indistinguishable proof of knowledge system from a given Σ-protocol Σ and a monotone predicate f.

We revisit here the notion introduced by Cramer, Damgård and Schoenmakers [CDS94]; a Σ-protocol of a proof of knowledge system which is also a witness-indistinguishable proof system Let R be a binary relation. Let $f(X_{i_1}, \ldots, X_{i_{a(f)}})$ be a boolean predicate over boolean variables $U = \{X_1, \ldots, X_u\}$.

Definition 1 (Cramer, Damgård and Schoenmakers [CDS94], Our Rewritten Form) *A relation R_f is defined by:*

$$R_f \stackrel{def}{=} \{(x = (x_{i_1}, \ldots, x_{i_{a(f)}}), w = (w_{i_1}, \ldots, w_{i_{a(f)}})) \in \{0,1\}^* \times \{0,1\}^*;$$
$$f(R(x_{i_1}, w_{i_1}), \ldots, R(x_{i_{a(f)}}, w_{i_{a(f)}})) = 1\}.$$

R_f is a generalization of the relation R_{OR} [CDS94,Dam10], where f is a boolean predicate with the single boolean connective OR: $X_1 \vee X_2$. Note that, if R is an NP relation, then R_f is also an NP relation under the assumption that a, the arity of f, is bounded by a polynomial in λ. The corresponding language is

$$L_f \stackrel{\mathrm{def}}{=} \{x \in \{0,1\}^*; \exists w, (x, w) \in R_f\}.$$

In [CDS94], a 3-move public-coin honest-verifier zero-knowledge proof of knowledge system for the language L_f was defined as a witness-indistinguishable proof system on any monotone predicate f (satisfied by a set of witnesses). Then, in [CDS94], a Σ-protocol of the WIPoK system on the relation R_f was studied at a high level by using the notion of the dual access structure of the access structure determined by f.

3.1 Our Procedure

Now we construct a concrete procedure of a protocol Σ_f of a WIPoK system on the relation R_f. Σ_f is a 3-move public-coin protocol of a proof of knowledge system $\Pi = (\mathbf{P}, \mathbf{V})$ between interactive PPT algorithms $\mathbf{P}$ and $\mathbf{V}$, and it consists of seven algorithms: $\Sigma_f = (\Sigma_f^{\mathrm{eval}}, \Sigma_f^1, \Sigma_f^2, \Sigma_f^3, \Sigma_f^{\mathrm{vrfy}}, \Sigma_f^{\mathrm{ke}}, \Sigma_f^{\mathrm{sim}})$. In our prover algorithm $\mathbf{P}$, there are three PPT subroutines Σ_f^{eval}, Σ_f^1 and Σ_f^3. On the other hand, in our verifier algorithm $\mathbf{V}$, there are two PPT subroutines Σ_f^2 and Σ_f^{vrfy}. Moreover, Σ_f^{vrfy} has two subroutines **VrfyCha** and **VrfyRes**. Fig. 1 shows the construction of our procedure Σ_f. (For the tree expressions of a boolean predicate f, see Section 2.2.)

Evaluation of Satisfiability. The prover $\mathbf{P}$ begins with evaluation of whether and how S satisfies f by running the evaluation algorithm Σ_f^{eval}. It labels each node of $\mathcal{T}_f$ with a value $v = 1$ (TRUE) or 0 (FALSE). For each leaf l, we label l with $v_l = 1$ if $\rho(l) \in S$ and $v_l = 0$ otherwise. (For the definition of the function ρ, see Section 2.2.) For each inner node n, we label n with $v_n = v_{n_{\mathrm{L}}} \wedge v_{n_{\mathrm{R}}}$ or $v_n = v_{n_{\mathrm{L}}} \vee v_{n_{\mathrm{L}}}$ according to AND/OR evaluation of two labels of its two children $n_{\mathrm{L}}, n_{\mathrm{R}}$. The computation is executed for every node from the root to each leaf, recursively, as in Fig. 2.

Commitment. The prover $\mathbf{P}$ computes a commitment value for each leaf by running the algorithm Σ_f^1 described in Fig. 3. Basically, Σ_f^1 runs for every node from the root to each leaf, recursively. As a result, Σ_f^1 generates for

$$
\begin{array}{l}
\Sigma_f^{\mathrm{eval}}(\mathcal{T}, S): \\
\quad \mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T}) \\
\quad \text{If } r(\mathcal{T}) \text{ is an } \wedge\text{-node } n, \text{ then } v_n := \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S) \wedge \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S), \\
\quad\quad \text{Return } (v_n, \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S), \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S)) \\
\quad \text{else if } r(\mathcal{T}) \text{ is an } \vee\text{-node } n, \text{ then } v_n := \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S) \vee \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S), \\
\quad\quad \text{Return } (v_n, \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S), \Sigma_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S)) \\
\quad \text{else if } r(\mathcal{T}) \text{ is a leaf } l, \text{ then } v_l := (\rho(l) \in_? S) \\
\quad\quad \text{Return } (v_l)
\end{array}
$$

Fig. 2. The subroutine Σ_f^{eval} of our Σ_f.

each leaf l a value $\textsc{Cmt}_l$; If $v_l = 1$, then $\textsc{Cmt}_l$ is computed honestly according to Σ^1. Else if $v_l = 0$, then $\textsc{Cmt}_l$ is computed in the simulated way according to Σ^{sim}. Other values, $(\textsc{Cha}_n)_n$ and $(\textsc{Res}_l)_l)$, are needed for the simulation. Note that the distinguished symbol $\star$ is used to indicate "it is under computation".

$$
\begin{array}{l}
\Sigma_f^1(x, w, \mathcal{T}, (v_n)_n, \textsc{Cha}): \\
\quad \mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T}) \\
\quad \text{If } \quad r(\mathcal{T}) \text{ is an } \wedge\text{-node } n, \text{ then } \textsc{Cha}_n := \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} := \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \textsc{Cha} \\
\quad\quad \text{Return } (\textsc{Cha}_n, \Sigma_f^1(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}), \\
\quad\quad\quad\quad\quad \Sigma_f^1(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})})) \\
\quad \text{else if } r(\mathcal{T}) \text{ is an } \vee\text{-node } n, \text{ then } \textsc{Cha}_n := \textsc{Cha} \\
\quad\quad \text{If } \quad v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \text{ and } v_{r(\mathcal{T}_{\mathrm{R}})} = 1, \text{ then } \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} := \star, \quad \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \star \\
\quad\quad \text{else if } v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \text{ and } v_{r(\mathcal{T}_{\mathrm{R}})} = 0, \text{ then } \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} := \star, \quad \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} \leftarrow \Sigma^2(1^\lambda) \\
\quad\quad \text{else if } v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \text{ and } v_{r(\mathcal{T}_{\mathrm{R}})} = 1, \text{ then } \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma^2(1^\lambda), \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \star \\
\quad\quad \text{else if } v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \text{ and } v_{r(\mathcal{T}_{\mathrm{R}})} = 0, \text{ then } \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma^2(1^\lambda), \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \textsc{Cha} \oplus \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} \\
\quad\quad \text{Return } (\textsc{Cha}_n, \Sigma_f^1(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}), \\
\quad\quad\quad\quad\quad \Sigma_f^1(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})})) \\
\quad \text{else if } r(\mathcal{T}) \text{ is a leaf } l, \text{ then } \textsc{Cha}_l := \textsc{Cha} \\
\quad\quad \text{If } \quad v_l = 1, \text{ then } \textsc{Cmt}_l \leftarrow \Sigma^1(x_{\rho(l)}, w_{\rho(l)}), \textsc{Res}_l := \star \\
\quad\quad \text{else if } v_l = 0, \text{ then } (\textsc{Cmt}_l, \textsc{Res}_l) \leftarrow \Sigma^{\mathrm{sim}}(x_{\rho(l)}, \textsc{Cha}) \\
\quad\quad \text{Return}(\textsc{Cmt}_l, \textsc{Cha}_l, \textsc{Res}_l)
\end{array}
$$

Fig. 3. The subroutine Σ_f^1 of our Σ_f.

Challenge. The verifier $\mathbf{V}$ chooses a challenge value (that is, a public coin) by Σ^2.

$$\Sigma_f^2(1^\lambda): \textsc{Cha} \leftarrow \Sigma^2(1^\lambda), \mathrm{Return}(\textsc{Cha})$$

Fig. 4. The subroutine Σ_f^2 of our Σ_f.

Response. The prover $\mathbf{P}$ computes a response value for each leaf by running the algorithm Σ_f^3 described in Fig. 5. Basically, the algorithm Σ_f^3 runs for every node from the root to each leaf, recursively. As a result, Σ_f^3 generates the challenge values $(\textsc{Cha}_n)_n$ for all the nodes $n \in \mathrm{Node}(\mathcal{T}_f)$ and the response values $(\textsc{Res}_l)_l$ for all the leaves $l \in \mathrm{Leaf}(\mathcal{T}_f)$. Note that the computations of all challenge values $(\textsc{Cha}_n)_n$ are completed (according to the "division rule" described in Section 1.2).
Verification. The verifier $\mathbf{V}$ computes a decision boolean by running from the root to each leaf, recursively, the following algorithm Σ_f^{vrfy}.

Now we have to check that Σ_f is certainly a Σ-protocol on the relation R_f.

Proposition 1 (Completeness) *The completeness holds for our Σ_f.*

Proof. Suppose that $v_{r(\mathcal{T}_f)} = 1$. We show that, for every node in $\mathrm{Node}(\mathcal{T}_f)$, either $v_n = 1$ or $\textsc{Cha}_n \neq *$ holds after executing Σ_f^1. The proof is by induction on the height of $\mathcal{T}_f$. The case of height 0 follows from $v_{r(\mathcal{T}_f)} = 1$ and the completeness of Σ. Suppose that the case of height k holds and consider the case of height $k+1$. The construction of Σ_f^1 assures the case of height $k+1$. $\square$

$\Sigma_f^3(x, w, \mathcal{T}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is an $\wedge$-node n, then $\textsc{Cha}_n := \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} := \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \textsc{Cha}$
Return$(\textsc{Cha}_n, \Sigma_f^3(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l),$
$\Sigma_f^3(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l))$
else if $r(\mathcal{T})$ is an $\vee$-node n, then $\textsc{Cha}_n := \textsc{Cha}$
If $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \Sigma^2(1^\lambda), \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \textsc{Cha} \oplus \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 1$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} := \textsc{Cha} \oplus \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})} := \textsc{Cha} \oplus \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0$ and $v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then do nothing
Return$(\textsc{Cha}_n, \Sigma_f^3(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l),$
$\Sigma_f^3(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l))$
else if $r(\mathcal{T})$ is a leaf l, then $\textsc{Cha}_l := \textsc{Cha}$
If $v_l = 1$, then $\textsc{Res}_l \leftarrow \Sigma^3(x_{\rho(l)}, w_{\rho(l)}, \textsc{Cmt}_l, \textsc{Cha})$
else if $v_l = 0$, then do nothing
Return$(\textsc{Cha}_l, \textsc{Res}_l)$

Fig. 5. The subroutine Σ_f^3 of our Σ_f.

$\Sigma_f^{\mathrm{vrfy}}(x, \mathcal{T}, (\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$:
Return(**VrfyCha**$(\mathcal{T}, \textsc{Cha}, (\textsc{Cha}_n)_n) \wedge$ **VrfyRes**$(x, \mathcal{T}, (\textsc{Cmt}_l)_l, (\textsc{Cha}_l)_l, (\textsc{Res}_l)_l)$)

VrfyCha$(\mathcal{T}, \textsc{Cha}, (\textsc{Cha}_n)_n)$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is an $\wedge$-node n, then
Return $((\textsc{Cha} =_? \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}) \wedge (\textsc{Cha} =_? \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge$**VrfyCha**$(\mathcal{T}_{\mathrm{L}}, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_n)_n) \wedge$ **VrfyCha**$(\mathcal{T}_{\mathrm{R}}, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_n)_n))$
else if $r(\mathcal{T})$ is an $\vee$-node n, then
Return $((\textsc{Cha} =_? \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})} \oplus \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge$**VrfyCha**$(\mathcal{T}_{\mathrm{L}}, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_n)_n) \wedge$ **VrfyCha**$(\mathcal{T}_{\mathrm{R}}, \textsc{Cha}_{r(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_n)_n))$
else if $r(\mathcal{T})$ is a leaf l, then
Return $(\textsc{Cha} \in_? \textsc{ChaSp}(1^\lambda))$

VrfyRes$(x, \mathcal{T}, (\textsc{Cmt}_l)_l, (\textsc{Cha}_l)_l, (\textsc{Res}_l)_l)$:
For $l \in \mathrm{Leaf}(\mathcal{T})$: If $\Sigma^{\mathrm{vrfy}}(x_{\rho(l)}, \textsc{Cmt}_l, \textsc{Cha}_l, \textsc{Res}_l) = 0$, then Return (0)
Return (1)

Fig. 6. The subroutine Σ_f^{vrfy} of our Σ_f.

Proposition 2 (Special Soundness) *The special soundness holds for our Σ_f.*

We can construct a knowledge extractor Σ_f^{ke} from a knowledge extractor Σ^{ke} of the underlying Σ-protocol Σ as follows. Then Lemma 1 assures the above proposition.

$\Sigma_f^{\mathrm{ke}}(x, f, (\textsc{Cmt}_l)_l,\ \ \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l,\ \ \textsc{Cha}', (\textsc{Cha}'_n)_n, (\textsc{Res}'_l)_l)$:
 If $\textsc{Cha} = \textsc{Cha}'$ then Return $\textsc{TheSameCha}$
 else if $\Sigma_f^{\mathrm{vrfy}}(x, \mathcal{T}_f, \textsc{Cha}, (\textsc{Cmt}_l)_l, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l) = 0$
  or $\Sigma_f^{\mathrm{vrfy}}(x, \mathcal{T}_f, \textsc{Cha}', (\textsc{Cmt}_l)_l, (\textsc{Cha}'_n)_n, (\textsc{Res}'_l)_l) = 0$, then Return $\perp$
 else
  For $l \in \mathrm{Leaf}(\mathcal{T}_f)$:
   If $\textsc{Cha}_l = \textsc{Cha}'_l$, then $\hat{w}_{\rho(l)} \in_R \{0,1\}^{\ell_w(\lambda)}$
   else $\hat{w}_{\rho(l)} \leftarrow \Sigma^{\mathrm{ke}}(x_{\rho(l)}, \textsc{Cmt}_l, \textsc{Cha}_l, \textsc{Res}_l, \textsc{Cha}'_l, \textsc{Res}'_l)$
  Return $(\hat{w} := (\hat{w}_{\mathtt{at}_j})_{1 \le j \le a(f)})$

Fig. 7. The knowledge-extractor Σ_f^{ke} of our Σ_f.

Lemma 1 (Witness Extraction) *The string $\hat{w}$ output by Σ_f^{ke} satisfies $(x, \hat{w}) \in R_f$.*

Proof. We prove the lemma by induction on the number of all $\vee$-nodes in $\mathrm{iNode}(T_f)$. First remark that $\textsc{Cha} \neq \textsc{Cha}'$.

Suppose that all nodes in $\mathrm{iNode}(T_f)$ are $\wedge$-nodes. Then the above claim follows immediately because $\textsc{Cha}_l \neq \textsc{Cha}'_l$ holds for all leaves.

Suppose that the case of k $\vee$-nodes holds and consider the case of $k+1$ $\vee$-nodes. Look at one of the lowest height $\vee$-node and name the height and the node as h^* and n^*, respectively. Then $\textsc{Cha}_{n^*} \neq \textsc{Cha}'_{n^*}$ because all nodes with their heights less than h^* are $\wedge$-nodes. So at least one of children of n^*, say n^*_L, satisfies $\textsc{Cha}_{n^*_L} \neq \textsc{Cha}'_{n^*_L}$. Divide the tree $\mathcal{T}_f$ into two subtrees by cutting the branch right above n^*, and the induction hypothesis assures the claim. $\square$

Proposition 3 (HVZK) *The honest-verifier zero-knowledge property holds for our Σ_f.*

Proof. We construct a polynomial-time simulator Σ_f^{sim}, which on input a statement $xinL_f$ and a predicate f returns an accepting conversation $((\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l))$, as follows.

$\Sigma_f^{\mathrm{sim}}(x, f)$:
 $\tilde{\textsc{Cha}} \leftarrow \Sigma_f^{\mathrm{sim}}(1^\lambda)$, $w \in_R \{0,1\}^{\ell_w(\lambda)}$, For $n \in \mathrm{Node}(\mathcal{T}_f) : v_n := 0$
 $((\tilde{\textsc{Cmt}}_l)_l, (\tilde{\textsc{Cha}}_n)_n, (\tilde{\textsc{Res}}_l)_l) \leftarrow \Sigma_f^1(x, w, \mathcal{T}_f, (v_n)_n, \tilde{\textsc{Cha}})$
 $\mathrm{Return}((\tilde{\textsc{Cmt}}_l)_l, \tilde{\textsc{Cha}}, (\tilde{\textsc{Cha}}_n)_n, (\tilde{\textsc{Res}}_l)_l))$

Fig. 8. The simulator Σ_f^{sim} of our Σ_f.

$\square$

We summarize the above results into the following theorem and corollary.

Theorem 1 (Σ_f is a Σ-protocol) *Suppose that a Σ-protocol Σ on a relation R and a boolean predicate f is given. Then, our procedure Σ_f is a Σ-protocol on the relation R_f.*

Theorem 2 (Σ_f is WIPoK) *Our Σ-protocol Σ_f is a procedure of a perfectly witness-indistinguishable proof of knowledge system on the relation R_f.*

Proof. For a fixed statement x and two witnesses w_1 and w_2 satisfying $R(x, w_1) = R(x, w_2) = 1$ or $R(x, w_1) = R(x, w_2) = 0$, $\mathbf{P}(x, w)$ and $\mathbf{V}(x)$ of Σ_f generate transcripts $((\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$ that have the same distribution. $\square$

3.2 Non-interactive Version

The Fiat-Shamir transform $\mathrm{FS}(\cdot)$ can be applied to any Σ-protocol Σ ([FS86,AABN02]). Therefore, the non-interactive version of our procedure Σ_f is obtained.

Theorem 3 (FS(Σ_f) is NIWIPoK) *Our FS(Σ_f) is a procedure of a non-interactive perfectly witness-indistinguishable proof of knowledge system on the relation R_f. A knowledge extractor is constructed in the random oracle model.*

3.3 Discussion

As is mentioned in [CDS94], the Σ-protocol Σ_f can be considered as a proto-type of an attribute-based identification scheme. Also, the non-interactive version $\mathrm{FS}(\Sigma_f)$ can be considered a proto-type of an attribute-based signature scheme. That is, Σ_f and $\mathrm{FS}(\Sigma_f)$ are an attribute-based identification scheme and an attribute-based signature scheme *without collusion resistance on collecting private secret keys*, respectively [6].

4 Conclusion

We provided a concrete procedure Σ_f of a Σ-protocol of the WIPoK system on monotone predicates. Our Σ_f can be considered as a proto-type of an attribute-based identification scheme, and also, $\mathrm{FS}(\Sigma_f)$ can be considered a proto-type of an attribute-based signature scheme [CDS94], without collusion resistance on private secret keys. Our procedure Σ_f for any monotone predicate serves as building blocks of the Σ-protocols in the pairing-free ABS schemes [Her14,Her16a].

Acknowledgements We appreciate sincere comments from Javier Herranz via e-mail communication [Her16b] on the topic in this paper. We would like to thank to Keita Emura and Takahiro Matsuda for their sincere comments and encouragements on the construction of attribute-based signature schemes. We would like to thank to Shingo Hasegawa and Masayuki Fukumitsu for their sincere comments on the construction of the Σ-protocol on monotone predicates.

References

[AABN02] Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*, pages 418–433, 2002.

[AAS14] Hiroaki Anada, Seiko Arita, and Kouichi Sakurai. Attribute-based signatures without pairings via the fiat-shamir paradigm. In *ASIAPKC2014*, volume 2 of *ACM-ASIAPKC*, pages 49–58. ACM, 2014.

[Bab85] László Babai. Trading group theory for randomness. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 421–429, 1985.

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, pages 41–55, 2004.

[BG92] Mihir Bellare and Oded Goldreich. On defining proofs of knowledge. In *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, pages 390–420, 1992.

[BP02] Mihir Bellare and Adriana Palacio. GQ and Schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *Advances in Cryptology - CRYPTO 2002, 22nd Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 2002, Proceedings*, pages 162–177, 2002.

[CDS94] Ronald Cramer, Ivan Damgård, and Berry Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994.

[6] In the related version [AAS14] of this paper, we attained the collusion resistance in the construction of an attribute-based identification scheme (ABID) and an attribute-based signature scheme (ABS) by a naive application of the credential bundle technique [MPR11]. But instead we lost the *attribute privacy* in the ABID and the ABS schemes though the attribute privacy was *wrongly* claimed in [AAS14].

[CL02] Jan Camenisch and Anna Lysyanskaya. A signature scheme with efficient protocols. In *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, pages 268–289, 2002.

[Cra96] Ronald Cramer. *Modular Designs of Secure, yet Practical Cyptographic Protocols.* PhD thesis, University of Amsterdam, Amsterdam, the Netherlands, 1996.

[Dam10] Ivan Damgård. On σ-protocols. In Course Notes, http://cs.au.dk/ ivan/CPT.html, 2010.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*, pages 186–194, 1986.

[FS90] Uriel Feige and Adi Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, May 13-17, 1990, Baltimore, Maryland, USA*, pages 416–426, 1990.

[GMR85] S Goldwasser, S Micali, and C Rackoff. The knowledge complexity of interactive proof-systems. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 291–304, New York, NY, USA, 1985. ACM.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 - November 3, 2006*, pages 89–98, 2006.

[GQ88] Louis C. Guillou and Jean-Jacques Quisquater. A "paradoxical" indentity-based signature scheme resulting from zero-knowledge. In *Advances in Cryptology - CRYPTO '88, 8th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1988, Proceedings*, pages 216–231, 1988.

[Her14] Javier Herranz. Attribute-based signatures from rsa. *Theoretical Computer Science*, 527:73–82, 2014.

[Her16a] Javier Herranz. Attribute-based versions of schnorr and elgamal. *Appl. Algebra Eng. Commun. Comput.*, 27(1):17–57, 2016.

[Her16b] Javier Herranz. Private communication via e-mail, dept. matemàtica aplicada iv, universitat politècnica de catalunya, July 2014, Sept 2015 and May 2016.

[Kat03] Jonathan Katz. Efficient and non-malleable proofs of plaintext knowledge and applications. In *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, pages 211–228, 2003.

[MPR11] Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In *Topics in Cryptology - CT-RSA 2011 - The Cryptographers' Track at the RSA Conference 2011, San Francisco, CA, USA, February 14-18, 2011. Proceedings*, pages 376–392, 2011.

[PS96] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In *Advances in Cryptology - EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 387–398, 1996.

[Sch89] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*, pages 239–252, 1989.