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 the Σ-protocol on Monotone Predicates[⋆]

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

Feb 26, 2018

Abstract. We propose a concrete procedure of the Σ-protocol introduced by Cramer, Damgård and Schoenmakers at CRYPTO '94, which is for proving knowledge that a set of witnesses satisfies a monotone predicate in witness-indistinguishable way. We provide the concrete procedure by extending the so-called OR-proof.

Keywords: proof system, sigma-protocol, OR-proof, witness indistinguishability

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 which satisfies the three requirements of completeness, special soundness and honest-verifier zero-knowledge. It is one of the simplest protocols of zero-knowledge interactive proof systems with the easy but special simulator. Also, it is one of the most typical proof of knowledge systems [BG92]; the knowledge-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 hardness assumption [BP02]. For example, instantiations of the Σ-protocol have been known as the Schnorr protocol [Sch89] and the Guillou-Quisquater protocol [GQ88] of identification schemes. The identification schemes can be converted into digital signature schemes by the Fiat-Shamir heuristic [FS86]. The signature schemes can be proved secure against chosen-message attacks in the random oracle model [PS96] based on the security of the identification schemes 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 the two or both witnesses while even

[⋆] 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.

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][1], a more general protocol was proposed; suppose a prover and a verifier are given a monotone predicate f over boolean variables. Here a monotone predicate means a boolean-valued function which is a boolean formula without negation; that is, as a boolean formula, boolean variables of f are connected by AND-gates or OR-gates, but no NOT-gate is used. '1' (True) is assigned into every variable in f at which the prover knows the corresponding witness, and '0' (False) is assigned into every remaining variable. The protocol attains the 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 and Related Works

In this paper, we provide a concrete procedure of the Σ-protocol proposed by Cramer, Damgård and Schoenmakers [CDS94]. We start with a given Σ-protocol Σ, and derive a Σ-protocol Σ_f for any monotone predicate f concretely. Then we show that the protocol Σ_f realized by our procedure is actually a Σ-protocol with the perfect witness indistinguishability.

Explanation on the relation to attribute-based cryptographic primitives should be in order[2]. Herranz [Her14] provided the first attribute-based signature scheme (ABS) with both the collusion resistance (against collecting private secret keys) and the computational attribute privacy, while the scheme is *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] Σ-protocols are used and described for the threshold-type predicate. Our concrete procedure of the Σ-protocol Σ_f serves as building blocks of their Σ-protocols for any monotone predicate (including the threshold-type predicate) to yield the pairing-free ABS schemes [Her14,Her16a].

1.2 Our Construction Idea

To construct a concrete procedure for the Σ-protocol Σ_f with the perfect 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 formula 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 the 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 the root node (which is also an $\wedge$-node or an $\vee$-node). A verification equation of the given Σ-protocol Σ is assigned to every leaf. If a challenge string $\textsc{Cha}$ of our Σ-protocol Σ_f is given by the verifier, 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 the 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 she reaches all the leaves. Basically, the OR-proof technique assures that, at every leaf, 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 at all the leaves.

[1] In the related paper [AAS14] of this ePrint, the authors could not refer to this previous work [CDS94]. We would like to refer to the work now.

[2] In the related paper [AAS14] of this ePrint, 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].

1.3 Organization of this Paper

In Section 2, we prepare for required notions and notations. 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 as $a \in_R S$. 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, $A(a) \rightarrow z$. When a algorithm A with input a and a algorithm B with input b interact with each other, we denote the transcript of the messages as $\langle A(a), B(b)\rangle$.

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 $|w| \leq \ell(|x|)$ for any $(x, w) \in R$. We say that R is an NP relation if it is polynomially bounded and 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 an NP relation R is defined as: $L \overset{\text{def}}{=} \{x \in \{0,1\}^*; \exists w \in \{0,1\}^*, (x, w) \in R\}$. We introduce a *relation function* $R(\cdot,\cdot)$ associated with the relation R by: $R(\cdot,\cdot) : \{0,1\}^* \times \{0,1\}^* \rightarrow \{0,1\}$, $(x, w) \mapsto 1$ if $(x, w) \in R$, and 0 otherwise. The function $R(\cdot,\cdot)$ is polynomial-time in $|x|$ as an algorithm. We denote the set of witnesses of a statement x by $W(x)$.

We denote an interactive proof system on an NP relation R [Bab85,GMR85] as $\Pi = (\mathbf{P}, \mathbf{V})$, where $\mathbf{P}$ and $\mathbf{V}$ are a pair of interactive Turing machines, which are called a prover and a verifier, respectively. In this paper, not only $\mathbf{V}$ but also $\mathbf{P}$ are assumed to be probabilistic polynomial-time (PPT). That is, $\Pi = (\mathbf{P}, \mathbf{V})$ is an interactive argument system.

2.1 Witness-Indistinguishable Proof System and Σ-protocol

Witness-Indistinguishable Proof System [FS90,Gol01] Let R be an NP relation. Suppose that an interactive proof system $\Pi = (\mathbf{P}, \mathbf{V})$ on the relation R is given. We consider the following property.
Witness Indistinguishability. For any PPT algorithm $\mathbf{V}^*$, any sequences $W^0 = (w_x^0)_{x\in L}$ and $W^1 = (w_x^1)_{x\in L}$ s.t. $w_x^0, w_x^1 \in W(x)$, any PPT algorithm D, any polynomial poly$(\cdot)$, any sufficiently long string $x \in L$ and any string $z \in \{0,1\}^*$,

$$\begin{aligned}&\Pr[D\big(x, z, \langle \mathbf{P}(x, w_x^0), \mathbf{V}^*(x, z)\rangle\big) = 1]\\ &- \Pr[D\big(x, z, \langle \mathbf{P}(x, w_x^1), \mathbf{V}^*(x, z)\rangle\big) = 1] < \frac{1}{\text{poly}(|x|)}.\end{aligned}$$

The interactive proof system Π with the above property is said to be a *witness-indistinguishable* proof system (WI, for short). A stronger notion is the perfect witness indistinguishability. If for any PPT algorithm $\mathbf{V}^*$, any sequences $W^0 = (w_x^0)_{x\in L}$ and $W^1 = (w_x^1)_{x\in L}$ s.t. $w_x^0, w_x^1 \in W(x)$, any string $x \in L$ and any string $z \in \{0,1\}^*$ the two distributions $\{(x, z, \langle \mathbf{P}(x, w_x^0), \mathbf{V}^*(x, z)\rangle)\}$ and $\{(x, z, \langle \mathbf{P}(x, w_x^1), \mathbf{V}^*(x, z)\rangle)\}$ are identical, then the interactive 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 an interactive proof system $\Pi = (\mathbf{P}, \mathbf{V})$ [Cra96,Dam10]. $\mathbf{P}$ sends the first message called a commitment Cmt to $\mathbf{V}$. Then $\mathbf{V}$ sends the second message called a challenge Cha to $\mathbf{P}$, which is a public random string. Then $\mathbf{P}$ sends the third message called a response Res to $\mathbf{V}$. Then $\mathbf{V}$ applies a decision test to $(x, \text{Cmt}, \text{Cha}, \text{Res})$ to return 1 (accept) or 0 (reject). If $\mathbf{V}$ accepts, then the triple (Cmt, Cha, Res) is said to be an *accepting transcript on x*. The challenge Cha is chosen uniformly at random from the challenge space $\text{ChaSp}(1^\lambda) := \{0,1\}^{l(\lambda)}$ with $l(\cdot)$ being a super-log function. To state the requirements for the Σ-protocol Σ, we introduce the following six PPT algorithms of Σ: $\Sigma = (\Sigma^1, \Sigma^2, \Sigma^3, \Sigma^{\text{vrfy}}, \Sigma^{\text{ke}}, \Sigma^{\text{sim}})$. The first algorithm Σ^1 is described as $\text{Cmt} \leftarrow \Sigma^1(x, w)$. That is, on input $(x, w) \in R$, it generates a commitment Cmt. Similarly,

the second, the third and the forth algorithms are described as $\textsc{Cha} \leftarrow \Sigma^2(1^\lambda)$, $\textsc{Res} \leftarrow \Sigma^3(x, w, \textsc{Cmt}, \textsc{Cha})$ and $b \leftarrow \Sigma^{\mathrm{vrfy}}(x, \textsc{Cmt}, \textsc{Cha}, \textsc{Res})$, respectively. Σ must satisfy the following three requirements.
Completeness. A prover $\mathbf{P}(x, w)$ with a witness $w \in W(x)$ makes $\mathbf{V}(x)$ accept with the probability 1.

The fifth algorithm is described as follows.
Special Soundness. There is a PPT algorithm called a *knowledge extractor* Σ^{ke}, which, given as input a statement x and two accepting transcripts $(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ and $(\textsc{Cmt}, \textsc{Cha}', \textsc{Res}')$, computes a witness $\hat{w}$ satisfying $(x, \hat{w}) \in R$ with an overwhelming probability, where the two challenges $\textsc{Cha}$ and $\textsc{Cha}'$ are different $(\textsc{Cha} \neq \textsc{Cha}')$:

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

The sixth algorithm is described as follows.
Honest-Verifier Zero-Knowledge. For any fixed statement x there is a PPT algorithm called a *simulator* Σ^{sim} such that

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

where the distribution of (simulated) transcripts $\{(\tilde{\textsc{Cmt}}, \tilde{\textsc{Cha}}, \tilde{\textsc{Res}})\}$ is the same as the distribution of (real) accepting transcripts $\{(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})\}$ generated as $\langle \mathbf{P}(x, w), \mathbf{V}(x) \rangle$ for any fixed witness $w \in W(x)$ and for the (honest) verifier $\mathbf{V}$.

For a Σ-protocol, the above simulator $\Sigma^{\mathrm{sim}}(x)$ is modified as follows. First generate a challenge $\tilde{\textsc{Cha}}$ by running $\Sigma^2(1^\lambda)$ (i.e. uniform random sampling from $\textsc{ChaSp}(1^\lambda)$), and then input the challenge $\tilde{\textsc{Cha}}$ to the modified simulator to generate a commitment $\tilde{\textsc{Cmt}}$ and a response $\tilde{\textsc{Res}}$:

$$\tilde{\textsc{Cha}} \leftarrow \Sigma^2(1^\lambda),\ (\tilde{\textsc{Cmt}}, \tilde{\textsc{Res}}) \leftarrow \Sigma^{\mathrm{sim}}(x, \tilde{\textsc{Cha}}).$$

We need this modified form of the simulator later.

We note that an interactive proof system $\Pi = (\mathbf{P}, \mathbf{V})$ with a Σ-protocol is known to be a proof of knowledge system. (For the notion of a proof of knowledge system, see [BG92].)

The OR-proof [Dam10] We consider a Σ-protocol Σ_{OR} on a relation R_{OR} about a boolean formula $f(X_0, X_1) = X_0 \vee X_1$, where

$$\begin{aligned} R_{\mathrm{OR}} =&\{(x = (x_0, x_1), w = (w_0, w_1)) \in (\{0,1\}^*)^2 \times (\{0,1\}^*)^2; \\ &f(R(x_0, w_0), R(x_1, w_1)) = 1\}. \end{aligned}$$

The corresponding language is

$$L_{\mathrm{OR}} = \{x \in (\{0,1\}^*)^2; \exists w \in (\{0,1\}^*)^2, (x, w) \in R_{\mathrm{OR}}\}.$$

Suppose that a Σ-protocol Σ on a relation R is given. Then we construct the protocol Σ_{OR} on the relation R_{OR} as follows. Suppose wolog $(x_0, w_0) \in R$. $\mathbf{P}$ computes $\textsc{Cmt}_0 \leftarrow \Sigma^1(x_0, w_0)$, $(\textsc{Cmt}_1, \textsc{Cha}_1, \textsc{Res}_1) \leftarrow \Sigma^{\mathrm{sim}}(x_1)$ and sends $(\textsc{Cmt}_0, \textsc{Cmt}_1)$ to $\mathbf{V}$. Then $\mathbf{V}$ chooses $\textsc{Cha} \leftarrow \Sigma^2(1^\lambda)$ and sends it to $\mathbf{P}$. Then $\mathbf{P}$ computes $\textsc{Cha}_0 := \textsc{Cha} \oplus \textsc{Cha}_1, \textsc{Res}_0 \leftarrow \Sigma^3(x_0, w_0, \textsc{Cmt}_0, \textsc{Cha}_0)$ and sends $(\textsc{Cha}_0, \textsc{Cha}_1)$ and $(\textsc{Res}_0, \textsc{Res}_1)$ to $\mathbf{V}$. Here $\oplus$ denotes a bitwise exclusive-OR operation. Then for each $i = 0, 1$, $(\textsc{Cmt}_i, \textsc{Cha}_i, \textsc{Res}_i)$ is an accepting transcript on x_i, and furthermore, the distribution of transcripts $\{(\textsc{Cmt}_i, \textsc{Cha}_i, \textsc{Res}_i)\}$ is the same as the distribution of accepting transcripts generated as $\langle \mathbf{P}(x_i, w_i), \mathbf{V}(x_i) \rangle$ for any fixed $w_i \in W(x_i)$.

The protocol Σ_{OR} is actually a Σ-protocol [CDS94,Dam10]. We often call Σ_{OR} the *OR-proof.* A proof system Π with the OR-proof is, as we see, perfectly witness-indistinguishable [CDS94,Dam10]. Thus, a proof system Π with the OR-proof is a perfectly witness-indistinguishable proof of knowledge system (WIPoK).

2.2 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 formula over boolean variables $U = \{X_1, \ldots, X_u\}$. Here we denote the arity of f as $a(f)$, and two variables among $X_{i_1}, \ldots, X_{i_a}$ are connected by a boolean connective, an AND-gate ($\wedge$) or an OR-gate ($\vee$). For example, $f = X_{i_1} \wedge ((X_{i_2} \wedge X_{i_3}) \vee X_{i_4})$ for some $i_1, i_2, i_3, i_4 \in \mathcal{U}$. Note that there is a bijective map ψ between boolean variables and attributes:

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

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

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

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

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

Under this definition, a boolean formula f can be seen as a map: $f : 2^{\mathcal{U}} \to \{0, 1\}$. We call a boolean formula 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* predicates and *monotone* access formulas.

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, an $\wedge$-gate or an $\vee$-gate, in f. Each leaf corresponds to a position X_i (not a variable X_i) 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:

$$\rho : \mathrm{Leaf}(\mathcal{T}) \to \mathcal{U},\ \rho(l) \overset{\text{def}}{=} (\psi(X_i) \text{ where } l \text{ corresponds to the position } X_i).$$

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.

3 Our Procedure of Σ-protocol on Monotone Predicate

In this section, we construct a Σ-protocol Σ_f of a perfectly witness-indistinguishable proof of knowledge system from a given Σ-protocol Σ and a monotone predicate f, so that Σ_f will be an extension of the OR-proof Σ_{OR}.

We revisit first the notion introduced by Cramer, Damgård and Schoenmakers [CDS94]; a Σ-protocol of a perfectly witness-indistinguishable proof of knowledge system. Let R be a binary relation. Let $f(X_{i_1}, \ldots, X_{i_{a(f)}})$ be a boolean formula 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:*

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

R_f is a generalization of the relation R_{OR} [CDS94,Dam10] where f was a boolean formula with a single boolean connective OR, i.e. $f = X_{i_1} \vee X_{i_2}$. Note that, if R is an NP relation, then R_f is also an NP relation under the assumption that the number of leaves of $\mathcal{T}_f$ is bounded by $\ell(|x|)$ The corresponding language is

$$L_f \overset{\text{def}}{=} \{x \in (\{0,1\}^*)^{a(f)}; \exists w \in (\{0,1\}^*)^{a(f)}, (x, w) \in R_f\}.$$

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

Fig. 1. Overview of our procedure of the Σ-protocol Σ_f on the relation 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: $\Sigma_f = (\Sigma_f^{\text{eval}}, \Sigma_f^1, \Sigma_f^2, \Sigma_f^3, \Sigma_f^{\text{vrfy}}, \Sigma_f^{\text{ke}}, \Sigma_f^{\text{sim}})$. In our prover algorithm $\mathbf{P}$, there are four PPT subroutines Σ_f^{eval}, Σ_f^1, Σ_f^3 and Σ_f^{sim}. 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 expression of a boolean formula 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 $v_n = v_{n_{\text{L}}} \wedge v_{n_{\text{R}}}$ or $v_n = v_{n_{\text{L}}} \vee v_{n_{\text{L}}}$ according to AND/OR evaluation of two labels of its two children, n_{L} and n_{R}. The computation is executed for every node from the root to each leaf, recursively, as in Fig. 2.

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

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

Commitment. The prover $\mathbf{P}$ computes a commitment 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 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 strings, $(\textsc{Cha}_n)_n$ and $(\textsc{Res}_l)_l$, are needed for the simulation. Note that the distinguished symbol $\star$ is used to indicate "it is under computation". **P** sends $(\textsc{Cmt}_l)_l$ to **V**.

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

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

Challenge. The verifier **V** computes a challenge $\textsc{Cha}$ by running the algorithm Σ^2_f described in Fig. 4. **V** sends $\textsc{Cha}$ to **P**.

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

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

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

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^1_f. 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^1_f assures the case of height $k + 1$. □

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

We construct a knowledge extractor Σ^ke_f by employing the knowledge extractor Σ^ke of the underlying Σ-protocol Σ as in Fig. 7. Then Lemma 1 assures the above proposition.

Lemma 1 (Knowledge Extraction) *The string $\hat{w}$ output by Σ^{ke}_f 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 $\textsc{Cha} \neq \textsc{Cha}'$.

```
$\Sigma_f^3(x, w, \mathcal{T}, (v_n)_n, (\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$ :
 $\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
 If      $r(\mathcal{T})$ is $\wedge$-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 $\vee$-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 $\wedge$-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 $\vee$-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.

```
$\Sigma_f^{\mathrm{ke}}(x, f, (\textsc{Cmt}_l)_l,\quad \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l,\quad \textsc{Cha}', (\textsc{Cha}'_n)_n, (\textsc{Res}'_l)_l)$ :
 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 $\hat{w}_{\rho(l)} \in_R \{0,1\}^{\ell(|x_{\rho(l)}|)}$
     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 ($\hat{w} := (\hat{w}_{i_j})_{1 \le j \le a(f)}$)
```

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

Suppose that all nodes in 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. □

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 $x \in L_f$ and a predicate f returns an accepting transcript $((\textsc{Cmt}_l)_l, \textsc{Cha}, (\textsc{Cha}_n)_n, (\textsc{Res}_l)_l)$, as in Fig. 8.

```
$\Sigma_f^{\mathrm{sim}}(x, f)$ :
  $\tilde{\textsc{Cha}} \leftarrow \Sigma_f^2(1^\lambda)$, $w \in_R \{0,1\}^{\ell(|x_{\rho(l)}|)}$, For $n \in \mathrm{Node}(\mathcal{T}_f) : v_n := 0$
  $((\tilde{\textsc{Cmt}}_l)_l, (\tilde{\textsc{Cha}}_n)_n, (\tilde{\textsc{Res}}_l)_l) \leftarrow \Sigma_f^1(x, w, \mathcal{T}_f, (v_n)_n, \tilde{\textsc{Cha}})$
  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.

□

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

Theorem 1 (Σ_f is a Σ-protocol) *If a given protocol Σ on a relation R is a Σ-protocol, and if an access formula f is monotone and the number of leaves of $\mathcal{T}_f$ is bounded by $\ell(|x|)$, then the protocol Σ_f with our procedure is a Σ-protocol on the relation R_f.*

Theorem 2 (Σ_f is a perfectly WIPoK system) *If a given protocol Σ on a relation R is a Σ-protocol, and if an access formula f is monotone and the number of leaves of $\mathcal{T}_f$ is bounded by $\ell(|x|)$, then the protocol Σ_f with our procedure is the protocol of a perfectly witness-indistinguishable proof of knowledge system on the relation R_f.*

Proof. For any statement x and any two witnesses w_1 and w_2 satisfying $R_f(x, w_1) = R_f(x, w_2) = 1$, the distribution of the transcript $\mathbf{P}(x, w_1)$ and $\mathbf{V}(x)$ of Σ_f and the distribution of the transcript $\mathbf{P}(x, w_2)$ and $\mathbf{V}(x)$ of Σ_f are identical. □

3.2 Non-interactive Version

The Fiat-Shamir transform 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 a non-interactive perfectly WIPoK system) *If a given protocol Σ on a relation R is a Σ-protocol, and if an access formula f is monotone and the number of leaves of $\mathcal{T}_f$ is bounded by $\ell(|x|)$, then the protocol FS(Σ_f) is the protocol 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 [AAHI13]. Also, the non-interactive version FS(Σ_f) can be considered a proto-type of an attribute-based signature scheme [MPR11]. That is, Σ_f and FS(Σ_f) are an attribute-based identification scheme and an attribute-based signature scheme *without the collusion resistance against collecting private secret keys*, respectively.

4 Conclusion

We provided a concrete procedure of a Σ-protocol Σ_f, which is of a perfectly witness-indistinguishable proof of knowledge system on an NP relation R_f, where f is an input monotone predicate. Our concrete procedure is for any monotone predicate f on condition that the number of leaves of $\mathcal{T}_f$ is bounded by $\ell(|x|)$, and it serves as building blocks of the Σ-protocols in the pairing-free ABS schemes of [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.

[AAHI13] Hiroaki Anada, Seiko Arita, Sari Handa, and Yosuke Iwabuchi. Attribute-based identification: Definitions and efficient constructions. In *Information Security and Privacy - 18th Australasian Conference, ACISP 2013, Brisbane, Australia, July 1-3, 2013. Proceedings*, pages 168–186, 2013.

[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.

[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.

[Gol01] Oded Goldreich. *The Foundations of Cryptography - Volume 1, Basic Techniques.* Cambridge University Press, 2001.

[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.