A Concrete Procedure of the $\Sigma$-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

Jan 11, 2018

Abstract. We propose a concrete procedure of the $\Sigma$-protocol proposed 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.

1 Introduction

A $\Sigma$-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 $\Sigma$-protocol is secure against active and concurrent attacks via a reduction to a number-theoretic assumption [BP02]. Instantiations of the $\Sigma$-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 $\Sigma$-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 $\Sigma$-protocol derived from an original $\Sigma$-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 proposed; suppose a prover and a verifier are given a monotone predicate $f$ over boolean variables. Here a

---

* 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.

4 In the related version [AAS14] of this paper, the authors could not refer to this previous work. Now we would like to refer.
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 \cite{CDS94} a high-level construction that employed a “semi-smooth” secret-sharing scheme was given. (As is stated in \cite{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 \( \Sigma \)-protocol \( \Sigma \), and derive a \( \Sigma \)-protocol \( \Sigma_f \) for any monotone predicate \( f \). Then we show that our \( \Sigma_f \) is actually a \( \Sigma \)-protocol with witness-indistinguishability.

Herranz \cite{Her14} provided the first attribute-based signature scheme (ABS) with both collusion resistance (against collecting private secret keys) and computational attribute privacy 

\[
\text{without pairings (pairing-free)}
\]

in the RSA setting. Recently, Herranz \cite{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 \cite{Her14,Her16a}, the concrete procedures were described in details for threshold-type access formulas. Our concrete procedure \( \Sigma_f \) of the \( \Sigma \)-protocol in \cite{CDS94} for any monotone predicate serves as building blocks of those \( \Sigma \)-protocols in the pairing-free ABS schemes \cite{Her14,Her16a}.

1.2 Our Construction Idea

To provide a concrete procedure for the above protocol \( \Sigma_f \) with witness-indistinguishability, we look into the technique employed in the OR-proof \cite{CDS94} and expand it so that it can treat any monotone predicate, as follows. First express the boolean predicate \( f \) as a binary tree \( T_f \). That is, we put leaves each of which corresponds to each position of a variable in \( f \). We connect two leaves by an \( \land \)-node or an \( \lor \)-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 \( \land \)-node or an \( \lor \)-node in the same way, until we reach to the root node (which is also an \( \land \)-node or an \( \lor \)-node). A verification equation of the \( \Sigma \)-protocol \( \Sigma \) is assigned to every leaf. If a challenge string \( \text{CH}_a \) of \( \Sigma \) is given, then the prover assigns the string \( \text{CH}_a \) to the root node. If the root node is an \( \land \)-node, then the prover assigns the same string \( \text{CH}_a \) to two children. Else if the root node is an \( \lor \)-node, then the prover divides \( \text{CH}_a \) into two random strings \( \text{CH}_L \) and \( \text{CH}_R \) under the constraint that \( \text{CH}_a = \text{CH}_L \oplus \text{CH}_R \), and assigns \( \text{CH}_L \) and \( \text{CH}_R \) 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 \( \Sigma \)-protocol \( \Sigma \) or execute the simulator of \( \Sigma \). Only when a set of witnesses satisfies the binary tree \( T_f \), 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 \( \Sigma \)-protocol \( \Sigma_f \). In Section 4, we conclude our work in this paper.

2 Preliminaries

The security parameter is denoted by \( \lambda \). The bit length of a string \( a \) is denoted by \( |a| \). The concatenation of a string \( a \) with a string \( b \) is denoted by \( a \| b \). A uniform random sampling of an element \( a \) from a set \( S \) is denoted by \( a \in_R S \). The expression \( a = a \) 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) 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 \( z \leftarrow (A(a), B(b)) \). 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 $A^{O \cup \{ \text{in} \}}$. 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 $E_1, \ldots, E_m$ occur in this order is denoted as $\Pr[E_1, \ldots, E_m : E]$.

2.1 Witness-Indistinguishable Proof System and $\Sigma$-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 $|w| \leq \ell(|x|)$ 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 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. 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 $\Pi = (P, V)$ on the relation $R$ with the following property holds.

Witness-Indistinguishability. For any ppt algorithm $A$, and for $x \in L$ and $w_0, w_1 \in W(x)$ of $A$’s choice, we have the following computational indistinguishability.

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

The proof system $\Pi$ 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 $A$, then the proof system $\Pi$ is said to be a perfectly witness-indistinguishable proof system.

$\Sigma$-protocol [Cra96,Dam10] Let $R$ be an NP relation. A $\Sigma$-protocol $\Sigma$ on a relation $R$ is a 3-move public-coin protocol of a proof system $\Pi = (P, V)$ [Dam10]. $P$ sends the first message called a commitment $\text{Cmt}$ to $V$, then $V$ sends the second message that is a public random string called a challenge $\text{Cha}$ to $P$, and then $P$ answers with the third message called a response $\text{Res}$ to $V$. Then $V$ applies a decision test on $(x, \text{Cmt}, \text{Cha}, \text{Res})$ to return $1$ (accept) or $0$ (reject). If $V$ accepts, then the triple $(\text{Cmt}, \text{Cha}, \text{Res})$ is said to be an accepting conversation on $x$. Here $\text{Cha}$ is chosen uniformly at random from $\text{ChaSp}(1^\lambda) := \{0,1\}^{\ell(\lambda)}$ with $\ell(\cdot)$ being a super-log function. Moreover, $\Sigma$ is detailed by the following six ppt algorithms, $\Sigma = \{\Sigma_1, \Sigma_2, \Sigma_3, \Sigma_{\text{vrfy}}, \Sigma_{\text{ke}}, \Sigma_{\text{sim}}\}$, $\text{Cmt} \leftarrow \Sigma_1(x, w)$. This is the process of generating the first message $\text{Cmt}$ according to the protocol $\Sigma$ on input $(x, w) \in R$. Similarly we denote $\text{Cha} \leftarrow \Sigma_2(1^\lambda)$, $\text{Res} \leftarrow \Sigma_3(x, w, \text{Cmt}, \text{Cha})$ and $b \leftarrow \Sigma_{\text{vrfy}}(x, \text{Cmt}, \text{Cha}, \text{Res})$. Furthermore, $\Sigma$ must satisfy the following three requirements.

Completeness. A prover $P$ with a witness $w$ makes $V$ accept with probability $1$.

Special Soundness. There is a ppt algorithm called a knowledge extractor $\Sigma_{\text{ke}}$, 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 $\text{Cmt}$ and with different challenge messages $\text{Cha} \neq \text{Cha}'$; $(\text{Cmt}, \text{Cha}, \text{Res})$ and $(\text{Cmt}, \text{Cha}', \text{Res}')$.

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

Honest-Verifier Zero-Knowledge. There is a ppt algorithm called a simulator $\Sigma_{\text{sim}}$ such that $(\text{Cmt}, \text{Cha}, \text{Res}) \leftarrow \Sigma_{\text{sim}}(x)$, where the distribution of $\{(\text{Cmt}, \text{Cha}, \text{Res})\}$ is the same as the distribution $\{(\text{Cmt}, \text{Cha}, \text{Res})\}$ generated as real transcripts of accepting conversations between $P(x, w)$ and $V(x)$. Another equivalent variant of the simulator is constructed by using the fact that the challenge message $\text{Cha}$ is a public-coin. That is, $\text{Cha}$ is generated by running $\Sigma_3(1^\lambda)$ (i.e. uniform random sampling from $\text{ChaSp}(1^\lambda)$), then it is input to (another variant of) the simulator to generate the commitment message $\text{Cmt}$ and the response message $\text{Res}$:

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

A proof system $\Pi = (P, V)$ with a $\Sigma$-protocol is known to be a proof of knowledge system [BG92].
The OR-proof [Dam10] For a boolean predicate $f(X_1, X_2) = X_1 \lor X_2$, we consider a $\Sigma$-protocol $\Sigma_{OR}$ on a relation $R_{OR}$ below.

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

The corresponding language is

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

Suppose that a $\Sigma$-protocol $\Sigma$ on a relation $R$ is given. Then we can construct the protocol $\Sigma_{OR}$ on the relation $R_{OR}$ as follows. For instance, suppose $(x_0, w_0) \in R$ holds. Then $P$ computes $\text{CMT}_0 \leftarrow \Sigma^1(x_0, w_0), \text{CH}_A \leftarrow \Sigma^2(1^\lambda), (\text{CMT}_1, \text{RES}_1) \leftarrow \Sigma^\Xi(x_1, \text{CH}_A)$ and sends $(\text{CMT}_0, \text{CH}_A)$ to $V$. Then $V$ sends $\text{CH}_A \leftarrow \Sigma^2(1^\lambda)$ to $P$. Then $P$ computes $\text{CH}_A := \text{CH}_A \oplus \text{CH}_A, \text{RES}_0 \leftarrow \Sigma^\Xi(x_0, w_0, \text{CMT}_0, \text{CH}_A)$ answers to $V$ with $(\text{CH}_A, \text{CH}_A)$ and $(\text{RES}_0, \text{RES}_1)$. Here $\oplus$ denotes a bitwise exclusive-OR operation. Then both $(\text{CMT}_0, \text{CH}_A, \text{RES}_0)$ and $(\text{CMT}_1, \text{CH}_A, \text{RES}_1)$ are accepting conversations on $x$ and have the same distribution as real accepting conversations. The protocol $\Sigma_{OR}$ can be proved to be a $\Sigma$-protocol [CDS94,Dam10]. We often call $\Sigma_{OR}$ the OR-proof. A proof system $\Pi$ 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 $U = \{1, \ldots, u\}$ be an attribute universe [GPSW06]. We must distinguish two cases: the case that $U$ is small (that is, $|U| = u$ is bounded by a polynomial in $\lambda$) and the case that $U$ 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 ($\land$) and OR-gate ($\lor$). For example, $f = X_{at_1} \land ((X_{at_2} \land X_{at_3}) \lor X_{at_4})$ for some $at_1, at_2, at_3, at_4, 1 \leq at_1 < at_2 < at_3 < at_4 \leq u$. Note that there is a bijective map between boolean variables and attributes:

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

For $f(X_{i_1}, \ldots, X_{i_a})$, we denote the set of indices (that is, attributes) $\{i_1, \ldots, i_a\}$ by $\text{Att}(f)$. We note the arity of $f$ as $a(f)$. 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 predicate $f$. For $S \in 2^U$, we evaluate the boolean value of $f$ at $S$ as follows:

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

Under this definition, a boolean predicate $f$ can be seen as a map: $f : 2^U \rightarrow \{0, 1\}$. We call a boolean predicate $f$ with this map an access formula over $U$. 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 $T_f$. Each inner node represents a boolean connective, $\land$-gate or $\lor$-gate, in $f$. Each leaf corresponds to a term $X_{at}$ (not a variable $X_{at}$) in $f$ in one-to-one way. For a finite binary tree tree $T$, 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 $\text{Node}(T), r(T), \text{Leaf}(T), \text{iNode}(T)$ and $\text{tNode}(T)$, respectively. Then the attribute map $\rho(\cdot)$ is defined as:

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

If $\rho$ is not injective, then we call the case multi-use of attributes.

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

\(^5\)This limitation can be removed by adding negation attributes to $U$ for each attribute in the original $U$ though the size of the attribute universe $|U|$ doubles.
Our Procedure

In this section, we construct a \( \Sigma \)-protocol \( \Sigma_f \) of a witness-indistinguishable proof of knowledge system from a given \( \Sigma \)-protocol \( \Sigma \) and a monotone predicate \( f \).

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

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

\[
R_f \overset{\text{def}}{=} \{(x = (x_1,\ldots,x_{n_f})), w = (w_1,\ldots,w_{n_{w(f)}}) \in \{0,1\}^n \times \{0,1\}^n, f(R(x_1, w_1), \ldots, R(x_{n_f}, w_{n_{w(f)}})) = 1\}.
\]

\( R_f \) is a generalization of the relation \( R_{OR} \) [CDS94,Dam10], where \( f \) is a boolean predicate with the single boolean connective OR: \( X_1 \lor X_2 \). 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 \( \lambda \). The corresponding language is

\[
L_f \overset{\text{def}}{=} \{x \in \{0,1\}^n : \exists w, (x,w) \in R_f\}.
\]

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

**Evaluation of Satisfiability.** The prover \( P \) begins with evaluation of whether and how \( S \) satisfies \( f \) by running the evaluation algorithm \( \Sigma_f^{val} \). It labels each node of \( T_f \) with a value \( v \) with 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 \( \rho \), see Section 2.2.) For each inner node \( n \), we label \( n \) with \( v_n = v_{nL} \land v_{nR} \) or \( v_n = v_{nL} \lor v_{nR} \) according to AND/OR evaluation of two labels of its two children \( n_L, n_R \). The computation is executed for every node from the root to each leaf, recursively, as in Fig. 2.

**Commitment.** The prover \( P \) computes a commitment value for each leaf by running the algorithm \( \Sigma_f^1 \) described in Fig. 3. Basically, \( \Sigma_f^1 \) runs for every node from the root to each leaf, recursively. As a result, \( \Sigma_f^1 \) generates for
The verifier $\Sigma$. Basically, the algorithm

Response. The prover $\Sigma$ following algorithm

Verification. The verifier $\Sigma$

Proposition 1 (Completeness) The completeness holds for our $\Sigma_f$.

Proof. Suppose that $v_{r(T_f)} = 1$. We show that, for every node in $\text{Node}(T_f)$, either $v_n = 1$ or $\text{CHA}_n \neq \star$ holds after executing $\Sigma^1_f$. The proof is by induction on the height of $T_f$. The case of height 0 follows from $v_{r(T_f)} = 1$ and the completeness of $\Sigma$. Suppose that the case of height $k$ holds and consider the case of height $k + 1$. The construction of $\Sigma^1_f$ assures the case of height $k + 1$. □
\[ \Sigma_f^3(x, w, T, (v_n)_n, (CMT))_l, Cha, (Cha_n)_n, (Res_i)_l) : \]
\[ \mathcal{T}_l := \text{Lsub}(T), \mathcal{T}_R := \text{Rsub}(T) \]
If \( \mathcal{r}(\mathcal{T}) \) is an \&-node \( n \), then \( \text{Cha}_n := \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} := \text{Cha}_{\mathcal{r}(\mathcal{T}_R)} := \text{Cha} \)
Return(\( \text{Cha}_n, \Sigma_f^3(x, w, \mathcal{T}_l, (v_n)_n, (CMT))_l, (Cha_{\mathcal{r}(\mathcal{T}_l)})_l, (Cha_{\mathcal{r}(\mathcal{T}_R)})_l, (Res_i)_l) \),
\[ \Sigma_f^3(x, w, \mathcal{T}_R, (v_n)_n, (CMT))_l, (Cha_{\mathcal{r}(\mathcal{T}_l)})_l, (Cha_{\mathcal{r}(\mathcal{T}_R)})_l, (Res_i)_l) \)
else if \( \mathcal{r}(\mathcal{T}) \) is an \&-node \( n \), then \( \text{Cha}_n := \text{Cha} \)
If \( v_{\mathcal{r}(\mathcal{T}_l)} = 1 \) and \( v_{\mathcal{r}(\mathcal{T}_R)} = 1 \), then \( \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} := \Sigma_f^3(1) \), \( \text{Cha}_{\mathcal{r}(\mathcal{T}_R)} := \text{Cha} \bigoplus \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} \)
else if \( v_{\mathcal{r}(\mathcal{T}_l)} = 1 \) and \( v_{\mathcal{r}(\mathcal{T}_R)} = 0 \), then \( \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} := \text{Cha} \bigoplus \text{Cha}_{\mathcal{r}(\mathcal{T}_R)} \)
else if \( v_{\mathcal{r}(\mathcal{T}_l)} = 0 \) and \( v_{\mathcal{r}(\mathcal{T}_R)} = 1 \), then \( \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} := \text{Cha} \bigoplus \text{Cha}_{\mathcal{r}(\mathcal{T}_R)} \)
else if \( v_{\mathcal{r}(\mathcal{T}_l)} = 0 \) and \( v_{\mathcal{r}(\mathcal{T}_R)} = 0 \), then do nothing
Return(\( \text{Cha}_n, \Sigma_f^3(x, w, \mathcal{T}_l, (v_n)_n, (CMT))_l, (Cha_{\mathcal{r}(\mathcal{T}_l)})_l, (Cha_{\mathcal{r}(\mathcal{T}_R)})_l, (Res_i)_l) \),
\[ \Sigma_f^3(x, w, \mathcal{T}_R, (v_n)_n, (CMT))_l, (Cha_{\mathcal{r}(\mathcal{T}_l)})_l, (Cha_{\mathcal{r}(\mathcal{T}_R)})_l, (Res_i)_l) \)
else if \( \mathcal{r}(\mathcal{T}) \) is a leaf \( l \), then \( \text{Cha}_n := \text{Cha} \)
If \( v_l = 1 \), then \( \text{Res}_l \leftarrow \Sigma_f^3(x_{\rho(l)}, w_{\rho(l)}, CMT, Cha) \)
else if \( v_l = 0 \), then do nothing
Return(\( \text{Cha}_n, \text{Res}_l \))

Fig. 5. The subroutine \( \Sigma_f^3 \) of our \( \Sigma_f \).

\[ \Sigma_f^{\text{vrfy}}(x, \mathcal{T}, (CMT))_l, Cha, (Cha_n)_n, (Res_i)_l) : \]
Return(\( \text{VrfyCha}(\mathcal{T}, Cha, (Cha_n)_n) \wedge VrfyRes(x, \mathcal{T}, (CMT))_l, (Cha_l)_l, (Res_i)_l) \))

\[ \text{VrfyCha}(\mathcal{T}, Cha, (Cha_n)_n) : \]
\[ \mathcal{T}_l := \text{Lsub}(\mathcal{T}), \mathcal{T}_R := \text{Rsub}(\mathcal{T}) \]
If \( \mathcal{r}(\mathcal{T}) \) is an \&-node \( n \), then
Return \(( (\text{Cha} = ? \text{Cha}_{\mathcal{r}(\mathcal{T}_l)}) \wedge (\text{Cha} = ? \text{Cha}_{\mathcal{r}(\mathcal{T}_R)}) \) 
\[ \wedge VrfyCha(\mathcal{T}_l, Cha_{\mathcal{r}(\mathcal{T}_l)}), (Cha_n)_n) \wedge VrfyCha(\mathcal{T}_R, Cha_{\mathcal{r}(\mathcal{T}_R)}), (Cha_n)_n) \))
else if \( \mathcal{r}(\mathcal{T}) \) is an \&-node \( n \), then
Return \(( (\text{Cha} = ? \text{Cha}_{\mathcal{r}(\mathcal{T}_l)} \bigoplus \text{Cha}_{\mathcal{r}(\mathcal{T}_R)}) \) 
\[ \wedge VrfyCha(\mathcal{T}_l, Cha_{\mathcal{r}(\mathcal{T}_l)}), (Cha_n)_n) \wedge VrfyCha(\mathcal{T}_R, Cha_{\mathcal{r}(\mathcal{T}_R)}), (Cha_n)_n) \))
else if \( \mathcal{r}(\mathcal{T}) \) is a leaf \( l \), then
Return \(( \text{Cha} \in \text{ChA}, \text{ChA}SP(1) \))

\[ \text{VrfyRes}(x, \mathcal{T}, (CMT)_l, (Cha_l)_l, (Res_i)_l) : \]
For \( l \in \text{Leaf}(\mathcal{T}) \) : If \( \Sigma_f^{\text{vrfy}}(x_{\rho(l)}, CMT_l, Cha_l, Res_l) = 0 \), then Return \(( 0 ) \)
Return \(( 1 ) \)

Fig. 6. The subroutine \( \Sigma_f^{\text{vrfy}} \) of our \( \Sigma_f \).
Lemma 1 (Witness Extraction) The string \( \hat{w} \) output by \( \Sigma_f^{ke} \) satisfies \( (x, \hat{w}) \in R_f \).

**Proof.** We prove the lemma by induction on the number of all \( \lor \)-nodes in \( \text{iNode}(T_f) \). First remark that \( \text{CH}_A \neq \text{CH}_A' \).

Suppose that all nodes in \( \text{iNode}(T_f) \) are \( \land \)-nodes. Then the above claim follows immediately because \( \text{CH}_A \neq \text{CH}_A' \) holds for all leaves.

Suppose that the case of \( k \) \( \lor \)-nodes holds and consider the case of \( k+1 \) \( \lor \)-nodes. Look at one of the lowest height \( \lor \)-node and name the height and the node as \( h^* \) and \( n^* \), respectively. Then \( \text{CH}_{A,n^*} \neq \text{CH}_{A',n^*} \) because all nodes with their heights less than \( h^* \) are \( \land \)-nodes. So at least one of children of \( n^* \), say \( n^*_j \), satisfies \( \text{CH}_{A,n^*_j} \neq \text{CH}_{A',n^*_j} \).

Divide the tree \( T_f \) into two subtrees by cutting the branch right above \( n^* \), and the induction hypothesis assures the claim. \( \square \)

Proposition 2 (Special Soundness) The honest-verifier zero-knowledge property holds for our \( \Sigma_f \).

**Proof.** We construct a polynomial-time simulator \( \Sigma_f^{sim} \), which on input a statement \( xinL_f \) and a predicate \( f \) returns an accepting conversation \( ((\text{CMT}_l), \text{CH}_A, (\text{CH}_A)_n, (\text{RES}_l)_l) \), as follows.

\[
\Sigma_f^{sim}(x, f) : \\
\text{CH}_A \leftarrow \Sigma_f^{sim}(1^\lambda), w \in_R \{0,1\}^{f(w)(\lambda)}, \text{For } n \in \text{Node}(T_f) : v_n := 0 \\
((\text{CMT}_l), (\text{CH}_A)_n, (\text{RES}_l)_l) \leftarrow \Sigma_f^{sim}(x, w, T_f, (v_n)_n, \text{CH}_A) \\
\text{Return}((\text{CMT}_l), \text{CH}_A, (\text{CH}_A)_n, (\text{RES}_l)_l))
\]

**Fig. 8.** The simulator \( \Sigma_f^{sim} \) of our \( \Sigma_f \). \( \square \)

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

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

**Theorem 2 (\( \Sigma_f \) is WIPoK)** Our \( \Sigma \)-protocol \( \Sigma_f \) 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 \( R(x, w_1) = R(x, w_2) = 0 \), \( P(x, w) \) and \( V(x) \) of \( \Sigma_f \) generate transcripts \( ((\text{CMT}_l), \text{CH}_A, (\text{CH}_A)_n, (\text{RES}_l)_l) \) that have the same distribution. \( \square \)
3.2 Non-interactive Version

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

**Theorem 3 (FS($\Sigma_f$) is NIWIPoK)** Our $FS(\Sigma_f)$ 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 $\Sigma$-protocol $\Sigma_f$ can be considered as a proto-type of an attribute-based identification scheme. Also, the non-interactive version $FS(\Sigma_f)$ can be considered a proto-type of an attribute-based signature scheme. That is, $\Sigma_f$ and $FS(\Sigma_f)$ are an attribute-based identification scheme and an attribute-based signature scheme without collusion resistance on collecting private secret keys, respectively.

4 Conclusion

We provided a concrete procedure $\Sigma_f$ of a $\Sigma$-protocol of the WIPoK system on monotone predicates. Our $\Sigma_f$ can be considered as a proto-type of an attribute-based identification scheme, and also, $FS(\Sigma_f)$ can be considered a proto-type of an attribute-based signature scheme [CDS94], without collusion resistance on private secret keys. Our procedure $\Sigma_f$ for any monotone predicate serves as building blocks of the $\Sigma$-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 $\Sigma$-protocol on monotone predicates.

**References**


---

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].


