Attribute-Based Signatures without Pairings by the Fiat-Shamir Transform
(Corrected Version)*

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

[1] Institute of Systems, Information Technologies and Nanotechnologies
anada@isit.or.jp
[2] Institute of Information Security
arita@iisec.ac.jp
[3] Department of Informatics, Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University
sakurai@inf.kyushu-u.ac.jp

September 24, 2015

Abstract. We propose an attribute-based signature scheme (ABS) with features of pairing-free, short signatures and security proof in the random oracle model. Our strategy is in the Fiat-Shamir paradigm; we first provide a concrete procedure of the Σ-protocol which enables a prover to prove possession of witnesses that satisfy a statement of a monotone boolean formula. Next, using a signature bundle scheme of the Fiat-Shamir signature as those witnesses in the Σ-protocol, we obtain a generic attribute-based identification scheme (ABID). Then, we apply the Fiat-Shamir transform to our ABID to obtain a scheme of ABS. The series of these generic constructions are obtained from a given Σ-protocol. Finally, we provide our ABID and ABS schemes concretely in the Discrete-Logarithm setting and the RSA setting. These concretions are pairing-free. Signatures of our ABS are linkable, hence attribute privacy does not hold; it holds only as a one-time signature.

Keywords: digital signature, access structure, attribute, boolean formula, proof system, Fiat-Shamir transform

1 Introduction

Since the invention of digital signature scheme by Diffie and Hellman in 1976, there have been significant evolutions in the area and many functional variants have been proposed. One of distinguished variant is *attribute-based signature scheme* (ABS), which has been developed since 2008 [22, 34, 29, 28, 30, 14, 31, 17, 26, 32, 23, 13, 12, 13, 18, 24]. In ABS, a message is associated with an access structure by which a signer can make a signature on it only when his attributes satisfy the access structure. The access structure is described as a boolean formula over their attributes, which is called access formula. A signer with a set of authorized attributes can make a legitimate signature on the message only when his attributes satisfy the access formula. Then, a verifier checks whether the pair of the message and the signature is valid or not in accordance with the

* The preliminary version of this paper appeared in *Proceedings in the 2nd ACM ASIA Public-Key Cryptography Workshop - ASIAPKC 2014*, pp. 49-58 under the title of *Attribute-Based Signatures without Pairings via the Fiat-Shamir Paradigm*. This is the full version. The statement on attribute privacy was corrected because our scheme does not possess the property. The first and the second authors are partially supported by a *kakenhi* Grant-in-Aid for Scientific Research (C) 15K00029 from Japan Society for the Promotion of Science. The third author is partially supported by a *kakenhi* Grant-in-Aid for Scientific Research (C) 15H02711 from Japan Society for the Promotion of Science.

access formula. If the access formula is given as a single identity, ABS coincides with identity-based signature scheme (IBS) [35]. In that sense, ABS is a natural extension of IBS.

To obtain such an ABS scheme, we work in the Fiat-Shamir paradigm [16] in this paper. The Fiat-Shamir transform [16, 1] is now an established technique in interactive proof theory and cryptography. By replacing the message of the second move with the hash value of the message of the first move, it transforms a public-coin 3-move interactive proof system into a non-interactive zero-knowledge proof system (NIZK). If the initial interactive proof system is an identification scheme and if an input message is concatenated with the message of the first move, then the resulting NIZK turns into a digital signature scheme. One of such interactive proof systems is the Σ-protocol [9, 11]. It allows a prover, in a 3-move, to convince a verifier that the prover knows a witness of a statement. That is, the Σ-protocol is a public-coin 3-move interactive proof of knowledge system. We can apply the Fiat-Shamir transform to the Σ-protocol and obtain a zero-knowledge proof of knowledge system (NIZKPoK) whose extractor is in the random oracle model. Another feature of the Σ-protocol is that we can construct the OR-proof protocol [10, 11] from a Σ-protocol. *Hiding which witness he knows*, it allows a prover to convince a verifier that the prover knows at least one of two witnesses of two statements, respectively.

In the Fiat-Shamir paradigm we first look into the protocol introduced by Cramer, Damgård and Schoenmakers [10][4] in CRYPTO '94. Given a boolean formula f, the protocol provides a witness hiding protocol [15]. That is, in f, a value '1' (TRUE) is substituted for boolean variables at which a prover has the corresponding witnesses; a value '0' (FALSE) is substituted for the remaining boolean variables. Then the prover in the protocol can show to a verifier that he knows a set of witnesses that satisfies the boolean formula f (that is, a set of witnesses that makes a whole evaluation of f TRUE), but a cheating verifier does not learn anything that is enough to compute a satisfying set of witnesses. We call this notion *boolean proof*. In [10], the notion of the boolean proof was provided at a high level, and the OR-proof , which was a special case where f is a single "OR", was explained with a concrete procedure. In this paper, we give a concrete procedure, $\mathbf{\Sigma}_f$, for a generic monotone boolean formula f. (Here a monotone boolean formula means a boolean formula without negation.) Besides, we will show that $\mathbf{\Sigma}_f$ is also a Σ-protocol.

Then we will proceed further into the Fiat-Shamir paradigm. By using a signature bundle scheme from the Fiat-Shamir signature as those witnesses of our boolean proof system, we obtain a generic attribute-based identification scheme (ABID). Finally, we apply the Fiat-Shamir transform to our ABID to obtain a generic ABS which can be proved to be existentially unforgeable against chosen-message attacks in the random oracle model. A remarkable property is that the series of constructions are obtained from a given single Σ-protocol, and our ABS can be constructed without pairing maps ([7]) (pairings, for short).

1.1 Our Construction Idea of Concrete Procedure of Boolean Proof

To provide a concrete procedure for the above boolean proof system from a given Σ-protocol and a monotone boolean formula f, we look into the technique employed in the OR-proof [10] and expand it so that it can treat any monotone boolean formula, as follows.

First express the boolean formula f as a binary tree $\mathcal{T}_f$. That is, we put leaf nodes each of which corresponds to each term in f. We connect two leaf nodes by an $\wedge$-node or an $\vee$-node according to an AND-gate or an OR-gate that is between two corresponding terms in f. Then

[4] In the preliminary version [3] the authors could not refer to this previous work. Now we refer this work and state the relation.

we connect the resulting nodes by an $\wedge$-node or an $\vee$-node in the same way, until we reach to the only $\wedge$-node or $\vee$-node called the root node. A verification equation of the Σ-protocol $\mathbf{\Sigma}$ is assigned to every leaf node. If a challenge string CHA of $\mathbf{\Sigma}$ is given, then assign the string CHA to the root node. If the root node is an $\wedge$-node, then assign the same string CHA to two children. Else if the root node is an $\vee$-node, then divide CHA into two random strings $\mathrm{CHA_L}$ and $\mathrm{CHA_R}$ under the constraint that $\mathrm{CHA} = \mathrm{CHA_L} \oplus \mathrm{CHA_R}$, and assign $\mathrm{CHA_L}$ and $\mathrm{CHA_R}$ to the left child and the right child, respectively. Here $\oplus$ means a bitwise exclusive-OR operation. Then continue to apply this rule at each height, step by step, until we reach to every leaf node. Then, basically, the OR-proof technique assures that we can either honestly execute the Σ-protocol $\mathbf{\Sigma}$ or execute the simulator of $\mathbf{\Sigma}$. Only when a set of witnesses satisfies the binary tree $\mathcal{T}_f$, the above procedure succeeds in satisfying verification equations for all leaf nodes.

1.2 Our Contributions

Our first contribution is to provide a concrete procedure of the boolean proof system [10]. That is, given a Σ-protocol $\mathbf{\Sigma}$ and a monotone boolean formula f, we construct a concrete procedure $\mathbf{\Sigma}_f$ in a recursive form that is suitable for implementation. Then we show that $\mathbf{\Sigma}_f$ is certainly a Σ-protocol. Especially we show that $\mathbf{\Sigma}_f$ is a protocol to prove knowledge of witnesses that satisfy the boolean formula f.

Our second contribution is to provide an attribute-based signature scheme (ABS) without pairings. We provide schemes of ABID and ABS concretely in both the Discrete-Logarithm setting and the RSA setting by employing the Schnorr identification scheme and the GQ identification scheme [33, 5] as $\mathbf{\Sigma}$, respectively. These constructions are pairing-free. We note that signatures of our ABS are linkable, and attribute privacy holds only as a one-time signature. Moreover, the reduction of advantages of adversaries to the corresponding number theoretic hardnesses is loose.

1.3 Related Work

The Σ-protocol was formalized in the Cramer's Ph.D thesis [9]. The Σ-protocol of OR-proof was described in Cramer, Damgård and Schoenmakers [10] (see also [11]). The Σ-protocol of their boolean proof system, an extension of the OR-proof system to any monotone boolean formula, was also discussed in the work [10] in an abstract manner. We provide a concrete procedure of the boolean proof system.

At a high level, our ABS is captured as a result of the Fiat-Shamir transform of our boolean proof system in which a set of witnesses is a signature bundle (credential bundle [30]). This construction can be compared with the generic construction of a scheme of ABS by Maji et al. [30]. They started with a signature bundle (of Boneh-Boyen signatures [8], for instance). Then they employed a non-interactive witness-indistinguishable proof of knowledge system (NIWIPoK) of Groth and Sahai [21] to prove the knowledge of a signature bundle which satisfies a given (monotone) access formula, in the standard model.

Okamoto and Takashima (OT11) [31] gave a scheme of ABS with full-security; security against adaptive target in the standard model under a non-q-type assumption. It can treat non-monotone access formula and multi-use of attributes, and possesses attribute privacy in the information-theoretic sense. The construction is based on their Dual Pairing Vector Space.

Herranz [23] provided the first ABS with both collusion resistance (against collecting private secret keys) and (computationally secure) attribute privacy without pairings in the RSA setting.

In the work [23], the concrete procedure was described for monotone access formulas. Our ABS can also be constructed without pairings and provide a concrete procedure for monotone access formulas, but does not achieve attribute privacy.

Ghadafi et al. [12] proposed the functionality of "User-Controlled Linkability" (UCL) in the case of attribute-based signatures. UCL property in the work [12] can be captured as a kind of public linkability. In general, public linkability is achieved with the expense of loosing attribute privacy in ABS, and hence the scheme [12] and our scheme do not possess attribute privacy. It is notable that the scheme [12] uses pairings and can be set up in the multi-authorities setting [32, 13, 18].

More on Attribute Privacy Intuitively, ABS is called to have attribute privacy if any cheating verifier cannot distinguish two distributions of signatures each of which is generated by different satisfying attribute set. Originally, attribute privacy was introduced as information theoretic security [30, 31]. That is, even an adversary with unconditional computational power should not distinguish two distributions. It is notable that in the work of Herranz [23], attribute privacy was expanded to broader sense; that is, as computational security. There, an adversary with polynomial-time computational power should not distinguish two distributions.

Our ABS does not have attribute privacy even in this broader sense because of its linkability. The linkability is caused by our construction that a part of a private secret key is included in signatures [25][5]. Hence, our ABS merely has attribute privacy *as a one-time signature scheme*.

1.4 Technical and Efficiency Comparison: Security, Functionality and Signature Length

We compare our scheme with the above previously proposed schemes from the view point of security, functionality and signature length. The comparison is summarized in Table 1 with notations as follows. A prime of bit length λ (the security parameter) is denoted by p. Though a pairing map e should be analysed for the asymmetric bilinear groups [20], we simply evaluate for the symmetric case in which both source groups are $\mathbb{G}_p$ of order p. We assume that an element of $\mathbb{G}_p$ is represented by 2λ bits. l and r mean the number of rows and columns of the share-generating matrix for monotone access formula f (that is, an access structure), respectively. CR means the collision resistance of an employed hash function. q-SDH means the Strong Diffie-Hellman assumption with q-type input for bilinear groups [7]. DLIN means the Decisional Linear assumption for bilinear groups [31]. DDH means the Decisional Diffie-Hellman assumption for a cyclic group [12]. DLog means the Discrete Logarithm assumption for a cyclic group [12]. q-SRSA means the strong RSA assumption with q-type input [23]. DDH in $QR(N)$ means the Decisional Diffie-Hellman assumption for quadratic residues modulo N (the RSA modulus) [23]. RSAInv means the RSA Inversion assumption for integers modulo N [5]. (info.) means information-theoretic security and (comp.) means computational security.

The scheme of [31] has advantages in the security-proof model, access formula and information-theoretically secure attribute privacy, whereas our ABS realizes shorter length of signature (less than a half). The scheme of [23] is in the RSA setting and its security parameter λ_{rsa} is almost 10 times longer than λ in the DLog setting. For example, $\lambda_{\mathrm{rsa}} = 2048$ is almost equivalent to $\lambda = 224$-bit security [36]. (θ is the threshold value of the threshold-type access structure. κ is

[5] We thank to Javier Herranz for his sincere comment for our error on the statement of attribute privacy in the first version of this ePrint.

Table 1. Comparison of Security, Functionality and Signature Length. λ is the security parameter and l is the number of attributes.

Scheme	Security Model	Assumption	Adaptive Target	Access Formula	Pairing -Free	Attribute Privacy	Length of Signature
Maji et al. [30]	Std.	q-SDH $\wedge$ DLIN	✓	Monotone	-	✓(info.)	$(2\lambda)(51l + 2r + 18\lambda l)$
Okamoto-Takashima [31]	Std.	DLIN $\wedge$ CR	✓	Non-mono.	-	✓(info.)	$(2\lambda)(9l + 11)$
Herranz [23]	R.O.	q-SRSA$\wedge$[DDH in $QR(N)$]$\wedge$CR	✓	Monotone	✓	✓(comp.)	$\lambda_{\mathrm{rsa}}(5 + \frac{\kappa}{\lambda_{\mathrm{rsa}}})l$ $+\lambda_{\mathrm{rsa}}3 - \kappa(\theta - 1)$
Ghadafi et al. [12]	R.O.	q-SDH $\wedge$ DDH $\wedge$ DLog $\wedge$ CR	✓	Monotone	-	-	$(2\lambda)(3l + r + 3) + \lambda(8l + 4)$
Our proposal	R.O.	[DLog ($\vee$RSAInv)] $\wedge$ CR	✓	Monotone	✓	-	$(2\lambda)l + \lambda(4l - 1)$

explained in the work [23].) Therefore, our ABS in the DLog setting realizes shorter length of a signature. Note that the scheme of [23] achieves computationally secure attribute privacy.

1.5 Organization of this Paper

In Section 2, we prepare for the required tools and notions. In Section 3, we describe a concrete procedure of the boolean proof system, $\mathbf{\Sigma}_f$. In Section 4, by using a signature bundle scheme of the Fiat-Shamir signature FS($\mathbf{\Sigma}$) as witnesses of our $\mathbf{\Sigma}_f$, we obtain our ABID. In Section 5, by applying the Fiat-Shamir transform to our ABID, we obtain our ABS. In Section 6, we conclude our work in this paper. In Appendix A, we show concrete instantiations of our ABS in the discrete-logarithm setting and the RSA setting. In Appendix B, we summarize the notion of NIWI proof of knowledge system. In Appendix C, we state a NIWIPoK system that is obtained from our $\mathbf{\Sigma}_f$.

2 Preliminaries

The security parameter is denoted by λ. Bit length of a string x is denoted as $|x|$. 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 $z \leftarrow \langle A(a), B(b)\rangle$. When A has oracle-access to $\mathcal{O}$, we denote it as $A^{\mathcal{O}}$. When A has concurrent oracle-access to n oracles $\mathcal{O}_1, \ldots, \mathcal{O}_n$, we denote it as $A^{\mathcal{O}_i|_{i=1}^n}$. Here "concurrent" means that A accesses to oracles in arbitrarily interleaved order of messages. We denote a concatenation of a string a with a string b as $a \parallel b$. The expression $a \stackrel{?}{=} b$ returns a value 1 (TRUE) when $a = b$ and 0 (FALSE) otherwise. The expression $a \stackrel{?}{\in} S$ returns a value 1 when $a \in S$ and 0 otherwise. 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 Language and Proof of Knowledge [4, 10, 11]

Language Let $R = \{(x, w)\} \subset \{1,0\}^* \times \{1,0\}^*$ be a binary relation. We say that R is polynomially bounded if there exists a polynomial *poly* such that $|w| \leq \mathit{poly}(|x|)$ for all $(x, w) \in R$. If

$(x, w) \in R$ then we call x a statement and w a witness of x. 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 in R.

A *language* for a relation R is defined as:

$$L_R \stackrel{\text{def}}{=} \{x \in \{1,0\}^*; \exists w \in \{1,0\}^*, (x, w) \in R\}.$$

L_R is called a NP language if R is an NP relation. Hereafter, we assume that R is an NP relation.

We introduce a *relation-function* $R(\cdot, \cdot)$ associated with the relation R by:

$$R(\cdot, \cdot) : \{1,0\}^* \times \{1,0\}^* \to \{1,0\},$$
$$(x, w) \mapsto 1 \text{ if } (x, w) \in R,\ 0 \text{ otherwise.}$$

Proof of Knowledge A *proof of knowledge system* (PoK for short) $\Pi = (\mathcal{P}, \mathcal{V})$ for a language L_R is a protocol between interactive PPT algorithms $\mathcal{P}$ and $\mathcal{V}$ on initial input $(x, w) \in R$ for $\mathcal{P}$ and x for $\mathcal{V}$, where $\mathcal{V}$ outputs 1 (accept) or 0 (reject) after finite rounds of interaction. $\mathcal{P}$ is called a prover and $\mathcal{V}$ is called a verifier. In general, a prover $\mathcal{P}$ has unbounded computational power, but in this paper we only consider the case that $\mathcal{P}$ is PPT.

Π must possess the following two properties.

Completeness. For any statement $x \in L_R$ and for any witness w such that $(x, w) \in R$, $\mathcal{P}$ with the witness w can make $\mathcal{V}$ accept for the statement x with probability 1:

$$\Pr[\langle \mathcal{P}(x, w), \mathcal{V}(x) \rangle = 1] = 1.$$

Knowledge Soundness. There are a PPT algorithm $\mathcal{KE}$ called a *knowledge extractor*, a function $\kappa : \{1,0\}^* \to [1,0]$ called a *knowledge error function* and a constant $c > 0$ that satisfy the following:

If there exists a PPT algorithm $\mathcal{A}$ that satisfies $p(x) := \Pr[1 \leftarrow \langle \mathcal{A}(x), \mathcal{V}(x) \rangle] > \kappa(x)$, then $\mathcal{KE}(x)$, employing $\mathcal{A}(x)$ as a subroutine that allows to be rewinded [6], outputs a witness w which satisfies $(x, w) \in R$ within an expected number of steps bounded by: $|x|^c/(p(x) - \kappa(x))$.

Σ-protocol [9, 11] A *Σ-protocol* on a relation R is a public coin 3-move protocol between interactive PPT algorithms $\mathcal{P}$ and $\mathcal{V}$ on initial input $(x, w) \in R$ for $\mathcal{P}$ and x for $\mathcal{V}$. $\mathcal{P}$ sends the first message called a commitment Cmt, then $\mathcal{V}$ sends a random bit string called a challenge Cha, and $\mathcal{P}$ answers with a third message called a response Res. Then $\mathcal{V}$ applies a decision test on $(x, \text{Cmt}, \text{Cha}, \text{Res})$ to return accept (1) or reject (0). If $\mathcal{V}$ accepts, then the triple $(\text{Cmt}, \text{Cha}, \text{Res})$ is said to be an *accepting conversation*. Cha is chosen uniformly at random from $\text{ChaSp}(1^\lambda) := \{1,0\}^{l(\lambda)}$ with $l(\cdot)$ being a super-log function.

This protocol is written by a PPT algorithm $\mathbf{\Sigma}$ as follows. $\text{Cmt} \leftarrow \mathbf{\Sigma}^1(x, w)$: the process of selecting the first message Cmt according to the protocol $\mathbf{\Sigma}$ on input $(x, w) \in R$. Similarly we denote $\text{Cha} \leftarrow \mathbf{\Sigma}^2(1^\lambda)$, $\text{Res} \leftarrow \mathbf{\Sigma}^3(x, w, \text{Cmt}, \text{Cha})$ and $b \leftarrow \mathbf{\Sigma}^{\text{vrfy}}(x, \text{Cmt}, \text{Cha}, \text{Res})$.

Σ-protocol must possess the following three properties.

Completeness. A prover $\mathcal{P}$ with a witness w can make $\mathcal{V}$ accept with probability 1.

Special Soundness. Any PPT algorithm $\mathcal{P}^*$ without any witness, a cheating prover, can only respond for one possible challenge Cha. In other words, there is a PPT algorithm called a

[6] In [4], it is described as "oracle-access to $\mathcal{A}_x$" instead of rewinding, which is more general statement but we do not need the generality in this paper.

knowledge extractor, $\mathbf{\Sigma}^{\mathrm{KE}}$, which, given a statement x and using $\mathcal{P}^*$ as a subroutine, can compute a witness w satisfying $(x, w) \in R$ with at most a negligible error probability, from two accepting conversations of the form $(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ and $(\textsc{Cmt}, \textsc{Cha}', \textsc{Res}')$ with $\textsc{Cha} \neq \textsc{Cha}'$.

Honest Verifier Zero-Knowledge. Given a statement x and a random challenge $\textsc{Cha} \leftarrow \mathbf{\Sigma}^2(1^\lambda)$, we can produce in polynomial-time, without knowing the witness w, an accepting conversation $(\textsc{Cmt}, \textsc{Cha}, \textsc{Res})$ whose distribution is the same as the real accepting conversation. In other words, there is a PPT algorithm called a *simulator*, $\mathbf{\Sigma}^{\mathrm{sim}}$, such that $(\textsc{Cmt}, \textsc{Res}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x, \textsc{Cha})$.

Any Σ-protocol can be proved to be a proof of knowledge system ([11]).

We will need in this paper a property called *unique answer property* [6] that for legitimately produced commitment $\textsc{Cmt}$ and challenge $\textsc{Cha}$, there exists one and only one response $\textsc{Res} =: w'$ that is accepted by a verifier. Known Σ-protocols such as the Schnorr protocol and the Guillou-Quisquater protocol [33, 5] possess this property. For such a unique answer w' we consider a statement x' such that $(x', w') \in R$. Then, we further assume that both a prover and a verifier can compute, in polynomial-time, such an x' from $(x, \textsc{Cmt}, \textsc{Cha})$. We denote the PPT algorithm as $\mathbf{\Sigma}^{\mathrm{stmtgen}}$. That is;

$\mathbf{\Sigma}^{\mathrm{stmtgen}}(x, \textsc{Cmt}, \textsc{Cha})$:
Compute x' s.t.
$\exists^1 w'$ s.t. $[(x', w') \in R \ \wedge (\textsc{Cmt}, \textsc{Cha}, \textsc{Res} := w')$ is an accepting conversation]
Return x'

Known Σ-protocols [33, 5] possess this *statement generation property* (see Appendix A).

The OR-proof [11] Consider the following relation for a boolean formula $f(X_1, X_2) = X_1 \vee X_2$.

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

The corresponding language for the relation R_{OR} is given as follows.

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

The OR-proof is defined as an interactive proof system for the language $L_{R_{\mathrm{OR}}}$.

Suppose that a Σ-protocol $\mathbf{\Sigma}$ on a relation R is given. Then we can construct a new protocol, $\mathbf{\Sigma}_{\mathrm{OR}}$, on a relation R_{OR} as follows. For instance, suppose $(x_0, w_0) \in R$ holds. $\mathcal{P}$ computes $\textsc{Cmt}_0 \leftarrow \mathbf{\Sigma}^1(x_0, w), \textsc{Cha}_1 \leftarrow \mathbf{\Sigma}^2(1^\lambda)$, $(\textsc{Cmt}_1, \textsc{Res}_1) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x_1, \textsc{Cha}_1)$ and sends $(\textsc{Cmt}_0, \textsc{Cmt}_1)$ to $\mathcal{V}$. Then $\mathcal{V}$ sends $\textsc{Cha} \leftarrow \mathbf{\Sigma}^2(1^\lambda)$ to $\mathcal{P}$. Then, $\mathcal{P}$ computes $\textsc{Cha}_0 := \textsc{Cha} \oplus \textsc{Cha}_1, \textsc{Res}_0 \leftarrow \mathbf{\Sigma}^3(x_0, w_0, \textsc{Cmt}_0, \textsc{Cha}_0)$ answers to $\mathcal{V}$ with $(\textsc{Cha}_0, \textsc{Cha}_1)$ and $(\textsc{Res}_0, \textsc{Res}_1)$. Here $\oplus$ denotes a bitwise exclusive-OR operation. Then both $(\textsc{Cmt}_0, \textsc{Cha}_0, \textsc{Res}_0)$ and $(\textsc{Cmt}_1, \textsc{Cha}_1, \textsc{Res}_1)$ are accepting conversations and have the same distribution as real accepting conversations. This protocol $\mathbf{\Sigma}_{\mathrm{OR}}$ can be proved to be a Σ-protocol. We often call this Σ-protocol $\mathbf{\Sigma}_{\mathrm{OR}}$ the *OR-proof*.

Boolean Proof [10, 3] We revisit the notion of a public coin interactive proof of knowledge system for the language L_f introduced by Cramer, Damgård and Schoenmakers [10], which we call a *boolean proof system*. Then we restate the definitions for the sake of clarity.

Let R be a binary relation. Let $f(X_{i_1}, \ldots, X_{i_a})$ be a boolean formula over boolean variables $U = \{X_1, \ldots, X_u\}$.

Definition 1 ([10], Rewritten Form) *A relation R_f is defined by:*

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

R_f is a generalization of the relation R_{OR} for the OR-proof [10, 11], where f is a boolean formula with the single boolean connective: $X_1 \vee X_2$. 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 for the relation R_f is given as follows.

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

Finally, we achieve the following definition.

Definition 2 *A boolean proof system is an interactive proof system for the language L_f.*

We will provide a concrete procedure of a Σ-protocol $\mathbf{\Sigma}_f$ that is a boolean proof system.

The Fiat-Shamir Transform [1] Suppose that a cryptographic hash function with collision resistance, $\mathit{Hash}_\mu(\cdot) : \{1,0\}^* \to \{1,0\}^{l(\lambda)}$, is given. We fix a hash key μ hereafter. A Σ-protocol $\mathbf{\Sigma}$ on a relation R can be transformed into a non-interactive zero-knowledge proof of knowledge system (NIZKPoK) with its knowledge extractor in the random oracle model. Hence a non-interactive witness-indistinguishable proof of knowledge system (NIWIPoK) can be obtained. When a Σ-protocol $\mathbf{\Sigma}$ is an identification scheme, the resulting scheme is a digital signature scheme. The transform is described as follows. (Here, a message m is omitted in the case of a NIWIPoK.) Given a message $m \in \{1,0\}^*$, execute: $a \leftarrow \mathbf{\Sigma}^1(x, w)$, $c \leftarrow \mathit{Hash}_\mu(a \parallel m)$, $z \leftarrow \mathbf{\Sigma}^3(x, w, a, c)$. Then $\sigma := (a, z)$ is a signature on m. We denote the above signing algorithm as $\mathrm{FS}(\mathbf{\Sigma})^{\mathrm{sign}}(x, w, m) \to (a, z) =: \sigma$. The verification algorithm $\mathrm{FS}(\mathbf{\Sigma})^{\mathrm{vrfy}}(x, m, \sigma)$ is given as: $c \leftarrow \mathit{Hash}_\mu(a \parallel m)$, Return $b \leftarrow \mathbf{\Sigma}^{\mathrm{vrfy}}(x, a, c, z)$.

The signature scheme $\mathrm{FS}(\mathbf{\Sigma}) = (R, \mathrm{FS}(\mathbf{\Sigma})^{\mathrm{sign}}, \mathrm{FS}(\mathbf{\Sigma})^{\mathrm{vrfy}})$ can be proved, in the random oracle model, to be *existentially unforgeable against chosen-message attacks* if and only if the underlying Σ-protocol $\mathbf{\Sigma}$ is secure against *passive attacks* as an identification scheme [1]. More precisely, let q_H denote the maximum number of hash queries issued by the adversary on $\mathrm{FS}(\mathbf{\Sigma})$. Then, for any PPT algorithm $\mathcal{F}$, there exists a PPT algorithm $\mathcal{B}$ which satisfies the following inequality ($\mathrm{neg}(\cdot)$ means a negligible function).

$$\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathrm{FS}(\mathbf{\Sigma})}(\lambda) \leq q_H \mathbf{Adv}^{\text{pa}}_{\mathcal{B},\mathbf{\Sigma}}(\lambda) + \mathrm{neg}(\lambda).$$

2.2 Signature Bundle Scheme (Credential Bundle Scheme [30])

A signature bundle scheme $\mathtt{SB}$ is an extended notion of signature scheme. It consists of three PPT algorithms: $\mathtt{SB} = (\mathbf{SB.KG}, \mathbf{SB.Sign}, \mathbf{SB.Vrfy})$. Below n is bounded by a polynomial in λ.

$\mathbf{SB.KG}(1^\lambda) \to (\mathrm{PK}, \mathrm{SK})$. Given 1^λ as input, it outputs a public key PK and a secret key SK.

$\mathbf{SB.Sign}(\mathrm{PK}, \mathrm{SK}, (m_1, \ldots, m_n)) \to (\tau, (\sigma_1, \ldots, \sigma_n))$. Given PK, SK and n messages $m_1, \ldots, m_n$, it outputs a *tag* τ and n signatures $\sigma_1, \ldots, \sigma_n$.

$\mathbf{SB.Vrfy}(\mathrm{PK}, (m_1, \ldots, m_n), (\tau, (\sigma_1, \ldots, \sigma_n))) \to 1/0$. Given PK, n messages $m_1, \ldots, m_n$, a tag τ and n signatures $\sigma_1, \ldots, \sigma_n$, it outputs 1 or 0.

Suppose that we are given a digital signature scheme $(\mathrm{KG}, \mathrm{Sign}, \mathrm{Vrfy})$. Then we can construct a signature bundle scheme as follows (according to [30]). **SB.KG** takes as input 1^λ and it runs $\mathrm{KG}(1^\lambda)$ to get $(\mathrm{PK}, \mathrm{SK})$. it outputs $(\mathrm{PK}, \mathrm{SK})$. **SB.Sign** takes as input PK, SK and a set of messages $(m_i)_{1 \le i \le n}$. It chooses a tag τ of length λ at random. Then it executes Sign on each tagged message $(\tau \parallel m_i), i = 1, \ldots, n$ and outputs signatures $\sigma_i, i = 1, \ldots, n$, respectively. **SB.Vrfy** takes as input PK, $(m_i)_{1 \le i \le n}$, τ and $(\sigma_i)_{1 \le i \le n}$. Then it executes Vrfy on each tagged message and signature, $((\tau \parallel m_i), \sigma_i), i = 1, \ldots, n$. It returns 1 if and only if Vrfy returns 1 for all i, $i = 1, \ldots, n$.

2.3 Pseudorandom Function Family [27]

A pseudorandom function family, $\{PRF_k\}_{k \in PRFkeysp(\lambda)}$, is a function family in which each function $PRF_k : \{1,0\}^* \to \{1,0\}^*$ is an efficiently-computable function that looks random to any polynomial-time distinguisher, where k is called a key and $PRFkeysp(\lambda)$ is called a key space. (See more details in, for example, the book [27].)

2.4 Access Structure [19]

Let $\mathcal{U} = \{1, \ldots, u\}$ be an attribute universe. 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 $U = \{X_1, \ldots, X_u\}$. That is, variables $X_{i_1}, \ldots, X_{i_a}$ are connected by boolean connectives; AND-gate ($\wedge$) and OR-gate ($\vee$). For example, $f = X_{i_1} \wedge ((X_{i_2} \wedge X_{i_3}) \vee X_{i_4})$ for some $i_1, i_2, i_3, i_4,\ 1 \le i_1 < i_2 < i_3 < i_4 \le u$. 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 $\mathrm{arity}(f)$. 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 formula f. For $S \in 2^{\mathcal{U}}$, we evaluate the boolean value of f at S as follows:

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

Under this definition, a boolean formula f can be seen as a map: $f : 2^{\mathcal{U}} \to \{1,0\}$. 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 only consider a *monotone* access formula f.[7]

[7] 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.

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 node corresponds to a term 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 leaf nodes, the set of all inner nodes (that is, all nodes excluding leaf nodes) 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 an 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.

2.5 Attribute-Based Identification Scheme [2]

An attribute-based identification scheme consists of four PPT algorithms:
$(\mathbf{ABID.Setup}, \mathbf{ABID.KG}, \mathcal{P}, \mathcal{V})$ [2].
$\mathbf{ABID.Setup}(1^\lambda, \mathcal{U}) \to (\mathbf{PK}, \mathbf{MSK})$**.** $\mathbf{ABID.Setup}$ takes as input the security parameter 1^λ and an attribute universe $\mathcal{U}$. It outputs a public key PK and a master secret key MSK.
$\mathbf{ABID.KG}(\mathbf{PK}, \mathbf{MSK}, S) \to \mathbf{SK}_S$**.** A key-generation algorithm $\mathbf{ABID.KG}$ takes as input the public key PK, the master secret key MSK and an attribute set $S \subset \mathcal{U}$. It outputs a signing key SK_S corresponding to S.
$\mathcal{P}(\mathbf{PK}, \mathbf{SK}_S, f)$ **and** $\mathcal{V}(\mathbf{PK}, f)$**.** $\mathcal{P}$ and $\mathcal{V}$ are interactive algorithms called a *prover* and a *verifier*, respectively. $\mathcal{P}$ takes as input the public key PK, the secret key SK_S and an access formula f. Here the secret key SK_S is given to $\mathcal{P}$ by an authority that runs $\mathbf{ABID.KG}(\mathrm{PK}, \mathrm{MSK}, S)$. $\mathcal{V}$ takes as input the public key PK and an access formula f. $\mathcal{P}$ and $\mathcal{V}$ interact with each other for at most constant rounds. Then, $\mathcal{V}$ returns its decision 1 or 0. When it is 1, we say that $\mathcal{V}$ *accepts* $\mathcal{P}$ for f. When it is 0, we say that $\mathcal{V}$ *rejects* $\mathcal{P}$ for f. We demand correctness of $\mathtt{ABID}$ that for any λ, and if $f(S) = 1$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{ABID.Setup}(1^\lambda, \mathcal{U}), \mathrm{SK}_S \leftarrow \mathbf{ABID.KG}(\mathrm{PK}, \mathrm{MSK}, S), b \leftarrow \langle \mathcal{P}(\mathrm{PK}, \mathrm{SK}_S), \mathcal{V}(\mathrm{PK}, f) \rangle : b = 1] = 1$.

Attacks on ABID and Security An adversary $\mathcal{A}$'s objective is impersonation. $\mathcal{A}$ tries to make a verifier $\mathcal{V}$ accept with an access formula f^*. The following experiment $\mathbf{Exprmt}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of *concurrent attack* on ABID.

$$\begin{aligned}
&\mathbf{Exprmt}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U}) : \\
&\ (\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{ABID.Setup}(1^\lambda, \mathcal{U}) \\
&\ (f^*, st) \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot), \mathcal{P}_j(\mathrm{PK},\mathrm{SK}_\cdot,\cdot)|_{j=1}^{q_\mathrm{p}}}(\mathrm{PK}, \mathcal{U}) \\
&\ b \leftarrow \langle \mathcal{A}(st), \mathcal{V}(\mathrm{PK}, f^*) \rangle \\
&\ \text{If } b = 1 \text{ then Return } \textsc{Win} \text{ else Return } \textsc{Lose}
\end{aligned}$$

In the experiment, $\mathcal{A}$ issues key-extraction queries to its key-generation oracle $\mathcal{KG}$. Giving an attribute set S_i, $\mathcal{A}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key SK_{S_i}. In addition, $\mathcal{A}$ invokes provers

$\mathcal{P}_j(\mathrm{PK}, \mathrm{SK}., \cdot),\ j = 1, \ldots, q'_{\mathrm{p}}, \ldots, q_{\mathrm{p}}$, by giving a pair (S_j, f_j) of an attribute set and an access formula. Acting as a verifier with an access formula f_j, $\mathcal{A}$ interacts with each $P_j(\mathrm{PK}, \mathrm{SK}_{S_j}, f_j)$ concurrently.

The access formula f^* declared by $\mathcal{A}$ is called a *target access formula*. Here we consider the *adaptive* target in the sense that $\mathcal{A}$ is allowed to choose f^* after seeing PK, issuing key-extraction queries and interacting with of provers. Two restrictions are imposed on $\mathcal{A}$ concerning f^*. In key-extraction queries, each attribute set S_i must satisfy $f^*(S_i) = 0$. In interactions with each prover, $f^*(S_j) = 0$. The number of key-extraction queries and the number of invoked provers are at most q_{k} and q_{p} in total, respectively, which are bounded by a polynomial in λ.

The *advantage* of $\mathcal{A}$ over $\mathtt{ABID}$ in the game of concurrent attack is defined as

$$\mathbf{Adv}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda) \stackrel{\mathrm{def}}{=} \Pr[\mathbf{Exprmt}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U}) \text{ returns } \textsc{Win}].$$

$\mathtt{ABID}$ is called *secure against concurrent attacks* if, for any PPT $\mathcal{A}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda)$ is negligible in λ.

The game of *passive attack* on ABID is obtained by replacing concurrent provers $\mathcal{P}_j(\mathrm{PK}, \mathrm{SK}., \cdot)|_{j=1}^{q_{\mathrm{p}}}$ with transcript oracle Transc in $\mathbf{Exprmt}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$. The following experiment $\mathbf{Exprmt}^{\mathrm{pa}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$ of an adversary $\mathcal{A}$ defines the game of *passive attack* on ABID.

$\mathbf{Exprmt}^{\mathrm{pa}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{ABID.Setup}(1^\lambda, \mathcal{U})$
$(f^*, st) \leftarrow \mathcal{A}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot),\mathrm{Transc}(\mathcal{P}(\mathrm{PK},\mathrm{SK}.,\cdot),\mathcal{V}(\mathrm{PK},\cdot))}(\mathrm{PK}, \mathcal{U})$
$b \leftarrow \langle \mathcal{A}(st), \mathcal{V}(\mathrm{PK}, f^*) \rangle$
If $b = 1$ then Return $\textsc{Win}$ else Return $\textsc{Lose}$

In the experiment, $\mathcal{A}$ issues key-extraction queries to its key-generation oracle $\mathcal{KG}$ and transcript queries to its transcript oracle Transc. In a transcript query, giving a pair (S_j, f_j) of an attribute set and an access formula, $\mathcal{A}$ queries $\mathrm{Transc}(\mathcal{P}(\mathrm{PK}, \mathrm{SK}., \cdot), \mathcal{V}(\mathrm{PK}, \cdot))$ for a whole transcript of messages interacted between $\mathcal{P}(\mathrm{PK}, \mathrm{SK}_{S_j}, f_j)$ and $\mathcal{V}(\mathrm{PK}, f_j)$.

The *advantage* $\mathbf{Adv}^{\mathrm{pa}}_{\mathcal{A},\mathtt{ABID}}(\lambda)$ and security are defined in the same way as the concurrent case. Concurrent security means passive security; for any PPT $\mathcal{A}$, there exists a PPT $\mathcal{B}$ that satisfies the following inequality.

$$\mathbf{Adv}^{\mathrm{pa}}_{\mathcal{A},\mathtt{ABID}}(\lambda) \leq \mathbf{Adv}^{\mathrm{ca}}_{\mathcal{B},\mathtt{ABID}}(\lambda). \tag{1}$$

2.6 Attribute-Based Signature Scheme [30, 31]

An attribute-based signature scheme, $\mathtt{ABS}$, consists of four PPT algorithms: $\mathtt{ABS} = (\mathbf{ABS.Setup}, \mathbf{ABS.KG}, \mathbf{ABS.Sign}, \mathbf{ABS.Vrfy})$.
$\mathbf{ABS.Setup}(1^\lambda, \mathcal{U}) \rightarrow (\mathbf{PK}, \mathbf{MSK})$**.** It takes as input the security parameter λ and an attribute universe $\mathcal{U}$. It outputs a public key PK and a master secret key MSK.
$\mathbf{ABS.KG}(\mathbf{PK}, \mathbf{MSK}, S) \rightarrow \mathbf{SK}_S$**.** It takes as input the public key PK, the master secret key MSK, and an attribute set $S \subset \mathcal{U}$. It outputs a signing key SK_S corresponding to S.
$\mathbf{ABS.Sign}(\mathbf{PK}, \mathbf{SK}_S, (m, f)) \rightarrow \sigma$**.** It takes as input a public key PK, a private secret key SK_S corresponding to an attribute set S, a pair (m, f) of a message $\in \{1, 0\}^*$ and an access formula. It outputs a signature σ.

ABS.Vrfy$(\mathbf{PK}, (m, f), \sigma)$**.** It takes as input a public key PK, a pair (m, f) of a message and an access formula, and a signature σ. It outputs a decision 1 or 0. When it is 1, we say that $((m, f), \sigma)$ is *valid*. When it is 0, we say that $((m, f), \sigma)$ is *invalid*. We demand correctness of ABS that, for any λ, any $\mathcal{U}$, any $S \subset \mathcal{U}$ and any (m, f) such that $f(S) = 1$, $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{ABS.Setup}(1^\lambda, \mathcal{U}), \mathrm{SK}_S \leftarrow \mathbf{ABS.KG}(\mathrm{PK}, \mathrm{MSK}, S), \sigma \leftarrow \mathbf{ABS.Sign}(\mathrm{PK}, \mathrm{SK}_S, (m, f)), b \leftarrow \mathbf{ABS.Vrfy}(\mathrm{PK}, (m, f), \sigma) : b = 1] = 1$.

Chosen-Message Attack on ABS and Security An adversary $\mathcal{F}$'s objective is to make an *existential forgery*. $\mathcal{F}$ tries to make a forgery $((m^*, f^*), \sigma^*)$ that consists of a message, a target access structure and a signature. The following experiment $\mathbf{Exprmt}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda, \mathcal{U})$ of a forger $\mathcal{F}$ defines the *chosen-message attack on ABS to make an existential forgery*.

$\mathbf{Exprmt}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda, \mathcal{U})$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{ABS.Setup}(1^\lambda, \mathcal{U})$
$((m^*, f^*), \sigma^*) \leftarrow \mathcal{F}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot), \mathcal{SIGN}(\mathrm{PK},\mathrm{SK}.,(\cdot,\cdot))}(\mathrm{PK})$
If $\mathbf{ABS.Vrfy}(\mathrm{PK}, (m^*, f^*), \sigma^*) = 1$ then Return Win
else Return Lose

In the experiment, $\mathcal{F}$ issues key-extraction queries to its key-generation oracle $\mathcal{KG}$ and signing queries to its signing oracle $\mathcal{SIGN}$. Giving an attribute set S_i, $\mathcal{F}$ queries $\mathcal{KG}(\mathrm{PK}, \mathrm{MSK}, \cdot)$ for the secret key SK_{S_i}. In addition, giving an attribute set S_j and a pair (m, f) of a message and an access formula, $\mathcal{F}$ queries $\mathcal{SIGN}(\mathrm{PK}, \mathrm{SK}., (\cdot, \cdot))$ for a signature σ that satisfies $\mathbf{ABS.Vrfy}(\mathrm{PK}, (m, f), \sigma) = 1$ when $f(S_j) = 1$.

The access formula f^* declared by $\mathcal{F}$ is called a *target access formula*. Here we consider the *adaptive* target in the sense that $\mathcal{F}$ is allowed to choose f^* after seeing PK and issuing some key-extraction queries and signing queries. Two restrictions are imposed on $\mathcal{F}$ concerning f^*. In key-extraction queries, S_i that satisfies $f^*(S_i) = 1$ was never queried. In signing queries, (m^*, f^*) was never queried. The number of key-extraction queries and the number of signing queries are at most q_k and q_s in total, respectively, which are bounded by a polynomial in λ.

The *advantage* of $\mathcal{F}$ over ABS in the game of chosen-message attack to make existential forgery is defined as

$$\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda) \stackrel{\text{def}}{=} \Pr[\mathbf{Exprmt}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda, \mathcal{U}) \text{ returns Win}].$$

ABS is called *existentially unforgeable against chosen-message attacks* if, for any PPT $\mathcal{F}$ and for any $\mathcal{U}$, $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda)$ is negligible in λ.

Attribute Privacy of ABS Roughly speaking, ABS is called to have attribute privacy if any unconditional cheating verifier cannot distinguish two distributions of signatures each of which is generated by different attribute set. The following definition is due to Maji et al. and Okamoto-Takashima.

Definition 3 (Attribute Privacy (Perfect Privacy [30, 31])) *ABS is called to have attribute privacy if, for all* $(PK, MSK) \leftarrow \mathbf{ABS.Setup}(1^\lambda, \mathcal{U})$*, for all message* m*, for all attribute sets* S_1 *and* S_2*, for all signing keys* $SK_{S_1} \leftarrow \mathbf{ABS.KG}(PK, MSK, S_1)$ *and* $SK_{S_2} \leftarrow \mathbf{ABS.KG}(PK, MSK, S_2)$ *and for all access formula* f *such that* $f(S_1) = 1$ *and* $f(S_2) = 1$ *or* $f(S_1) \neq 1$ *and* $f(S_2) \neq 1$*, two distributions* $\mathbf{ABS.Sign}(PK, SK_{S_1}, (m, f))$ *and* $\mathbf{ABS.Sign}(PK, SK_{S_2}, (m, f))$ *are identical.*

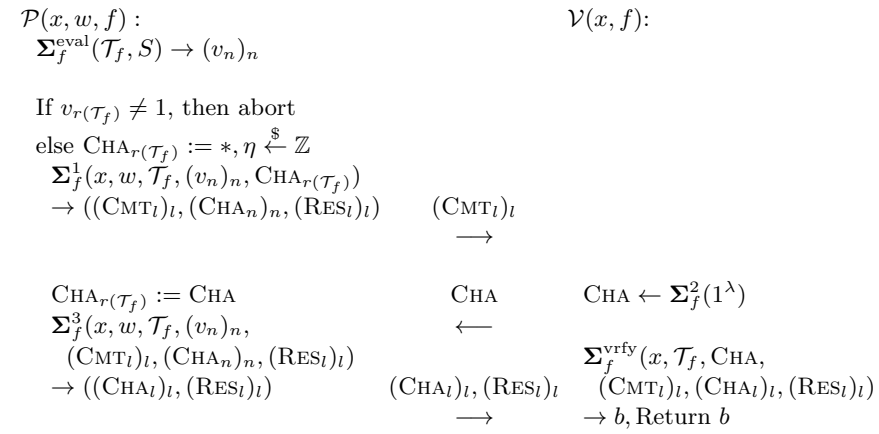

Fig. 1. Our Boolean Proof $\mathbf{\Sigma}_f$ for the language L_f.

3 Our Construction of Boolean Proof

In this section, we first construct a proof system $\mathbf{\Sigma}_f$ from a given Σ-protocol $\mathbf{\Sigma}$ and a boolean formula f. Then we prove that our $\mathbf{\Sigma}_f$ is a Σ-protocol on the relation R_f. That is, we prove that our $\mathbf{\Sigma}_f$ is a Σ-protocol which is an boolean proof for the language L_f. In Appendix C, we apply the Fiat-Shamir transform $\mathrm{FS}(\cdot)$ to our Σ-protocol $\mathbf{\Sigma}_f$ to obtain a non-interactive witness-indistinguishable proof of knowledge (NIWIPoK) system for the language L_f.

3.1 Our Procedure of Boolean Proof

Basically $\mathbf{\Sigma}_f$ is a 3-move protocol between interactive PPT algorithms $\mathcal{P}$ and $\mathcal{V}$ on initial input $(x:=(x_{i_j})_{1\leq j\leq \mathrm{arity}(f)}, w:=(w_{i_j})_{1\leq j\leq \mathrm{arity}(f)})\in R_f$ for $\mathcal{P}$ and x for $\mathcal{V}$. In our prover-algorithm $\mathcal{P}$, there are three PPT subroutines; $\mathbf{\Sigma}_f^{\mathrm{eval}}$, $\mathbf{\Sigma}_f^1$ and $\mathbf{\Sigma}_f^3$. On the other hand, in our verifier-algorithm $\mathcal{V}$, there are two PPT subroutines; $\mathbf{\Sigma}_f^2$ and $\mathbf{\Sigma}_f^{\mathrm{vrfy}}$. $\mathbf{\Sigma}_f^{\mathrm{vrfy}}$ has two subroutines; **VrfyCha** and **VrfyRes**. Fig. 1 shows our procedure of boolean proof, $\mathbf{\Sigma}_f$.

Evaluation of Satisfiability. The prover $\mathcal{P}$ begins with evaluation concerning whether and how S satisfies f. We label each node of $\mathcal{T}$ with a value $v=1$ (TRUE) or 0 (FALSE). For each leaf node l, we label l with $v_l=1$ if $\rho(l)\in S$ and $v_l=0$ otherwise. 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 node to each leaf node, recursively, in the following way.

$\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T},S)$:
$\mathcal{T}_\mathrm{L}:=\mathrm{Lsub}(\mathcal{T}),\mathcal{T}_\mathrm{R}:=\mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is an $\wedge$-node, then Return $v_{r(\mathcal{T})}:=(\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_\mathrm{L},S)\wedge\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_\mathrm{R},S))$
else if $r(\mathcal{T})$ is an $\vee$-node, then Return $v_{r(\mathcal{T})}:=(\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_\mathrm{L},S)\vee\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_\mathrm{R},S))$
else if $r(\mathcal{T})$ is a leaf node, then Return $v_{r(\mathcal{T})}:=(\rho(r(\mathcal{T}))\stackrel{?}{\in}S)$

Commitment. Prover's computation of a commitment value for each leaf node is described in Fig. 2. Basically, the algorithm $\mathbf{\Sigma}_f^1$ runs for every node from the root node to each leaf node, recursively. As a result, $\mathbf{\Sigma}_f^1$ generates for each leaf node l a value CMT_l; If $v_l=1$, then CMT_l is

computed honestly according to $\mathbf{\Sigma}^1$. Else if $v_l = 0$, then CMT_l is computed in the simulated way according to $\mathbf{\Sigma}^{\mathrm{sim}}$. Other values, $(\mathrm{CHA}_t)_t$ and $(\mathrm{RES}_l)_l$), are needed for the simulation. Note that a distinguished symbol '$*$' is used for those other values to indicate the honest computation.

$\mathbf{\Sigma}^1_f(x, w, \mathcal{T}, (v_n)_n, \mathrm{CHA})$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is $\wedge$-node, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}$
Return$(\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \mathbf{\Sigma}^1_f(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, \mathbf{\Sigma}^1_f(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}))$
else if $r(\mathcal{T})$ is $\vee$-node, then
If $v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := *, \quad \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := *$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := *, \quad \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} \leftarrow \mathbf{\Sigma}^2(1^\lambda)$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \mathbf{\Sigma}^2(1^\lambda), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := *$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \mathbf{\Sigma}^2(1^\lambda), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
Return$(\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \mathbf{\Sigma}^1_f(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}), \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, \mathbf{\Sigma}^1_f(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}))$
else if $r(\mathcal{T})$ is a leaf-node, then
If $v_{r(\mathcal{T})} = 1$, then $\mathrm{CMT}_{r(\mathcal{T})} \leftarrow \mathbf{\Sigma}^1(x_{\rho(r(\mathcal{T}))}, w_{\rho(r(\mathcal{T}))}), \mathrm{RES}_{r(\mathcal{T})} := *$
else if $v_{r(\mathcal{T})} = 0$, then $(\mathrm{CMT}_{r(\mathcal{T})}, \mathrm{RES}_{r(\mathcal{T})}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x_{\rho(r(\mathcal{T}))}, \mathrm{CHA})$
Return$(\mathrm{CMT}_{r(\mathcal{T})}, \mathrm{RES}_{r(\mathcal{T})})$

Fig. 2. The subroutine $\mathbf{\Sigma}^1_f$ of our $\mathbf{\Sigma}_f$.

Challenge. Verifier picks up a challenge value by using $\mathbf{\Sigma}^2$.

$$\mathbf{\Sigma}^2_f(1^\lambda) : \mathrm{CHA} \leftarrow \mathbf{\Sigma}^2(1^\lambda), \mathrm{Return}(\mathrm{CHA})$$

Response. Prover's computation of a response value for each leaf node is described in Fig. 3. Basically, the algorithm $\mathbf{\Sigma}^3_f$ runs for every node from the root node to each leaf node, recursively. As a result, $\mathbf{\Sigma}^3_f$ generates values, $(\mathrm{CHA}_t)_t$ and $(\mathrm{RES}_l)_l$). Note that all challenge values $(\mathrm{CHA}_t)_t$ are completed according to the "division rule" described in Section 1.1.

$\mathbf{\Sigma}^3_f(x, w, \mathcal{T}, (v_n)_n, (\mathrm{CMT}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l)$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is $\wedge$-node, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}$
Return$(\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \mathbf{\Sigma}^3_f(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\mathrm{CMT}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l),$
$\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, \mathbf{\Sigma}^3_f(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\mathrm{CMT}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l))$
else if $r(\mathcal{T})$ is $\vee$-node, then
If $v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \leftarrow \mathbf{\Sigma}^2(1^\lambda), \quad \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \quad \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}$
else if $v_{r(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{r(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \quad \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}$
Return$(\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, \mathbf{\Sigma}^3_f(x, w, \mathcal{T}_{\mathrm{L}}, (v_n)_n, (\mathrm{CMT}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l),$
$\mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, \mathbf{\Sigma}^3_f(x, w, \mathcal{T}_{\mathrm{R}}, (v_n)_n, (\mathrm{CMT}_l)_l, (\mathrm{CHA}_n)_n, (\mathrm{RES}_l)_l))$
else if $r(\mathcal{T})$ is a leaf-node, then
If $v_{r(\mathcal{T})} = 1$, then $\mathrm{RES}_{r(\mathcal{T})} \leftarrow \mathbf{\Sigma}^3(x_{\rho(r(\mathcal{T}))}, w_{\rho(r(\mathcal{T}))}, \mathrm{CMT}_{r(\mathcal{T})}, \mathrm{CHA})$
else if $v_{r(\mathcal{T})} = 0$, then $\mathrm{RES}_{r(\mathcal{T})} \leftarrow \mathrm{RES}_{r(\mathcal{T})}$
Return$(\mathrm{RES}_{r(\mathcal{T})})$

Fig. 3. The subroutine $\mathbf{\Sigma}^3_f$ of our $\mathbf{\Sigma}_f$.

Verification. Verifier's computation is executed for each leaf node as follows.

$\mathbf{\Sigma}_f^{\mathrm{vrfy}}(x, \mathcal{T}, \mathrm{CHA}, \mathrm{CMT}_l)_l, (\mathrm{CHA}_l)_l, (\mathrm{RES}_l)_l)$:
Return($\mathbf{VrfyCha}(\mathcal{T}, \mathrm{CHA}, (\mathrm{CHA}_l)_l) \wedge \mathbf{VrfyRes}(x, \mathcal{T}, (\mathrm{CMT}_l, \mathrm{CHA}_l, \mathrm{RES}_l)_l)$)

$\mathbf{VrfyCha}(\mathcal{T}, \mathrm{CHA}, (\mathrm{CHA}_l)_l)$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $r(\mathcal{T})$ is an $\wedge$-node,
then Return $((\mathrm{CHA} \stackrel{?}{=} \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}) \wedge (\mathrm{CHA} \stackrel{?}{=} \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{L}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_l)_l) \wedge \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{R}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_l)_l))$
else if $r(\mathcal{T})$ is an $\vee$-node,
then Return $((\mathrm{CHA} \stackrel{?}{=} \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})} \oplus \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})})$
$\wedge \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{L}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_l)_l) \wedge \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{R}}, \mathrm{CHA}_{r(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_l)_l))$
else if $r(\mathcal{T})$ is a leaf node,
then Return $(\mathrm{CHA} \stackrel{?}{\in} \mathrm{CHASP}(1^\lambda))$

$\mathbf{VrfyRes}(x, \mathcal{T}, (\mathrm{CMT}_l, \mathrm{CHA}_l, \mathrm{RES}_l)_l)$:
For $l \in \mathrm{Leaf}(\mathcal{T})$
If $\mathbf{\Sigma}^{\mathrm{vrfy}}(x_{\rho(l)}, \mathrm{CMT}_l, \mathrm{CHA}_l, \mathrm{RES}_l) = 0$, then Return (0)
Return (1)

Now we have to check that $\mathbf{\Sigma}_f$ is certainly a Σ-protocol for the language L_f.

Proposition 1 (Completeness) *Completeness holds for our $\mathbf{\Sigma}_f$. More precisely, Suppose that $v_{r(\mathcal{T}_f)} = 1$. Then, for every node in Node($\mathcal{T}_f$), either $v_n = 1$ or $\mathrm{CHA}_n \neq *$ holds after executing $\mathbf{\Sigma}_f^1$.*

Proof. 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 $\mathbf{\Sigma}$. Suppose that the case of height k holds and consider the case of height $k+1$. The construction of $\mathbf{\Sigma}_f^1$ assures the case of height $k+1$. □

Proposition 2 (Special Soundness) *Special soundness holds for our $\mathbf{\Sigma}_f$.*

We can construct a knowledge extractor $\mathbf{\Sigma}_f^{\mathrm{KE}}$ from a knowledge extractor $\mathbf{\Sigma}^{\mathrm{KE}}$ of the underlying Σ-protocol $\mathbf{\Sigma}$ as follows.

$\mathbf{\Sigma}_f^{\mathrm{KE}}(x, (\mathrm{CMT}_l, \mathrm{CHA}_l, \mathrm{RES}_l)_{l \in \mathrm{Leaf}(\mathcal{T}_f)}, (\mathrm{CMT}_l, \mathrm{CHA}'_l, \mathrm{RES}'_l)_{l \in \mathrm{Leaf}(\mathcal{T}_f)})$:
For $1 \leq j \leq \mathrm{arity}(f) : w^*_{i_j} := *$
For $l \in \mathrm{Leaf}(\mathcal{T}_f)$
If $\mathrm{CHA}_l \neq \mathrm{CHA}'_l$, then $w^*_{\rho(l)} \leftarrow \mathbf{\Sigma}^{\mathrm{KE}}(x_{\rho(l)}, (\mathrm{CMT}_l, \mathrm{CHA}_l, \mathrm{RES}_l), (\mathrm{CMT}_l, \mathrm{CHA}'_l, \mathrm{RES}'_l))$
else If $w^*_{\rho(l)} = *$, then $w^*_{\rho(l)} \leftarrow \{1, 0\}^*$
Return $(w^* := (w^*_{i_j})_{1 \leq j \leq \mathrm{arity}(f)})$

Then Lemma 1 assures the proposition.

Lemma 1 (Witness Extraction) *The set w^* output by $\mathbf{\Sigma}_f^{KE}$ satisfies $(x, w^*) \in R_f$.*

Proof. Induction on the number of all $\vee$-nodes in iNode(T_f). First remark that $\textsc{Cha} \neq \textsc{Cha}'$.

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 leaf nodes.

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 height 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 (Honest Verifier Zero-Knowledge) *Honest verifier zero-knowledge property holds for our $\mathbf{\Sigma}_f$.*

Proof. This is the immediate consequence of honest verifier zero-knowledge property of $\mathbf{\Sigma}$. That is, we can construct a polynomial-time simulator $\mathbf{\Sigma}_f^{\mathrm{sim}}$ which, on input $(\mathrm{PK}, \textsc{Cha})$, outputs commitment and response message of $\mathbf{\Sigma}_f$. □

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

Theorem 1 *Our procedure $\mathbf{\Sigma}_f$ obtained from a Σ-protocol $\mathbf{\Sigma}$ on the relation R and a boolean formula f is a Σ-protocol on the relation R_f.*

Corollary 1 *Our procedure $\mathbf{\Sigma}_f$ is an boolean proof for the language L_f.*

4 Our Attribute-Based Identification Scheme

In this section, by combining our Σ-protocol of boolean proof in Section 3 with a signature bundle scheme of the Fiat-Shamir signature, we obtain a verifier-policy attribute-based identification scheme.

4.1 Our ABID

By combining our $\mathbf{\Sigma}_f$ in Section 3 with a signature bundle scheme FS($\mathbf{\Sigma}$), we obtain a scheme of VP-ABID, `ABID`, which has collusion resistance against collecting private secret keys. Our `ABID` has a feature that it can be constructed without pairings. Fig. 4 shows the construction of our ABID, $\texttt{ABID} = (\mathbf{ABID.Setup}, \mathbf{ABID.KG}, \mathcal{P}, \mathcal{V})$.

4.2 Security of Our ABID

Theorem 2 (Concurrent Security) *If the employed signature scheme FS($\mathbf{\Sigma}$) is existentially unforgeable against chosen-message attacks, then our `ABID` is secure against concurrent attacks. More precisely, for any PPT algorithm $\mathcal{A}$, there exists a PPT algorithm $\mathcal{F}$ which satisfies the following inequality ($neg(\cdot)$ means a negligible function).*

$$\mathbf{Adv}^{ca}_{\mathcal{A},\texttt{ABID}}(\lambda) \leq (\mathbf{Adv}^{euf\text{-}cma}_{\mathcal{F},FS(\mathbf{\Sigma})}(\lambda))^{1/2} + neg(\lambda).$$

$\mathbf{ABID.Setup}(1^\lambda, \mathcal{U})$:
$(x_{\mathrm{mst}}, w_{\mathrm{mst}}) \leftarrow \mathrm{Instance}_R(1^\lambda)$
$\mu \leftarrow \mathit{Hashkeysp}(\lambda)$
$\mathrm{PK} := (x_{\mathrm{mst}}, \mathcal{U}, \mu), \mathrm{MSK} := (w_{\mathrm{mst}})$
Return $(\mathrm{PK}, \mathrm{MSK})$

$\mathbf{ABID.KG}(\mathrm{PK}, \mathrm{MSK}, S)$:
$k \leftarrow \mathit{PRFkeysp}(\lambda), \tau \leftarrow \{1,0\}^\lambda$
For $i \in S$
$m_i := (\tau \parallel i), a_i \leftarrow \mathbf{\Sigma}^1(x_{\mathrm{mst}}, w_{\mathrm{mst}})$
$c_i \leftarrow \mathit{Hash}_\mu(a_i \parallel m_i), w_i \leftarrow \mathbf{\Sigma}^3(x_{\mathrm{mst}}, w_{\mathrm{mst}}, a_i, c_i)$
$\mathrm{SK}_S := (k, \tau, (a_i, w_i)_{i \in S})$
Return SK_S

$\mathcal{P}(\mathrm{PK}, \mathrm{SK}_S, f)$:
$\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_f, S) \to (v_n)_n$
If $v_{r(\mathcal{T}_f)} \neq 1$, then abort
else $\textsc{Cha}_{r(\mathcal{T}_f)} := *$
$\mathbf{Supp}(\mathrm{PK}, \mathrm{SK}_S, f) \to (a_{i_j}, w_{i_j})_{1 \le j \le \mathrm{arity}(f)}$
$w := (w_{i_j})_{1 \le j \le \mathrm{arity}(f)}$
$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1 \le j \le \mathrm{arity}(f)})$
$\to (x_{i_j})_{1 \le j \le \mathrm{arity}(f)} =: x$
$\mathbf{\Sigma}_f^1(x, w, \mathcal{T}_f, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_f)})$ $\quad \tau, (a_{i_j})_{1 \le j \le \mathrm{arity}(f)}$
$\to ((\textsc{Cmt}_l)_l,$ $\quad (\textsc{Cmt}_l)_l$
$(\textsc{Cha}_n)_n,$ $\quad \longrightarrow$
$(\textsc{Res}_l)_l)$

$\mathcal{V}(\mathrm{PK}, f)$:
$\textsc{Cha} \leftarrow \mathbf{\Sigma}_f^2(1^\lambda)$
$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1 \le j \le \mathrm{arity}(f)})$
$\to (x_{i_j})_{1 \le j \le \mathrm{arity}(f)} =: x$
$\mathbf{\Sigma}_f^{\mathrm{vrfy}}(x, \mathcal{T}_f,$
$(\textsc{Cmt}_l)_l,$
$\textsc{Cha}, (\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l) \to b$
Return b

$\textsc{Cha}_{r(\mathcal{T}_f)} := \textsc{Cha}$ $\quad \textsc{Cha}$
$\mathbf{\Sigma}_f^3(x, w, \mathcal{T}_f, (v_n)_n,$ $\quad \longleftarrow$
$(\textsc{Cmt}_l)_l,$
$\textsc{Cha}_{r(\mathcal{T}_f)}, (\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$
$\to ((\textsc{Cha}_n)_{n \in}, \quad (\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l) \quad (\textsc{Res}_l)_l$
$\longrightarrow$

Fig. 4. The scheme of our ABID.

Note that $\mathrm{FS}(\mathbf{\Sigma})$ is only known to be secure in the random oracle model.

Proof. Employing any given adversary $\mathcal{A}$ as subroutine, we construct a signature forger $\mathcal{F}$ as follows. $\mathcal{F}$ can answer to $\mathcal{A}$'s key-extraction queries for a secret key SK_S because $\mathcal{F}$ can query his signing oracle about $(\tau \parallel i; i \in S)$, where $\mathcal{F}$ choose τ at random. $\mathcal{F}$ can simulate any concurrent prover with SK_S which $\mathcal{A}$ invokes because $\mathcal{F}$ can generate SK_S in the above way. After the learning phase above, $\mathcal{F}$ simulates a verifier with which $\mathcal{A}$ begins to interact as a prover. There $\mathcal{F}$ rewinds $\mathcal{A}$ once and $\mathcal{F}$ can obtain a witness w^* by running $\mathbf{\Sigma}_f^{\mathrm{KE}}$, as well as the set of attributes S^* where the rewinding works. We use here the Reset Lemma [5], which reduces the advantage $\mathbf{Adv}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda)$ to $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathrm{FS}(\mathbf{\Sigma})}(\lambda)$ with a loss of exponent by 1/2. Finally $\mathcal{F}$ converts (w^*, S^*) into SK_{S^*}. □

More on Reduction of Concurrent Security We mean by "a number theoretic problem" the discrete-logarithm problem or the RSA-inverse problem ([5]). There exists the following security reduction to a number theoretic problem.

$$\mathbf{Adv}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda) \leq q_H^{1/2}(\mathbf{Adv}^{\text{num.prob.}}_{\mathcal{S},\mathbf{Grp}}(\lambda))^{1/4} + \mathrm{neg}(\lambda). \tag{2}$$

Here we denote q_H as the maximum number of hash queries issued by forger $\mathcal{F}$ on $\mathrm{FS}(\mathbf{\Sigma})$ in the random oracle model.

Proof. As is discussed in Section 2.1, we can reduce the advantage $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathrm{FS}(\mathbf{\Sigma})}(\lambda)$ to the advantage $\mathbf{Adv}^{\mathrm{pa}}_{\mathcal{B},\mathbf{\Sigma}}(\lambda)$ of passive security of the underlying Σ-protocol, in the random oracle model, with a loss factor q_H. Applying the Reset Lemma [5], we can reduce $\mathbf{Adv}^{\mathrm{pa}}_{\mathcal{B},\mathbf{\Sigma}}(\lambda)$ to the advantage $\mathbf{Adv}^{\text{num.prob.}}_{\mathcal{S},\mathbf{Grp}}(\lambda)$ of a PPT solver $\mathcal{S}$ of a number theoretic problem, with a loss of exponent by 1/2. □

In Appendix A, we provide two instantiations of our $\mathbf{\Sigma}_f$ and $\mathtt{ABID}$; in the discrete-logarithm setting and in the RSA setting.

5 Our Attribute-Based Signature Scheme

In this section, by applying the Fiat-Shamir transform to our ABID in Section 4.1, we obtain an attribute-based signature scheme. Our ABS is collusion resistant against collecting private secret keys, EUF-CMA secure in the random oracle model, and has a feature that it can be constructed without pairings. We note that our ABS does not have attribute privacy.

5.1 Our ABS

By applying $\mathrm{FS}(\cdot)$ to our $\mathtt{ABID}$ in Section 4.1, we obtain an attribute-based signature scheme, $\mathtt{ABS}$. Fig. 5 shows shows the construction of our ABS, $\mathtt{ABS} = (\mathbf{ABS.Setup}, \mathbf{ABS.KG}, \mathbf{ABS.Sign}, \mathbf{ABS.Vrfy})$.

ABS.Setup chooses, on input 1^λ and $\mathcal{U}$, a pair $(x_{\mathrm{mst}}, w_{\mathrm{mst}})$ at random from $R = \{(x, w)\}$ by running $\mathrm{Instance}_R(1^\lambda)$, where $|x|$ and $|w|$ are bounded by a polynomial in λ. It also chooses a hash key μ at random from a hash-key space $\mathit{Hashkeysp}(\lambda)$. It outputs a public key $\mathrm{PK} = (x_{\mathrm{mst}}, \mathcal{U}, \mu)$ and a master secret key $\mathrm{MSK} = (w_{\mathrm{mst}})$.

$\mathbf{ABS.Setup}(1^\lambda, \mathcal{U})$:
$(x_{\mathrm{mst}}, w_{\mathrm{mst}}) \leftarrow \mathrm{Instance}_R(1^\lambda), \mu \leftarrow \mathit{Hashkeysp}(\lambda)$
$\mathrm{PK} := (x_{\mathrm{mst}}, \mathcal{U}, \mu), \mathrm{MSK} := (w_{\mathrm{mst}})$
$\mathrm{Return}(\mathrm{PK}, \mathrm{MSK})$

ABS.**KG**, chooses, on input $\mathrm{PK}, \mathrm{MSK}, S$, a PRF key k from $\mathit{PRFkeysp}(\lambda)$ at random and a random string τ from $\{1,0\}^\lambda$ at random. Then KG applies the signature bundle technique [30] for each message $m_i := (\tau \parallel i), i \in S$. Here we employ the Fiat-Shamir signing algorithm $\mathrm{FS}(\mathbf{\Sigma})^{\mathrm{sign}}$ (see 2.1).

$$
\begin{aligned}
&\mathbf{ABS.KG}(\mathrm{PK}, \mathrm{MSK}, S):\\
&\ k \leftarrow \mathit{PRFkeysp}(\lambda), \tau \leftarrow \{1,0\}^\lambda\\
&\ \text{For } i \in S:\\
&\quad m_i := (\tau \parallel i), a_i \leftarrow \mathbf{\Sigma}^2(x_{\mathrm{mst}}, w_{\mathrm{mst}})\\
&\quad c_i \leftarrow \mathit{Hash}_\mu(a_i \parallel m_i), w_i \leftarrow \mathbf{\Sigma}^3(x_{\mathrm{mst}}, w_{\mathrm{mst}}, a_i, c_i)\\
&\ \mathrm{SK}_S := (k, \tau, (a_i, w_i)_{i \in S}), \text{Return } \mathrm{SK}_S.
\end{aligned}
$$

ABS.**Sign** uses a supplementation algorithm **Supp** and a statement-generator algorithm **StmtGen**.

Supp runs for $j,\ 1 \le j \in \mathrm{arity}(f)$, and generates simulated keys (a_{i_j}, w_{i_j}) for $i_j \notin S$.

$$
\begin{aligned}
&\mathbf{Supp}(\mathrm{PK}, \mathrm{SK}_S, f):\\
&\ \text{For } j = 1 \text{ to } \mathrm{arity}(f):\\
&\quad \text{If } i_j \notin S, \text{ then}\\
&\qquad m_{i_j} := (\tau \parallel i_j), c_{i_j} \leftarrow \mathit{PRF}_k(m_{i_j} \parallel 0)\\
&\qquad (a_{i_j}, w_{i_j}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x_{\mathrm{mst}}, c_{i_j}; \mathit{PRF}_k(m_{i_j} \parallel 1))\\
&\ \text{Return } (a_{i_j}, w_{i_j})_{1 \le j \le \mathrm{arity}(f)}
\end{aligned}
$$

StmtGen generates, for each $j,\ 1 \le j \in \mathrm{arity}(f)$, a statement x_{i_j}. Note that we employ here the algorithm $\mathbf{\Sigma}^{\mathrm{stmtgen}}$ which is associated with $\mathbf{\Sigma}$, and whose existence is assured by our assumption (see Section 2.1).

$$
\begin{aligned}
&\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1 \le j \le \mathrm{arity}(f)}):\\
&\ \text{For } j = 1 \text{ to } \mathrm{arity}(f):\\
&\quad m_{i_j} := (\tau \parallel i_j), c_{i_j} \leftarrow \mathit{Hash}_\mu(a_{i_j} \parallel m_{i_j})\\
&\quad x_{i_j} \leftarrow \mathbf{\Sigma}^{\mathrm{stmtgen}}(x_{\mathrm{mst}}, a_{i_j}, c_{i_j})\\
&\ \text{Return } (x_{i_j})_{1 \le j \le \mathrm{arity}(f)}
\end{aligned}
$$

Note that $(x_i, w_i) \in R$ for $i \in S$ but $\Pr[(x_i, w_i) \in R] = \mathrm{neg}(\lambda)$ for $i \notin S$.

Then **ABS**.**Sign** is obtained by adding the following procedures.

$$
\begin{aligned}
&\mathbf{Supp}(\mathrm{PK}, \mathrm{SK}_S, f) \to (a_{i_j}, w_{i_j})_{1 \le j \le \mathrm{arity}(f)}\\
&w := (w_{i_j})_{1 \le j \le \mathrm{arity}(f)}\\
&\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1 \le j \le \mathrm{arity}(f)})\\
&\quad \to (x_{i_j})_{1 \le j \le \mathrm{arity}(f)} =: x
\end{aligned}
$$

The above procedures are needed to input a pair of statement and witness, $(x = (x_{i_j})_{1 \le j \le \mathrm{arity}(f)}, w = (w_{i_j})_{1 \le j \le \mathrm{arity}(f)})$, to $\mathbf{\Sigma}^1_f$. Note here that $(x_{i_j}, w_{i_j}) \in R$ for any $i_j \in S$. On the other hand, $(x_{i_j}, w_{i_j}) \notin R$ for any $i_j \notin S$, without a negligible probability, $\mathrm{neg}(\lambda)$.

Hence the message on the first move has to include not only commitments $(\textsc{Cmt}_l)_l$ but also a string τ and elements $(a_{i_j})_{1\le j\le \mathrm{arity}(f)}$ for the verifier $\mathcal{V}$ to be able to produce the same statement x.

Finally, **ABS.Vrfy** utilizes **StmtGen** and $\boldsymbol{\Sigma}_f^{\mathrm{vrfy}}$ to check validity of the pair of message and access formula, (m, f), and the signature σ, under the public key PK.

ABS.Setup$(1^\lambda, \mathcal{U})$:
$(x_{\mathrm{mst}}, w_{\mathrm{mst}}) \leftarrow \mathrm{Instance}_R(1^\lambda)$
$\mu \leftarrow \mathit{Hashkeysp}(\lambda)$
$\mathrm{PK} := (x_{\mathrm{mst}}, \mathcal{U}, \mu), \mathrm{MSK} := (w_{\mathrm{mst}})$
Return$(\mathrm{PK}, \mathrm{MSK})$

ABS.KG$(\mathrm{PK}, \mathrm{MSK}, S)$:
$k \leftarrow \mathit{PRFkeysp}(\lambda), \tau \leftarrow \{1,0\}^\lambda$
For $i \in S$
$m_i := (\tau \parallel i), a_i \leftarrow \boldsymbol{\Sigma}^1(x_{\mathrm{mst}}, w_{\mathrm{mst}})$
$c_i \leftarrow \mathit{Hash}_\mu(a_i \parallel m_i), w_i \leftarrow \boldsymbol{\Sigma}^3(x_{\mathrm{mst}}, w_{\mathrm{mst}}, a_i, c_i)$
$\mathrm{SK}_S := (k, \tau, (a_i, w_i)_{i\in S})$
Return SK_S

ABS.Sign$(\mathrm{PK}, \mathrm{SK}_S, (m, f))$:

$\boldsymbol{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_f, S) \to (v_n)_n$
If $v_{r(\mathcal{T}_f)} \neq 1$, then abort
else $\textsc{Cha}_{r(\mathcal{T}_f)} := *$

$\mathbf{Supp}(\mathrm{PK}, \mathrm{SK}_S, f) \to (a_{i_j}, w_{i_j})_{1\le j\le \mathrm{arity}(f)}$
$w := (w_{i_j})_{1\le j\le \mathrm{arity}(f)}$
$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1\le j\le \mathrm{arity}(f)})$
$\to (x_{i_j})_{1\le j\le \mathrm{arity}(f)} =: x$

$\boldsymbol{\Sigma}_f^1(x, w, \mathcal{T}_f, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_f)})$
$\to ((\textsc{Cmt}_l)_l,$
$(\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$

$\textsc{Cha} \leftarrow \mathit{Hash}_\mu((a_{i_j})_{1\le j\le \mathrm{arity}(f)} \parallel (\textsc{Cmt}_l)_l) \parallel m)$
$\textsc{Cha}_{r(\mathcal{T}_f)} := \textsc{Cha}$

$\boldsymbol{\Sigma}_f^3(x, w, \mathcal{T}_f, (v_n)_n, \textsc{Cha}_{r(\mathcal{T}_f)},$
$(\textsc{Cmt}_l)_l,$
$(\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$
$\to ((\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$

Return $\sigma := (\tau, (a_{i_j})_{1\le j\le \mathrm{arity}(f)},$
$(\textsc{Cmt}_l)_l,$
$(\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$

ABS.Vrfy$(\mathrm{PK}, (m, f), \sigma := (\tau, (a_{i_j})_{1\le j\le \mathrm{arity}(f)},$
$(\textsc{Cmt}_l)_l,$
$(\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)):$

$\textsc{Cha} \leftarrow \mathit{Hash}_\mu((a_{i_j})_{1\le j\le \mathrm{arity}(f)} \parallel (\textsc{Cmt}_l)_l \parallel m)$

$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{i_j})_{1\le j\le \mathrm{arity}(f)})$
$\to (x_{i_j})_{1\le j\le \mathrm{arity}(f)} =: x$

$\boldsymbol{\Sigma}_f^{\mathrm{vrfy}}(x, \mathcal{T}_f,$
$(\textsc{Cmt}_l)_l,$
$\textsc{Cha}, (\textsc{Cha}_n)_n,$
$(\textsc{Res}_l)_l)$
$\to b$
Return b

Fig. 5. The scheme of our ABS.

5.2 Security of Our ABS

Applying the standard technique in the work of Abdalla et al. [1] shows that the security of our ABS is equivalent to the security of an attribute-based identification scheme, ABID, against passive attacks, where our ABID is obtained by combining our Σ-protocol $\boldsymbol{\Sigma}_f$ with the signature

bundle scheme of the Fiat-Shamir signature FS($\mathbf{\Sigma}$) in the same way as ABS (See Appendix 4 for our ABID).

Theorem 3 (Unforgeability) *Our attribute-based signature scheme ABS is existentially unforgeable against chosen-message attacks in the random oracle model, based on the passive security of ABID. More precisely, let q_H denote the maximum number of hash queries issued by a forger $\mathcal{F}$ on ABS. Then, for any PPT algorithm $\mathcal{F}$, there exists a PPT algorithm $\mathcal{B}$ which satisfies the following inequality ($neg(\cdot)$ means a negligible function).*

$$\mathbf{Adv}^{euf\text{-}cma}_{\mathcal{F},ABS}(\lambda) \leq q_H \mathbf{Adv}^{pa}_{\mathcal{B},ABID}(\lambda) + neg(\lambda). \tag{3}$$

Proof. First, our ABS is considered to be obtained by applying the Fiat-Shamir transform to our ABID. This is because, in the first message of our ABID, the tag τ and the elements $(a_{i_j})_{1\leq j\leq \mathrm{arity}(f)}$ are fixed even when the 3-move protocol is repeated between the prover $\mathcal{P}$ with a secret key SK_S and the verifier $\mathcal{V}$ with an access structure f. So ABS = FS(ABID).

As is discussed in Section 2.1, we can reduce the advantage $\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda)$ to the advantage $\mathbf{Adv}^{\text{pa}}_{\mathcal{B},\mathtt{ABID}}(\lambda)$ of passive security of the underlying ABID, in the random oracle model, with a loss factor q_H. This is because $\mathcal{B}$ can simulate key-extraction queries of $\mathcal{F}$ perfectly with the aid of the key-generation oracle of $\mathcal{B}$. □

More on Reduction of Unforgeability Let q_H denote the maximum number of hash queries issued by a forger $\mathcal{F}$ on ABS and a forger $\mathcal{F}'$ on FS($\mathbf{\Sigma}$). Combining the inequality (3) with the inequalities (1) and (2) in Section 2.5 and Section 4, we obtain the following security reduction of advantages.

$$\mathbf{Adv}^{\text{euf-cma}}_{\mathcal{F},\mathtt{ABS}}(\lambda) \leq q_H^{3/2}(\mathbf{Adv}^{\text{num.prob.}}_{\mathcal{S},\mathbf{Grp}}(\lambda))^{1/4} + \mathrm{neg}(\lambda).$$

Attribute Privacy Our ABS does not have attribute privacy defined in Section 2.6 because of its linkability; that is, the constant components $\tau, (a_{i_j})_j$ make two signatures linkable (especially, τ is a component of a private secret key SK_S). Hence, our ABS merely has attribute privacy *as a one-time signature scheme.*

6 Conclusions

We proposed an attribute-based signature scheme (ABS) with short signatures and security proof in the random oracle model. First we provided a concrete procedure of the Σ-protocol which enables a prover to prove possession of witnesses that satisfy a statement of a monotone boolean formula. Next, combining the Σ-protocol with a signature bundle scheme of the Fiat-Shamir signature, we obtained a generic attribute-based identification scheme (ABID). Then we applied the Fiat-Shamir transform to our ABID to obtain a scheme of ABS. The series of these generic constructions were obtained from a given single Σ-protocol. Finally, we provided our ABID and ABS schemes concretely in the Discrete-Logarithm setting and the RSA setting. These concretions are pairing-free.

Signatures of our ABS are linkable, and attribute privacy holds only as a one-time signature. To construct an efficient ABS with attribute privacy without pairings in the way of the boolean proof system is a challenging problem.

Acknowledgements We appreciate sincere comments of Keita Emura and Shingo Hasegawa at SCIS2014 as well as anonymous reviewers of ASIAPKC2014.

We also thank to Javier Herranz for his sincere comment via private communication [25].

References

1. M. Abdalla, J. H. An, M. Bellare, and C. Namprempre. From identification to signatures via the fiat-shamir transform: Minimizing assumptions for security and forward-security. In *EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 418–433. Springer, 2002.
2. H. Anada, S. Arita, S. Handa, and Y. Iwabuchi. Attribute-based identification: Definitions and efficient constructions. In *ACISP 2013*, volume 7959 of *LNCS*, pages 168–186. Springer, 2013.
3. H. Anada, S. Arita, and K. Sakurai. Attribute-based signatures without pairings via the fiat-shamir paradigm. In *ASIAPKC2014*, volume 2 of *ACM-ASIAPKC*, pages 49–58. ACM, 2014.
4. M. Bellare and O. Goldreich. On defining proofs of knowledge. In *CRYPTO '92*, volume 740 of *LNCS*, pages 390–420. Springer, 1992.
5. M. Bellare and A. Palacio. Gq and schnorr identification schemes: Proofs of security against impersonation under active and concurrent attacks. In *CRYPTO 2002*, volume 2442 of *LNCS*, pages 162–177. Springer, 2002.
6. M. Bellare and S. Shoup. Two-tier signatures, strongly unforgeable signatures, and fiat-shamir without random oracles. In *Public Key Cryptography - PKC 2007, 10th International Conference on Practice and Theory in Public-Key Cryptography, Beijing, China, April 16-20, 2007, Proceedings*, pages 201–216, 2007.
7. D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 223–238. Springer, 2004.
8. D. Boneh and X. Boyen. Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, pages 56–73, 2004.
9. R. Cramer. *Modular Designs of Secure, yet Practical Cyptographic Protocols*. PhD thesis, University of Amsterdam, Amsterdam, the Netherlands, 1996.
10. R. Cramer, I. Damgård, and B. Schoenmakers. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO '94*, pages 174–187. Springer-Verlag, 1994.
11. I. Damgård. On σ-protocols. In Course Notes, https://services.brics.dk/java/courseadmin/CPT/documents, 2011.
12. A. El Kaafarani, L. Chen, E. Ghadafi, and J. H. Davenport. Attribute-based signatures with user-controlled linkability. In *Cryptology and Network Security - 13th International Conference, CANS 2014, Heraklion, Crete, Greece, October 22-24, 2014. Proceedings*, pages 256–269, 2014.
13. A. El Kaafarani, E. Ghadafi, and D. Khader. Decentralized traceable attribute-based signatures. In *Topics in Cryptology - CT-RSA 2014 - The Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings*, pages 327–348, 2014.
14. A. Escala, J. Herranz, and P. Morillo. Revocable attribute-based signatures with adaptive security in the standard model. In *Progress in Cryptology - AFRICACRYPT 2011 - 4th International Conference on Cryptology in Africa, Dakar, Senegal, July 5-7, 2011. Proceedings*, pages 224–241, 2011.
15. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, STOC '90, pages 416–426, New York, NY, USA, 1990. ACM.
16. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO '86*, volume 263 of *LNCS*, pages 186–194. Springer, 1987.
17. M. Gagné, S. Narayan, and R. Safavi-Naini. Short pairing-efficient threshold-attribute-based signature. In *Pairing-Based Cryptography - Pairing 2012 - 5th International Conference, Cologne, Germany, May 16-18, 2012, Revised Selected Papers*, pages 295–313, 2012.
18. E. Ghadafi. Stronger security notions for decentralized traceable attribute-based signatures and more efficient constructions. In *Topics in Cryptology - CT-RSA 2015, The Cryptographer's Track at the RSA Conference 2015, San Francisco, CA, USA, April 20-24, 2015. Proceedings*, pages 391–409, 2015.
19. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM-CCS '06*, volume 263, pages 89–98. ACM, 2006.

20. R. Granger, T. Kleinjung, and J. Zumbrägel. Breaking '128-bit secure' supersingular binary curves - (or how to solve discrete logarithms in $\mathrm{f}_{2^4\,1223}$ and $\mathrm{f}_{2^{12}\,367}$). In *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part II*, pages 126–145, 2014.
21. J. Groth and A. Sahai. Efficient non-interactive proof systems for bilinear groups. In *Proceedings of the Theory and Applications of Cryptographic Techniques 27th Annual International Conference on Advances in Cryptology*, EUROCRYPT'08, pages 415–432, Berlin, Heidelberg, 2008. Springer-Verlag.
22. S. Guo and Y. Zeng. Attribute-based signature scheme. In *ISA '08*, pages 509–511. IEEE, 2008.
23. J. Herranz. Attribute-based signatures from rsa. *Theoretical Computer Science*, 527:73–82, 2014.
24. J. Herranz. Attribute-based versions of schnorr and elgamal. *IACR Cryptology ePrint Archive*, 2015:213, 2015.
25. J. Herranz. Private communication via e-mail, dept. matemàtica aplicada iv, universitat politècnica de catalunya, July 2014 and Sept 2015.
26. J. Herranz, F. Laguillaumie, B. Libert, and C. Ràfols. Short attribute-based signatures for threshold predicates. In *Topics in Cryptology - CT-RSA 2012 - The Cryptographers' Track at the RSA Conference 2012, San Francisco, CA, USA, February 27 - March 2, 2012. Proceedings*, pages 51–67, 2012.
27. J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2008.
28. S. Kumar, S. Agrawal, S. Balaraman, and C. P. Rangan. Attribute based signatures for bounded multi-level threshold circuits. In *Public Key Infrastructures, Services and Applications - 7th European Workshop, EuroPKI 2010, Athens, Greece, September 23-24, 2010. Revised Selected Papers*, pages 141–154, 2010.
29. J. Li, M. H. Au, W. Susilo, D. Xie, and R. K. Attribute-based signature and its applications. In *ASIA-CCS '10*, volume 5, pages 60–69. ACM, 2010.
30. H. K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-based signatures. In *CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer, 2011. Full version available at IACR Cryptology ePrint Archive, 2010/595, http://eprint.iacr.org/.
31. T. Okamoto and K. Takashima. Efficient attribute-based signatures for non-monotone predicates in the Standard Model. In *PKC 2011*, volume 6571 of *LNCS*, pages 35–52. Springer, 2011.
32. T. Okamoto and K. Takashima. Decentralized attribute-based signatures. In *Public-Key Cryptography - PKC 2013 - 16th International Conference on Practice and Theory in Public-Key Cryptography, Nara, Japan, February 26 - March 1, 2013. Proceedings*, pages 125–142, 2013.
33. C. P. Schnorr. Efficient identification and signatures for smart cards. In *CRYPTO '89*, volume 435 of *LNCS*, pages 239–252. Springer, 1990.
34. S. F. Shahandashti and R. Safavi-Naini. Threshold attribute-based signatures and their application to anonymous credential systems. In *Progress in Cryptology - AFRICACRYPT 2009, Second International Conference on Cryptology in Africa, Gammarth, Tunisia, June 21-25, 2009. Proceedings*, pages 198–216, 2009.
35. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO '84*, volume 196 of *LNCS*, pages 47–53. Springer, 1985.
36. M. Yasuda, T. Shimoyama, J. Kogure, and T. Izu. On the strength comparison of the ECDLP and the IFP. In *Security and Cryptography for Networks - 8th International Conference, SCN 2012, Amalfi, Italy, September 5-7, 2012. Proceedings*, pages 302–325, 2012.

A Instantiations

In this section, we provide two instantiations of our $\mathbf{\Sigma}_f$, `ABID` and `ABS`; that is, in the discrete-logarithm setting and the RSA setting.

A.1 Discrete-Logarithm Setting

A prime of bit length λ is denoted by p. A multiplicative cyclic group of order p is denoted by $\mathbb{G}_p$. We fix a base $g \in \mathbb{G}_p, \langle g \rangle = \mathbb{G}_p$. The ring of the exponent domain of $\mathbb{G}_p$, which consists of integers from 0 to $p-1$ with modulo p operation, is denoted by $\mathbb{Z}_p$.

ABS.Setup takes as input $(1^\lambda, \mathcal{U})$. Let $R_\lambda := \{(\beta, \alpha) \in \mathbb{G}_p \times \mathbb{Z}_p; \beta = g^\alpha\}$. Then $\mathrm{Instance}_R(1^\lambda)$ chooses an element $(\beta, \alpha) \in R_\lambda$ at random. **ABS.Setup** outputs a public key and a master secret key: $\mathrm{PK} = ((g, \beta), \mathcal{U}, \mu), \mathrm{MSK} = \alpha$.

ABS.KG outputs SK_S with signatures, for $i \in S$, $\sigma_i = (a_i = g^{r_i},\ w_i = r_i + c_i\alpha)$. Here we use a key k obtained by $k \leftarrow \mathit{Hash}_\mu(\alpha \parallel \tau)$, put $m_i = \tau \parallel i$, and $r_i \in \mathbb{Z}_p$ is chosen at random according to a random tape: $\mathit{PRF}_k(m_i)$, and c_i is obtained by $c_i \leftarrow \mathit{Hash}_\mu(a_i \parallel m_i)$. $\mathbf{\Sigma}^{\mathrm{stmtgen}}(\beta, a_i, c_i)$ is an algorithm that computes $x_i := a_i\beta^{c_i} \in \mathbb{G}_p$.

The rest of protocol is executed according to $\mathbf{\Sigma}_f$ on input (x, w) and with the following setting.

$$\mathrm{CMT}_l = g^{r_l}, \mathrm{RES}_l = r_l + \mathrm{CHA}_l\ w_{\rho(l)},$$
$$\text{Verification Equation} : g^{\mathrm{RES}_l} \stackrel{?}{=} \mathrm{CMT}_l\ (x_{\rho(l)})^{\mathrm{CHA}_l}.$$

A.2 RSA Setting

An RSA modulus of bit length λ is denoted by N. An RSA exponent of odd prime is denoted by e.

ABS.Setup takes as input $(1^\lambda, \mathcal{U})$. Let $R_\lambda := \{(\beta, \alpha) \in \mathbb{Z}_N \times \mathbb{Z}_N; \beta = \alpha^e\}$. Then $\mathrm{Instance}_R(1^\lambda)$ chooses an element $(\beta, \alpha) \in R_\lambda$ at random. **ABS.Setup** outputs a public key and a master secret key: $\mathrm{PK} = ((N, e, \beta), \mathcal{U}, \mu), \mathrm{MSK} = \alpha$.

ABS.KG outputs SK_S with signatures, for $i \in S$, $\sigma = (a_i = r_i^e,\ w_i = r_i\alpha^{c_i})$. Here we use a key k obtained by $k \leftarrow \mathit{Hash}_\mu(\alpha \parallel \tau)$, put $m_i = \tau \parallel i$, and $r_i \in \mathbb{Z}_N$ is chosen at random according to a random tape: $\mathit{PRF}_k(m_i)$, and c_i is obtained by $c_i \leftarrow \mathit{Hash}_\mu(a_i \parallel m_i)$. $\mathbf{\Sigma}^{\mathrm{stmtgen}}(\beta, a_i, c_i)$ is an algorithm that computes $x_i := a_i\beta^{c_i} \in \mathbb{Z}_N$.

The rest of protocol is executed according to $\mathbf{\Sigma}_f$ on input (x, w) and with the following setting.

$$\mathrm{CMT}_l = {r_l}^e, \mathrm{RES}_l = r_l(w_{\rho(l)})^{\mathrm{CHA}_l},$$
$$\text{Verification Equation} : {\mathrm{RES}_l}^e \stackrel{?}{=} \mathrm{CMT}_l\ (x_{\rho(l)})^{\mathrm{CHA}_l}.$$

B Non-interactive Witness-Indistinguishable Proof of Knowledge [21]

A *non-interactive witness-indistinguishable proof of knowledge system* (NIWIPoK, for short) $\mathit{\Pi} = (K, \mathcal{P}, \mathcal{V})$ for a language L_R is a protocol, where a PPT algorithm K, on input 1^λ, outputs crs called a *common reference string*; a PPT algorithm $\mathcal{P}$, on input $(x, w) \in R$ and crs, outputs π called a *proof*; and a PPT algorithm V, on input (x, π) and crs, outputs 1 (accept) or 0 (reject).

$\mathit{\Pi}$ must possess the following three properties.

Completeness. For any statement $x \in L_R$ and for any witness w such that $(x, w) \in R$, $\mathcal{P}$ with the witness w can make $\mathcal{V}$ accept on the statement x with probability 1:

$$\Pr[\pi \leftarrow \mathcal{P}(x, w) : \mathcal{V}(x, \pi) = 1] = 1.$$

Knowledge Soundness. There are an algorithm $\mathcal{KE}$ called a *knowledge extractor*, a function $\kappa : \{1, 0\}^* \to [1, 0]$ called a *knowledge error function* and a constant $c > 0$ that satisfy the following:

If there exists a PPT algorithm $\mathcal{A}$ that satisfies $p(x) := \Pr[\mathrm{crs} \leftarrow K(1^\lambda), \pi \leftarrow \mathcal{A}(\mathrm{crs}) : \mathcal{V}(x, \pi) = 1] > \kappa(x)$, then $\mathcal{KE}(x)$, employing $\mathcal{A}(x)$ as a subroutine that allows to be rewinded, outputs a witness w which satisfies $(x, w) \in R$ within an expected number of steps bounded by: $|x|^c/(p(x) - \kappa(x))$.

Witness-Indistinguishability. There is a polynomial-time algorithm S called a *simulator*, such that for any non-uniform polynomial-time algorithm $\mathcal{A}$ we have

$$\Pr[\mathrm{crs} \leftarrow K(1^\lambda) : \mathcal{A}(\mathrm{crs}) = 1] \approx \Pr[\mathrm{crs} \leftarrow S(1^\lambda) : \mathcal{A}(\mathrm{crs}) = 1]$$
$$\text{(computationally indistinguishable)}$$

and for any unbounded algorithm $\mathcal{A}$, we have

$$\begin{aligned}&\Pr[\mathrm{crs} \leftarrow S(1^\lambda), (x, w_0, w_1) \leftarrow \mathcal{A}(\mathrm{crs}), \pi \leftarrow \mathcal{P}(\mathrm{crs}, x, w_0) : \mathcal{A}(\pi) = 1] \\ =&\Pr[\mathrm{crs} \leftarrow S(1^\lambda), (x, w_0, w_1) \leftarrow \mathcal{A}(\mathrm{crs}), \pi \leftarrow \mathcal{P}(\mathrm{crs}, x, w_1) : \mathcal{A}(\pi) = 1]\end{aligned}$$

where $\big(R(x, w_0) = 1 \wedge R(x, w_1) = 1\big) \vee \big(R(x, w_0) = 0 \wedge R(x, w_1) = 0\big)$ holds.

C Our Non-interactive Witness-Indistinguishable Proof of Knowledge System

The Fiat-Shamir transform $\mathrm{FS}(\cdot)$ can be applied to any Σ-protocol $\mathbf{\Sigma}$ ([16, 1], see Section 2.1) to obtain a NIZKPoK system. As a NIZKPoK system is a NIWIPoK system, we obtain a NIWIPoK system. Here the generator K of common reference strings is becomes as follows.

$$K(1^\lambda) : \ \mu \leftarrow \mathit{Hashkeysp}(\lambda), \mathrm{crs} := \mu, \ \text{Return crs}$$

Hence we obtain the following theorem.

Theorem 4 *$FS(\mathbf{\Sigma}_f)$ is a non-interactive witness-indistinguishable proof of knowledge system for the language L_f. A knowledge extractor can be constructed in the random oracle model.*