Attribute-Based Signatures without Pairings via the Fiat-Shamir Paradigm

Hiroaki Anada *
Institute of Systems, Information Technologies and Nanotechnologies (ISIT)
Fukuoka SRP Center Bldg. 7F
Momochihama 2-1-22, Sawara-ku, Fukuoka, 814-0001, Japan
anada@isit.or.jp

Seiko Arita
Graduate School of Information Security, Institute of Information Security (IISEC)
2-14-1 Tsuruyacho, Kanagawa-ku
Yokohama, 221-0835, Japan
arita@iisec.ac.jp

Kouichi Sakurai †
Department of Informatics, Graduate School of ISEE
Faculty, Kyushu University
West Bldg. No.2
Moto-oka 744, Nishi-ku, Fukuoka, 819-0395, Japan
sakurai@csce.kyushu-u.ac.jp

ABSTRACT

We propose the first practical attribute-based signature (ABS) scheme with attribute privacy without pairings in the random oracle model. Our strategy is in the Fiat-Shamir paradigm; we first provide a generic construction of a *boolean proof system* of Σ-protocol type. Our boolean proof system is a generalization of the well-known OR-proof system; that is, it can treat any boolean formula instead of a single OR-gate. Then, by combining our boolean proof system with a credential bundle scheme of the Fiat-Shamir signature, we obtain a generic attribute-based identification (ABID) scheme of proof of knowledge. Finally, we apply the Fiat-Shamir transform to our ABID scheme to obtain a generic ABS scheme which possesses attribute privacy and can be proved to be secure in the random oracle model. Our ABS scheme can be constructed without pairings.

Categories and Subject Descriptors

F.1 [**Computation by Abstract Devices**]: Modes of Computation—*Interactive and Reactive Computation*

General Terms

Theory; Security

Keywords

access control; attribute; signature; proof system; boolean formula; identification; Fiat-Shamir transform

*He is also with IISEC
†He is also with ISIT.

AsiaPKC'14, June 3, 2014, Kyoto, Japan.

http://dx.doi.org/10.1145/2600694.2600696.

1. INTRODUCTION

Since the invention of digital signature scheme by Diffie and Hellman in 1976, there has been significant evolutions in the area and many functional variants have been proposed. A distinguished variant is *attribute-based signature* (ABS), which has been developed since 2008 [11, 13]. In ABS scheme, a message is associated with an access policy that limits signers by their attributes which the signers possess. The access policy is described with a boolean formula over those attributes. A signer with a set of authorized attributes can make a legitimate signature on the message only when his attributes satisfy the access policy. Then, a verifier can check whether the pair of the message and the signature is valid or not concerning the access policy. If the access policy limits accepting attributes to a single identity, ABS coincides with identity-based signature (IBS). In that sense, ABS is a natural extension of IBS.

One remarkable property of ABS is *attribute privacy* for signers. In the case that a verifier knows nothing about prover's attributes except that the prover's attributes satisfy the associated access policy of the signed message, the ABS scheme is called to have attribute privacy. Some ABS scheme ([11]) did not care attribute privacy but it is now desirable for ABS to possess.

On the other hand, The Fiat-Shamir transform [9] is now an established technique in interactive proof theory and cryptography. It transforms a public-coin interactive proof of knowledge system into a non-interactive proof of knowledge one. If the interactive proof system is an identification scheme, then the non-interactive one can be used as a digital signature scheme.

One of the most basic interactive proof systems is a proof of knowledge system by the Σ-protocol [7, 8]; it allows a prover, in a 3-move, to convince a verifier that the prover knows an answer (that is, a witness) of a hard problem instance. An extended notion, the OR-proof system [8], is a well-known Σ-protocol that allows a prover to convince a verifier that the prover knows one (or both) of witnesses of two hard problem instances, *not revealing which witness he knows.* The OR-proof system has been developed to achieve, for example, man-in-the-middle security.

In this paper, we will provide an attribute-based signature scheme via the Fiat-Shamir paradigm. First, we develop a

boolean proof system of Σ-protocol type, which is a generalization of the OR-proof system [8]. In our boolean proof system, a boolean formula f, in which boolean variables corresponds to attributes, is used. A prover has a set of witnesses each of which corresponds to an attribute. Then a prover can convince a verifier that the prover knows a set of witnesses that satisfy f.

Then we will proceed further into the Fiat-Shamir paradigm. By combining our boolean proof system with a credential bundle scheme of the Fiat-Shamir signature, we obtain a generic attribute-based identification (ABID) scheme of proof of knowledge. And finally, we apply the Fiat-Shamir transform to our ABID scheme to obtain a generic ABS scheme with attribute privacy, which can be proved to be secure in the random oracle model. According to the above strategy, we find a remarkable property that our ABS scheme can be constructed without pairing maps ([6]) (pairings, for short) with the expense that its security proof is in the random oracle model. As a result, our ABS scheme can be implemented with more efficiency than previous ABS schemes. To the best of authors' knowledge, previous ABS schemes with attribute privacy have been constructed with pairings.

1.1 Our Construction Idea

To construct the above boolean proof system, we will look into the technique employed in the OR-proof system [8] and expand it so as to treat any boolean formula (without negations), as follows.

First express a boolean formula (an access formula) as a binary tree (an access tree); that is, with its inner nodes being AND gates and OR gates and its leaf nodes being terms that map to attributes. A verification equation of $\mathbf{\Sigma}$ is assigned to each leaf node. Suppose that a challenge string $\textsc{Cha}$ of $\mathbf{\Sigma}$ is given. Assign $\textsc{Cha}$ to the root node. If the root node is an AND gate, assign the same string $\textsc{Cha}$ to two children. Else if the root node is an OR gate, divide $\textsc{Cha}$ into two random strings $\textsc{Cha}_{\textsc{L}}$ and $\textsc{Cha}_{\textsc{R}}$ which satisfy $\textsc{Cha} = \textsc{Cha}_{\textsc{L}} \oplus \textsc{Cha}_{\textsc{R}}$, and assign $\textsc{Cha}_{\textsc{L}}$ and $\textsc{Cha}_{\textsc{R}}$ to the left and right children, 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 each leaf node. Then, basically, the OR-proof technique assures we can either honestly execute Σ-protocol $\mathbf{\Sigma}$ or execute $\mathbf{\Sigma}$ in a simulated way. Only when a set of attributes satisfies the access tree the above procedure succeeds in satisfying verification equations for all leaf nodes. We call the above procedure a *boolean proof*. The boolean proof is a natural, but non-trivial extension of the OR-proof.

1.2 Our Contributions

As our first contribution, we provide a boolean proof system of a Σ-protocol, $\mathbf{\Sigma}_f$. Given a boolean formula f without negation and a Σ-protocol $\mathbf{\Sigma}$, we generically construct $\mathbf{\Sigma}_f$ so that $\mathbf{\Sigma}_f$ is a Σ-protocol to prove knowledge of a witness set that satisfies f. A remark is that to construct such a proof system in a 3-move with attribute privacy was difficult.

As our second contribution, we provide ABID schemes and ABS schemes *without pairings*. More precisely, As a Σ-protocol $\mathbf{\Sigma}$ can be constructed without pairings (for example, the case of Schnorr scheme [15, 5]), our generic, theoretical construction can produce a concrete, practical boolean proof system $\mathbf{\Sigma}_f$ and hence, an ABID scheme and an ABS scheme, without pairings. We must note that, in our construction of ABID and ABS, their security can be proved in the random oracle model. We also note that our ABS attains attribute privacy.

1.3 Related Work

The OR-proof system had been developed with the formalization of Σ-protocol in the present form [7, 8]. Our boolean proof system of Σ-protocol type can be seen as a natural extension of the OR-proof system.

At a high level, our ABS scheme is constructed by combining a credential bundle scheme [13] with our boolean proof system. This construction is compared to the generic construction of the ABS scheme by Maji et al. [13]. They started with a credential bundle scheme. Then they employed a non-interactive witness indistinguishable proof of knowledge (NIWIPoK) system to prove the knowledge of a credential bundle which satisfies a given access formula. The difference between our construction and their construction becomes apparent when we look into for what system the knowledge extractors (KEs) are needed. In our construction, KE is needed for our *interactive* proof system. In contrast, in their construction, KE is needed for a *non-interactive* WIPoK system.

From a practical point of view, it is notable that almost all attribute-based cryptographic primitives so far, originated in [10], and including ABS, employ linear secret sharing schemes (LSSS) ([3]) and pairings ([6]). Our approach for ABS is different from that construction.

2. PRELIMINARIES

The security parameter is denoted by λ. When an algorithm A with input a outputs z, we denote it as $z \leftarrow A(a)$, or, because of space limitation, $A(a) \to z$. 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 (TRUE) when $a \in S$ and 0 (FALSE) otherwise.

2.1 Access Formula [10]

Let $\mathcal{U} = \{\mathrm{att}_1, \ldots, \mathrm{att}_U\}$ be an attribute universe. We must distinguish two cases: the case that $\mathcal{U}$ is small (that is, $|\mathcal{U}| = U$ is bounded by some polynomial in λ) and the case that $\mathcal{U}$ is large (that is, U is not necessarily bounded). We assume the small case unless we state the large case explicitly.

Suppose that we are given an access policy as a boolean formula f over variables $\{X_{\mathrm{att}}\}_{\mathrm{att} \in \mathcal{U}}$ (for example, $f = X_{\mathrm{att}_1} \wedge ((X_{\mathrm{att}_2} \wedge X_{\mathrm{att}_3}) \vee X_{\mathrm{att}_4})$). That is, each term X_{att} is a predicate which, on input $S \in 2^{\mathcal{U}}$, takes a boolean value according to whether $\mathrm{att} \in S$ or not. Then the boolean output of f at S is evaluated according to boolean connectives, that is, AND gates ($\wedge$-gates) or OR gates ($\vee$-gates). Hence f is a function: $f : 2^{\mathcal{U}} \to \{1, 0\}$. We call the boolean formula f an *access formula* over $\mathcal{U}$.

In this paper, we do *not consider NOT gates (¬)*, that is, we only consider monotone boolean formulas[1].

2.1.1 Access Tree

We consider in this paper a finite binary tree $\mathcal{T}$, that is, a tree that has finite number of nodes and each non-leaf node has two branches. For a tree $\mathcal{T}$, let $\mathrm{Nd}(\mathcal{T})$, $\mathrm{rt}(\mathcal{T})$, $\mathrm{Lf}(\mathcal{T})$, $\mathrm{iNd}(\mathcal{T})$ and $\mathrm{tNd}(\mathcal{T})$ 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) in $\mathcal{T}$, respectively. An access formula f can be represented by a finite binary tree $\mathcal{T}_f$. Each leaf node corresponds to a term X_{att} in f in one-to-one way. Each inner node represents an operator, $\wedge$-gate or $\vee$-gate, in f. An attribute map $\mathrm{att}(\cdot) : \mathrm{Lf}(\mathcal{T}) \to \mathcal{U}$ is defined, for $\mathrm{lf} \in \mathrm{Lf}(\mathcal{T})$, as $\mathrm{att}(\mathrm{lf}) :=$ (the attribute att of the term X_{att}).

If $\mathcal{T}$ is of height greater than 0, $\mathcal{T}$ has two subtrees whose root nodes are two children of $\mathrm{rt}(\mathcal{T})$. 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.

2.2 Proof of Knowledge System [4, 8]

Let R be a binary polynomial-time relation $R = \{(x, w)\} \subset \{1,0\}^* \times \{1,0\}^*$. x is called a *statement* and w is called a *witness* for the statement x. We assume that, given a statement x as input, any PPT algorithm can only output a witness w satisfying $(x, w) \in R$ with a negligible probability. If this assumption holds, we say that R is a *hard relation* ([8]).

A *proof of knowledge system* (PoK for short) is a protocol between interactive PPT algorithms P and V on initial input $(x, w) \in R$ for P and x for V.

Completeness. An honest prover P with a legitimate witness w can make V accept with probability 1.

Knowledge Soundness. There is a PPT algorithm called a *knowledge extractor*, which, given a statement x and employing P as a subroutine, can compute a witness w satisfying $(x, w) \in R$ with at most a negligible error probability.

2.3 Σ-protocol [7, 8]

Let R be a binary polynomial-time relation $R = \{(x, w)\} \subset \{1,0\}^* \times \{1,0\}^*$.

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

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

Σ-protocol satisfies three constraints:

Completeness. An honest prover P with a legitimate witness w can make V accept with probability 1.

Special Soundness. Any PPT algorithm 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 *knowledge extractor*, $\mathbf{\Sigma}^{\mathrm{KE}}$, which, given a statement x and using 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 $(\mathrm{Cmt}, \mathrm{Cha}, \mathrm{Res})$ and $(\mathrm{Cmt}, \mathrm{Cha}', \mathrm{Res}')$ with $\mathrm{Cha} \neq \mathrm{Cha}'$.

Honest-Verifier Zero-Knowledge. Given a statement x and a *random* challenge $\mathrm{Cha} \leftarrow \mathbf{\Sigma}^2(\lambda)$, we can produce in polynomial-time, without knowing the witness w, an accepting conversation $(\mathrm{Cmt}, \mathrm{Cha}, \mathrm{Res})$ 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 $(\mathrm{Cmt}, \mathrm{Res}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x, \mathrm{Cha})$.

Σ-protocols are known to be proofs of knowledge system ([8]).

We will use in this paper a property that both a prover and a verifier can compute, in polynomial-time, a new statement x' which has the response message Res as its corresponding witness, on input $(x, \mathrm{Cmt}, \mathrm{Cha})$. We denote the algorithm as $\mathbf{\Sigma}^{\mathrm{stmtforRes}}$:

$$x' \leftarrow \mathbf{\Sigma}^{\mathrm{stmtforRes}}(x, \mathrm{Cmt}, \mathrm{Cha})$$
$$\text{s.t. } (\mathrm{Cmt}, \mathrm{Cha}, \mathrm{Res}) : \text{an accepting conversation}$$
$$\wedge\, (x', \mathrm{Res}) \in R.$$

Known Σ-protocols such as the Schnorr protocol and the Guillou-Quisquater protocol [15, 5] possess this property.

2.3.1 The OR-proof System[8]

Suppose that a Σ-protocol $\mathbf{\Sigma}$ on a relation R is given. Consider the following relation.

$$R_{\mathrm{OR}} = \{(\{x_0, x_1\}, \{w_i\}_{i \in S}) \in \{1,0\}^* \times \{1,0\}^*;$$
$$(0 \in S \vee 1 \in S) \ \wedge\ {}^{\forall} i \in \{0,1\} \cap \mathcal{S}, (x_i, w_i) \in R\}.$$

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

2.3.2 The Fiat-Shamir Transform [1]

Employing a cryptographic hash function with collision resistance, $H_\lambda(\cdot) : \{1,0\}^* \to \{1,0\}^{l(\lambda)}$, a Σ-protocol $\mathbf{\Sigma}$ can be transformed into a digital signature scheme as follows. Given a message $m \in \{1,0\}^*$, execute: $a \leftarrow \mathbf{\Sigma}^1(x, w)$, $c \leftarrow H_\lambda(m \parallel a)$, $z \leftarrow \mathbf{\Sigma}^3(x, w, a, c)$. Then $\sigma := (a, z)$ is a signature on m. We denote the above signing algorithm as $\mathrm{FS}^{\mathrm{sign}}_{\mathbf{\Sigma}}(x, w, m) \to (a, z) =: \sigma$. The verification algorithm is given as follows: $\mathrm{FS}^{\mathrm{vrfy}}_{\mathbf{\Sigma}}(x, m, \sigma) : c \leftarrow H_\lambda(m \parallel a)$, Return $b \leftarrow \mathbf{\Sigma}^{\mathrm{vrfy}}(x, a, c, z)$.

The signature scheme $\mathrm{FS}_{\mathbf{\Sigma}} = (R, \mathrm{FS}^{\mathrm{sign}}_{\mathbf{\Sigma}}, \mathrm{FS}^{\mathrm{vrfy}}_{\mathbf{\Sigma}})$ can be proved, in the random oracle model, to be *secure in the existential unforgeability game against chosen-message attacks*

[1]This limitation can be removed by adding *negation attributes* to $\mathcal{U}$ for each attribute in the original $\mathcal{U}$ (but as a result, the size $|\mathcal{U}|$ doubles).

if and only if underlying Σ-protocol $\mathbf{\Sigma}$ is secure against *passive attacks* [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).

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

2.4 Credential Bundle Scheme [13]

Credential bundle is an extended notion of digital signature. Suppose that we are given a digital signature scheme $(\mathrm{KG}, \mathrm{Sign}, \mathrm{Vrfy})$. To construct a credential bundle scheme, first choose a string τ, a *tag*. τ can be chosen as a random string or a publicly known string such as an e-mail address. Then, for a set of messages $m_1, \ldots, m_n \in \{1,0\}^*$, execute Sign on each *tagged message* $(\tau \parallel m_i), i = 1, \ldots, n$. Verify is applied in a obvious way.

2.5 Attribute-Based Identification Scheme [2]

In this paper, we will describe *verifier-policy* attribute-based identification schemes [2]. An attribute-based identification scheme, ABID, consists of four PPT algorithms: (Setup, KG, P, V).

Setup$(\lambda, \mathcal{U}) \rightarrow (\mathbf{PK}, \mathbf{MSK})$**.** Setup 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.

$\mathbf{KG}(\mathbf{PK}, \mathbf{MSK}, S) \rightarrow \mathbf{SK}_S$**.** A key-generation algorithm KG takes as input the public key PK, the master secret key MSK and an attribute set $S \subset \mathcal{U}$. It outputs a secret key SK_S corresponding to S.

$\mathbf{P}(\mathbf{PK}, \mathbf{SK}_S)$ **and** $\mathbf{V}(\mathbf{PK}, f)$**.** P and V are interactive algorithms called a *prover* and a *verifier*, respectively. P takes as input the public key PK and the secret key SK_S. Here the secret key SK_S is given to P by an authority that runs KG(PK,MSK,S). V takes as input the public key PK and an access formula f. P is provided V's access formula f by the first move. P and V interact with each other for at most constant rounds. Then, V returns its decision 1 or 0. When it is 1, we say that V *accepts* P for f. When it is 0, we say that V *rejects* P for f. We demand correctness of ABID that for any λ, and if $f(S) = 1$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_S \leftarrow \mathrm{KG}(\mathrm{PK}, \mathrm{MSK}, S); b \leftarrow \langle \mathrm{P}(\mathrm{PK}, \mathrm{SK}_S), \mathrm{V}(\mathrm{PK}, f)\rangle : b = 1] = 1$.

2.5.1 Attacks on ABID and Security

An adversary $\mathcal{A}$'s objective is impersonation. $\mathcal{A}$ tries to make a verifier 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.

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

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 $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}.)$, $j = 1, \ldots, q'_\mathrm{p}, \ldots, q_\mathrm{p}$, 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 $\mathrm{P}_j(\mathrm{PK}, \mathrm{SK}_{S_j})$ 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 ABID in the game of concurrent attack is defined as

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

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

The game of *passive attack* on ABID is defined by replacing concurrent provers $\mathbf{P}_j(\mathrm{PK}, \mathrm{SK}.)|_{j=1}^{q_\mathrm{p}}$ with transcript oracle Transc in $\mathbf{Exprmt}^{\mathrm{ca}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$. In a transcript query, giving a pair (S_j, f) of an attribute set and an access formula, $\mathcal{A}$ queries $\mathrm{Transc}(\mathbf{P}(\mathrm{PK}, \mathrm{SK}.), \mathbf{V}(\mathrm{PK}, \cdot))$ for a whole transcript of messages interacted between $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{S_j})$ and $\mathbf{V}(\mathrm{PK}, f_j)$. The *advantage* $\mathbf{Adv}^{\mathrm{pa}}_{\mathcal{A},\mathtt{ABID}}(\lambda)$ and security are defined in the same way as the concurrent case. Note that concurrent security means passive security; for any PPT $\mathcal{A}$, there exists a PPT $\mathcal{B}$ that satisfies the following inequality.

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

2.5.2 Attribute Privacy of ABID

Consider the following experiment $\mathbf{Exprmt}^{\text{att-prv}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$. (In the experiment, an adversary $\mathcal{A}$ interacts with $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_{S_b})$ as a verifier with f^*.)

$\mathbf{Exprmt}^{\text{att-prv}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U})$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U}), (S_0, S_1, f^*) \leftarrow \mathcal{A}(\mathrm{PK})$
s.t. $(f^*(S_0) = f^*(S_1) = 1) \vee (f^*(S_0) = f^*(S_1) = 0)$
$\mathrm{SK}_{S_0} \leftarrow \mathbf{KG}(\mathrm{PK}, \mathrm{MSK}, S_0), \mathrm{SK}_{S_1} \leftarrow \mathbf{KG}(\mathrm{PK}, \mathrm{MSK}, S_1)$
$b \leftarrow \{1,0\}, \hat{b} \leftarrow \mathcal{A}^{\mathbf{P}(\mathrm{PK},\mathrm{SK}_{S_b})}(\mathrm{PK}, \mathrm{SK}_{S_0}, \mathrm{SK}_{S_1})$
If $b = \hat{b}$ Return Win else Return Lose

We say that ABID has *attribute privacy* if, for any PPT $\mathcal{A}$, the following advantage is negligible in λ.

$$\mathbf{Adv}^{\text{att-prv}}_{\mathcal{A},\mathtt{ABID}}(\lambda) \stackrel{\mathrm{def}}{=} |\Pr[\mathbf{Exprmt}^{\text{att-prv}}_{\mathcal{A},\mathtt{ABID}}(\lambda, \mathcal{U}) \text{ returns Win}] - 1/2|.$$

2.6 Attribute-Based Signature Scheme [14]

An attribute-based signature scheme, ABS, consists of four PPT algorithms: (Setup, KG, Sign, Vrfy).

Setup$(\lambda, \mathcal{U}) \rightarrow (\mathbf{PK}, \mathbf{MSK})$**.** Setup 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.

$\mathbf{KG}(\mathbf{PK}, \mathbf{MSK}, S) \rightarrow \mathbf{SK}_S$**.** A key-generation algorithm 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.

$\mathbf{Sign}(\mathbf{PK}, \mathbf{SK}_S, (m, f)) \rightarrow \sigma$**.** A signing algorithm Sign takes as input a public key PK, a private secret key SK_S corre-

sponding to an attribute set S, a pair (m, f) of a message $\in \{1,0\}^*$ and an access formula. It outputs a signature σ.
Vrfy$(\mathbf{PK}, (m, f), \sigma)$**.** A verification algorithm Vrfy 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, and if $f(S) = 1$, then $\Pr[(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathrm{Setup}(\lambda, \mathcal{U}); \mathrm{SK}_S \leftarrow \mathrm{KG}(\mathrm{PK}, \mathrm{MSK}, S); \sigma \leftarrow \mathbf{Sign}(\mathrm{PK}, \mathrm{SK}_S, (m, f)); b \leftarrow \mathbf{Vrfy}(\mathrm{PK}, (m, f), \sigma) : b = 1] = 1$.

2.6.1 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 $((m^*, f^*), \sigma^*)$ of a message, a target access policy and a signature. The following experiment $\mathbf{Exprmt}^{\mathrm{eufcma}}_{\mathcal{F},\mathtt{ABS}}(\lambda, \mathcal{U})$ of a forger $\mathcal{F}$ defines the *existential unforgeability game by chosen-message attack* on ABS.

$\mathbf{Exprmt}^{\mathrm{eufcma}}_{\mathcal{F},\mathtt{ABS}}(\lambda, \mathcal{U})$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U})$
$((m^*, f^*), \sigma^*) \leftarrow \mathcal{F}^{\mathcal{KG}(\mathrm{PK},\mathrm{MSK},\cdot), \mathcal{SIGN}(\mathrm{PK},\mathrm{SK}_\cdot,(\cdot,\cdot))}$
If $\mathbf{Vrfy}(\mathrm{PK}, (m^*, f^*), \sigma^*) = 1$ 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, $\mathcal{A}$ 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 $\mathcal{SIGN}(\mathrm{PK}, \mathrm{SK}_\cdot, (\cdot, \cdot))$ for a signature σ that satisfies $\mathbf{Vrfy}(\mathrm{PK}, (m, f), \sigma) = 1$ 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, issuing of key-extraction queries and signing queries. Two restrictions are imposed on $\mathcal{F}$ concerning f^*. In key-extraction queries, each attribute set S_i must satisfy $f^*(S_i) = 0$. 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 existential unforgeability game by chosen-message attack is defined as

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

ABS is called *secure in the existential unforgeability game against chosen-message attacks* if, for any PPT $\mathcal{F}$, $\mathbf{Adv}^{\mathrm{eufcma}}_{\mathcal{F},\mathtt{ABS}}(\lambda)$ is negligible in λ.

2.6.2 Attribute Privacy of ABS

Consider the following experiment $\mathbf{Exprmt}^{\mathrm{att\text{-}prv}}_{\mathcal{A},\mathtt{ABS}}(\lambda, \mathcal{U})$.

$\mathbf{Exprmt}^{\mathrm{att\text{-}prv}}_{\mathcal{A},\mathtt{ABS}}(\lambda, \mathcal{U})$:
$(\mathrm{PK}, \mathrm{MSK}) \leftarrow \mathbf{Setup}(\lambda, \mathcal{U}), (S_0, S_1, f^*) \leftarrow \mathcal{A}(\mathrm{PK})$
s.t. $(f^*(S_0) = f^*(S_1) = 1) \vee ((f^*(S_0) = f^*(S_1) = 0)$
$\mathrm{SK}_{S_0} \leftarrow \mathbf{KG}(\mathrm{PK}, \mathrm{MSK}, S_0), \mathrm{SK}_{S_1} \leftarrow \mathbf{KG}(\mathrm{PK}, \mathrm{MSK}, S_1)$
$b \leftarrow \{1,0\}, \hat{b} \leftarrow \mathcal{A}^{\mathcal{SIGN}(\mathrm{PK},\mathrm{SK}_{S_b},\cdot,\cdot)}(\mathrm{PK}, \mathrm{SK}_{S_0}, \mathrm{SK}_{S_1})$
If $b = \hat{b}$ Return Win else Return Lose

We say that ABS has *attribute privacy* if, for any PPT $\mathcal{A}$, the following advantage of $\mathcal{A}$ is negligible in λ.

$$\mathbf{Adv}^{\mathrm{att\text{-}prv}}_{\mathcal{A},\mathtt{ABS}}(\lambda) \stackrel{\mathrm{def}}{=} |\Pr[\mathbf{Exprmt}^{\mathrm{att\text{-}prv}}_{\mathcal{A},\mathtt{ABS}}(\lambda, \mathcal{U}) \text{ returns Win}] - 1/2|.$$

2.7 Pseudorandom Function Family [12]

A pseudorandom function family, $\{PRF_\kappa\}_{\kappa \in PRFkeysp(\lambda)}$, is a function family in which each function $PRF_\kappa : \{1,0\}^* \to \{1,0\}^*$ is an efficiently-computable function that looks random to any polynomial-time distinguisher, where κ is called a key and $PRFkeysp(\lambda)$ is called a key space. (See more detail in [12].)

3. DEFINITION OF OUR BOOLEAN PROOF SYSTEM OF Σ-PROTOCOL TYPE

In this section, we propose a notion of *boolean proof system* in the form of a Σ-protocol, $\mathbf{\Sigma}_f$, where f is a given access formula. Our $\mathbf{\Sigma}_f$ is a generalization of the OR-proof system $\mathbf{\Sigma}_{\mathrm{OR}}$ [8]; that is, it treats any boolean formula instead of a single OR-gate. In addition, as is the case for usual Σ-protocol $\mathbf{\Sigma}$, our boolean proof system $\mathbf{\Sigma}_f$ is a proof of knowledge system; that is, it possesses a knowledge extractor that extracts a witness set which satisfies the boolean formula f.

Let R be a binary polynomial-time relation $R = \{(x, w)\} \subset \{1,0\}^* \times \{1,0\}^*$. Let f be an access formula over $\mathcal{U}$. Then we create a new relation R_f:

$$R_f \stackrel{\mathrm{def}}{=} \{(X := \{x_{\mathrm{att}}\}_{\mathrm{att} \in \mathrm{Att}(f)}, W := \{w_{\mathrm{att}}\}_{\mathrm{att} \in S}) \in (\{1,0\}^*)^2; f(S) = 1 \ \wedge \ {}^\forall \mathrm{att} \in \mathrm{Att}(f) \cap S, (x_{\mathrm{att}}, w_{\mathrm{att}}) \in R\}.$$

Note that R_f is a generalization of the relation R_{OR}.

Then a *boolean proof system* $\mathbf{\Sigma}_f$ on the relation R_f is defined as a 3-move protocol between interactive PPT algorithms P and V on initial input $(X := \{x_{\mathrm{att}}\}, W := \{w_{\mathrm{att}}\}) \in R_f$ for P, and X for V. P sends the first message called a commitment Cmt, then V sends a random bit string called a challenge Cha, and P answers with a third message called a response Res. Then V applies a local decision test on $(X, \mathrm{Cmt}, \mathrm{Cha}, \mathrm{Res})$ to return accept (1) or reject (0). Here Cha is chosen at random from $\mathrm{ChaSp}(\lambda) := \{1,0\}^{l(\lambda)}$ with $l(\cdot)$ being a super-log function.

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

To be a Σ-protocol, $\mathbf{\Sigma}_f$ must satisfy three constraints:
Completeness An honest prover P with a legitimate witness set W can make V accept with probability 1.
Special Soundness. Any PPT algorithm P^* without any witness set (called a cheating prover) can only respond for one possible challenge Cha. In other words, there is a PPT algorithm called a *knowledge extractor*, $\mathbf{\Sigma}^{\mathrm{KE}}_f$, which, given a statement X and using P^* as a subroutine, can compute a witness set W satisfying $(X, W) \in R_f$ with at most a negligible error probability, from two accepting conversations of the form $(\mathrm{Cmt}, \mathrm{Cha}, \mathrm{Res})$ and $(\mathrm{Cmt}, \mathrm{Cha}', \mathrm{Res}')$ with $\mathrm{Cha} \neq \mathrm{Cha}'$.
Honest-Verifier Zero-Knowledge. Given a statement set X and a *random* challenge $\mathrm{Cha} \leftarrow \mathbf{\Sigma}^2_f(\lambda)$, we can produce in

polynomial-time, without knowing the witness set 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 $(\textsc{Cmt}, \textsc{Res}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(X, \textsc{Cha})$.

The boolean proof system $\mathbf{\Sigma}_f$ constructed in the above way can be proved to be a proof of knowledge system. The reason is the same as the reason of usual Σ-protocol ([8]).

4. CONSTRUCTION OF OUR BOOLEAN PROOF SYSTEM

In this section, we construct our boolean proof system $\mathbf{\Sigma}_f$ from a given Σ-protocol $\mathbf{\Sigma}$ and an access formula f. Because of space limitation, all proofs for Proposition, Lemma and Theorem are hereafter omitted.

Fig. 1 shows our protocol $\mathbf{\Sigma}_f$. Basically $\mathbf{\Sigma}_f$ is a 3-move protocol between interactive PPT algorithms P and V on initial input $(X := \{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)}, W := \{w_{\mathrm{att}}\}_{\mathrm{att}\in S}) \in R_f$ for P and X for V.

Evaluation of Satisfiability. The prover 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 lf, we label lf with $v_{\mathrm{lf}} = 1$ if $\mathrm{att}(\mathrm{lf}) \in S$ and $v_{\mathrm{lf}} = 0$ otherwise. For each inner node nd, we label nd according to AND/OR evaluation result of two labels of its two children. 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 $\mathrm{rt}(\mathcal{T})$ is an $\wedge$-node,
then Return $v_{\mathrm{rt}(\mathcal{T})} := (\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S) \wedge \mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S))$
else if $\mathrm{rt}(\mathcal{T})$ is an $\vee$-node,
then Return $v_{\mathrm{rt}(\mathcal{T})} := ((\mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{L}}, S) \vee \mathbf{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_{\mathrm{R}}, S))$
else if $\mathrm{rt}(\mathcal{T})$ is a leaf node,
then Return $v_{\mathrm{rt}(\mathcal{T})} := (\mathrm{att}(\mathrm{rt}(\mathcal{T})) \stackrel{?}{\in} S)$

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 lf a value $\textsc{Cmt}_{\mathrm{lf}}$; If $v_{\mathrm{lf}} = 1$, then $\textsc{Cmt}_{\mathrm{lf}}$ is computed honestly according to $\mathbf{\Sigma}^1$. Else if $v_{\mathrm{lf}} = 1$, then $\textsc{Cmt}_{\mathrm{lf}}$ is computed in the simulated way according to $\mathbf{\Sigma}^{\mathrm{sim}}$. Other values, $(\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)}$ and $(\textsc{Res}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)}$), are needed for the simulation. Note that a distinguished symbol '$*$' is used for those other values to indicate the honest computation.

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

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

Response. Prover's computation of a response value for each leaf node is described in Fig. 3. Basically, the algorithm $\mathbf{\Sigma}_f^3$ runs for every node from the root node to each leaf node, recursively. As a result, $\mathbf{\Sigma}_f^3$ generates values, $(\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)}$ and $(\textsc{Res}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)}$). Note that all challenge values $(\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)}$ are completed according to the "division rule" described in Section 1.1.

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

$\mathbf{\Sigma}_f^{\mathrm{vrfy}}(X, \mathcal{T}, \textsc{Cha},$
$(\textsc{Cmt}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T})}, (\textsc{Res}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})})$:
Return($\mathbf{VrfyCha}(\mathcal{T}, \textsc{Cha}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T})})$
$\wedge\ \mathbf{VrfyRes}(X, \mathcal{T}, (\textsc{Cmt}, \textsc{Cha}, \textsc{Res})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})})$)

$\mathbf{VrfyCha}(\mathcal{T}, \textsc{Cha}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T})})$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If $\mathrm{rt}(\mathcal{T})$ is an $\wedge$-node,
then Return ($\textsc{Cha} \stackrel{?}{=} \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} \wedge \textsc{Cha} \stackrel{?}{=} \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}$
$\wedge\ \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{L}}, \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{L}})})$
$\wedge\ \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{R}}, \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{R}})})$)
else if $\mathrm{rt}(\mathcal{T})$ is an $\vee$-node,
then Return ($\textsc{Cha} \stackrel{?}{=} \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} \oplus \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}$
$\wedge\ \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{L}}, \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{L}})})$
$\wedge\ \mathbf{VrfyCha}(\mathcal{T}_{\mathrm{R}}, \textsc{Cha}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, (\textsc{Cha}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{R}})})$)
else if $\mathrm{rt}(\mathcal{T})$ is a leaf node,
then Return ($\textsc{Cha} \stackrel{?}{\in} \textsc{ChaSp}(\lambda)$)

$\mathbf{VrfyRes}(X, \mathcal{T}, (\textsc{Cmt}, \textsc{Cha}, \textsc{Res})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})})$:
For $\mathrm{lf} \in \mathrm{Lf}(\mathcal{T})$
If $\mathbf{\Sigma}^{\mathrm{vrfy}}(x_{\mathrm{att}(\mathrm{lf})}, \textsc{Cmt}_{\mathrm{lf}}, \textsc{Cha}_{\mathrm{lf}}, \textsc{Res}_{\mathrm{lf}}) = 0$, then Return (0)
Return (1)

Now we have to check that $\mathbf{\Sigma}_f$ is certainly a boolean proof system of Σ-protocol type.

Proposition 1 (Completeness). *Completeness holds for our $\mathbf{\Sigma}_f$. More precisely, Suppose that $v_{rt(\mathcal{T}_f)} = 1$. Then, for every node in $Nd(\mathcal{T}_f)$, either $v_{nd} = 1$ or $\textsc{Cha}_{nd} \neq *$ holds after executing $\mathbf{\Sigma}_f^1$.*

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

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

Proposition 3 (Honest Verifier Zero-Knowledge). *Honest verifier zero-knowledge property holds for our $\mathbf{\Sigma}_f$.*

Theorem 1. *$\mathbf{\Sigma}_f$ is certainly a boolean proof system of Σ-protocol type on the relation R_f.*

5. APPLICATION TO ATTRIBUTE-BASED IDENTIFICATION

In this section, by combining our boolean proof system $\mathbf{\Sigma}_f$ with a credential bundle scheme of the Fiat-Shamir signature $\mathrm{FS}_{\mathbf{\Sigma}}$, we obtain an attribute-based identification scheme `ABID` of proof of knowledge, which has collusion resistance against collecting private secret keys. Our ABID scheme `ABID` has a feature that it can be constructed without pairings.

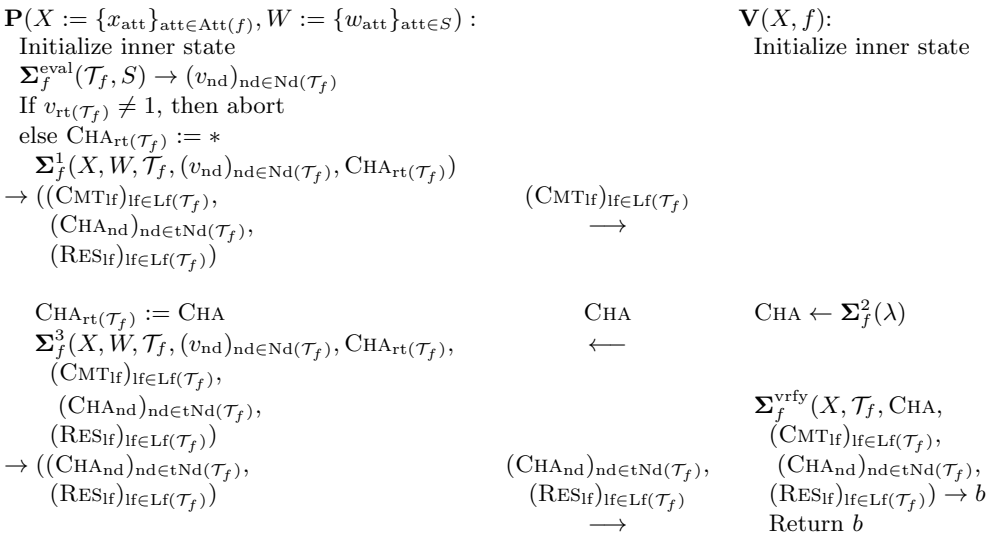

Figure 1: Our boolean proof system $\mathbf{\Sigma}_f$ on a relation $R_f = \{(X, W)\}$.

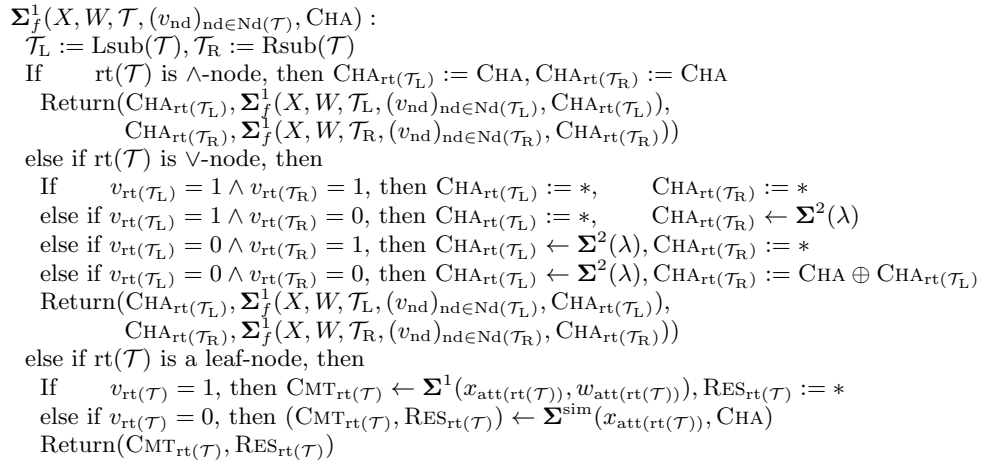

Figure 2: The subroutine $\mathbf{\Sigma}_f^1$ of our $\mathbf{\Sigma}_f$.

5.1 Our ABID Scheme

Our $\mathtt{ABID} = (\mathrm{Setup}, \mathrm{KG}, \mathrm{P}, \mathrm{V})$ is described as follows.

Setup$(\lambda, \mathcal{U})$:
$(x_{\mathrm{M}}, w_{\mathrm{M}}) \leftarrow \mathrm{Instance}_R(\lambda), \mathrm{PK} := (x_{\mathrm{M}}, \mathcal{U}), \mathrm{MSK} := w_{\mathrm{M}}$
Return$(\mathrm{PK}, \mathrm{MSK})$

Setup computes a public key PK and a master secret key MSK by running $\mathrm{Instance}_R(\lambda)$, which chooses an element $(x_{\mathrm{M}}, w_{\mathrm{M}})$ that corresponds to the security parameter λ, at random from a fixed hard relation $R = \{(x, w)\}$.

KG$(\mathrm{PK}, \mathrm{MSK}, S)$:
$\kappa \leftarrow \mathit{PRFkeysp}(\lambda), \ \tau \leftarrow \{1, 0\}^{\lambda}$
For att $\in S$: $m_{\mathrm{att}} := (\tau \parallel \mathrm{att})$
$\sigma_{\mathrm{att}} := (a_{\mathrm{att}}, w_{\mathrm{att}}) \leftarrow \mathrm{FS}_{\mathbf{\Sigma}}^{\mathrm{sign}}(x_{\mathrm{M}}, w_{\mathrm{M}}, m_{\mathrm{att}})$
$\mathrm{SK}_S := (\kappa, \tau, (\sigma_{\mathrm{att}})_{\mathrm{att}\in S}),$ Return SK_S.

KG first chooses a random string, a tag τ, at random. Then KG applies the credential bundle technique [13] for each message $m_{\mathrm{att}} := (\tau \parallel \mathrm{att}), \mathrm{att} \in S$. Here we employ the Fiat-Shamir signing algorithm $\mathrm{FS}_{\mathbf{\Sigma}}^{\mathrm{sign}}$ that is obtained from the same Σ-protocol $\mathbf{\Sigma}$.

Supp$(\mathrm{PK}, \mathrm{SK}_S, f)$:
For att $\in \mathrm{Att}(f)$
If att $\notin S$, then $c_{\mathrm{att}} \leftarrow \mathit{PRF}_{\kappa}(0 \parallel \mathrm{att})$
$(a_{\mathrm{att}}, z_{\mathrm{att}}) \leftarrow \mathbf{\Sigma}^{\mathrm{sim}}(x_{\mathrm{M}}, c_{\mathrm{att}}; \mathit{PRF}_{\kappa}(1 \parallel \mathrm{att}))$
Return $(a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}$

P uses supplementary algorithm, Supplement, to generate values a_{att} for all att $\in \mathrm{Att}(f)$.

StmtGen$(\mathrm{PK}, \tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)})$:
For att $\in \mathrm{Att}(f)$:
$m_{\mathrm{att}} := (\tau \parallel \mathrm{att}), c_{\mathrm{att}} \leftarrow H_{\lambda}(m_{\mathrm{att}} \parallel a_{\mathrm{att}})$
$x_{\mathrm{att}} \leftarrow \mathbf{\Sigma}^{\mathrm{stmtforRES}}(x_{\mathrm{M}}, a_{\mathrm{att}}, c_{\mathrm{att}})$
Return $\{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)}$

$\boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T})}, \mathrm{CHA}, (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T})}, (\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T})})$:
$\mathcal{T}_{\mathrm{L}} := \mathrm{Lsub}(\mathcal{T}), \mathcal{T}_{\mathrm{R}} := \mathrm{Rsub}(\mathcal{T})$
If rt($\mathcal{T}$) is $\wedge$-node, then $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}$
Return$(\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, \boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}_{\mathrm{L}}, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{L}})}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{L}})}),$
$\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, \boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}_{\mathrm{R}}, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{R}})}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{R}})}))$
else if rt($\mathcal{T}$) is $\vee$-node, then
If $v_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} \leftarrow \boldsymbol{\Sigma}^2(\lambda)$, $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}$
else if $v_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} = 1 \wedge v_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}$
else if $v_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} = 1$, then $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}$, $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA} \oplus \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}$
else if $v_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} = 0 \wedge v_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} = 0$, then $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})} := \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}$, $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})} := \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}$
Return$(\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, \boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}_{\mathrm{L}}, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{L}})}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{L}})}, (\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{L}})}),$
$\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, \boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}_{\mathrm{R}}, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{R}})}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_{\mathrm{R}})}, (\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_{\mathrm{R}})}))$
else if rt($\mathcal{T}$) is a leaf-node, then
If $v_{\mathrm{rt}(\mathcal{T})} = 1$, then $\mathrm{RES}_{\mathrm{rt}(\mathcal{T})} \leftarrow \boldsymbol{\Sigma}^3(x_{\mathrm{att}(\mathrm{rt}(\mathcal{T}))}, w_{\mathrm{att}(\mathrm{rt}(\mathcal{T}))}, \mathrm{CMT}_{\mathrm{rt}(\mathcal{T})}, \mathrm{CHA})$
else if $v_{\mathrm{rt}(\mathcal{T})} = 0$, then do nothing
Return$(\mathrm{RES}_{\mathrm{rt}(\mathcal{T})})$

Figure 3: The subroutine $\boldsymbol{\Sigma}_f^3$ of our $\boldsymbol{\Sigma}_f$.

P and V utilizes a statement-generator algorithm, StmtGen. This generates for each att $\in \mathrm{Att}(f)$ a statement x_{att} that makes the output w_{att} of KG be a corresponding witness. Note that we employ here the algorithm $\boldsymbol{\Sigma}^{\mathrm{stmtforRES}}$ which is associated with $\boldsymbol{\Sigma}$, and whose existence is assured by assumption (see Section 2.3).

Finally, a whole ABID scheme is obtained by adding the following procedures (1) and (2) to our Σ-protocol $\boldsymbol{\Sigma}_f$.
(1) In $\mathbf{P}(\mathrm{PK}, \mathrm{SK}_S)$, add the following procedures to give a statement set and a witness set (X, W) to $\boldsymbol{\Sigma}_f^1$.

$$\mathbf{Supp}(\mathrm{PK}, \mathrm{SK}_S, f) \to (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}$$
$$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}) \to \{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)} =: X, W := \{w_{\mathrm{att}}\}_{\mathrm{att}\in S}.$$

At the first move, P sends to the verifier V not only commitments $(\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)}$ but also elements τ and $(a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}$.
(2) In $\mathbf{V}(\mathrm{PK}, f)$, add the following procedure to give a statement set X to $\boldsymbol{\Sigma}_f^{\mathrm{vrfy}}$.

$$\mathbf{StmtGen}(\mathrm{PK}, \tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}) \to \{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)} =: X.$$

THEOREM 2. *Suppose that a given Σ-protocol $\boldsymbol{\Sigma}$ possesses a polynomial-time algorithm $\boldsymbol{\Sigma}^{stmtforRES}$. Then our ABID is also a Σ-protocol on the relation $R_f := \{(X, W)\}$.*

5.2 Security of Our ABID

THEOREM 3. *If the employed signature scheme $FS_{\boldsymbol{\Sigma}}$ is secure in the existential unforgeability game 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).*

$$\mathbf{Adv}^{ca}_{\mathcal{A},ABID}(\lambda) \leq (\mathbf{Adv}^{eufcma}_{\mathcal{F},FS_{\boldsymbol{\Sigma}}}(\lambda))^{1/2} + neg(\lambda).$$

5.3 Discussion

5.3.1 Attribute-Based Proof of Knowledge

Our ABID is a proof of knowledge system. That is, for a fixed access policy f, a PPT knowledge extractor can be constructed, which extracts a secret key SK_{S^*} for some attribute set S^* with $f(S^*) = 1$.

5.3.2 Attribute Privacy of our ABID.

For the case of an honest verifier, that is, if an adversary $\mathcal{A}$ chooses a challenge CHA at random from $\mathrm{CHASP}(\lambda)$ in the attribute privacy game in Section 2.5.2 , attribute privacy follows from the honest verifier zero-knowledge property. However, in general, attribute privacy is not obvious. It seems an open problem to the best of authors' knowledge.

5.3.3 Security Reduction.

We mean a number theoretic problem here as the discrete-logarithm problem or the RSA-inverse problem ([5]).

There exists the following security reduction to a number theoretic problem.

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

Here we denote q_H as the maximum number of hash queries issued by the adversary on $\mathrm{FS}_{\boldsymbol{\Sigma}}$ in the random oracle model.

6. FURTHER APPLICATION TO ATTAIN ATTRIBUTE-BASED SIGNATURE

In this section, by applying the Fiat-Shamir transform [1] to our ABID, we obtain an attribute-based signature scheme ABS. Our ABS is secure in the random oracle model, possesses attribute privacy, and has a feature that it can be constructed without pairings.

6.1 Our ABS Scheme

The Fiat-Shamir transform ([9, 1]), briefly described in Section 2.3.2, can be directly applied to our ABID because our ABID is a Σ-protocol. Fig. 4 shows our construction of ABS scheme, ABS =(Setup, KG, Sign, Vrfy).

6.2 Security of Our ABS

We can apply the discussions in [1]. Then the security of the obtained attribute-based signature scheme is equivalent to the security of our ABID against passive attacks.

THEOREM 4. *Our attribute-based signature scheme ABS obtained by applying the Fiat-Shamir transform to our ABID is secure in the existential unforgeability game against chosen-message attacks, in the random oracle model, based on the passive security of ABID. More precisely, let q_H denote the*

Setup$(\lambda, \mathcal{U})$:
$(x_{\mathrm{M}}, w_{\mathrm{M}}) \leftarrow \mathrm{Instance}_R(\lambda)$
$\mathrm{PK} := (x_{\mathrm{M}}, \mathcal{U}), \mathrm{MSK} := w_{\mathrm{M}}$
Return$(\mathrm{PK}, \mathrm{MSK})$

KG$(\mathrm{PK}, \mathrm{MSK}, S)$:
$\kappa \leftarrow \mathit{PRFkeysp}(\lambda),\ \tau \leftarrow \{1,0\}^{\lambda}$
For att $\in S$
$m_{\mathrm{att}} := (\tau \parallel \mathrm{att})$
$\sigma_{\mathrm{att}} := (a_{\mathrm{att}}, w_{\mathrm{att}}) \leftarrow \mathrm{FS}^{\mathrm{sign}}_{\boldsymbol{\Sigma}}(x_{\mathrm{M}}, w_{\mathrm{M}}, m_{\mathrm{att}})$
$\mathrm{SK}_S := (\kappa, \tau, (\sigma_{\mathrm{att}})_{\mathrm{att}\in S})$
Return SK_S

Sign$(\mathrm{PK}, \mathrm{SK}_S, (m, f))$:
Initialize inner state
$\boldsymbol{\Sigma}_f^{\mathrm{eval}}(\mathcal{T}_f, S) \to (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_f)}$
If $v_{\mathrm{rt}(\mathcal{T}_f)} \neq 1$, then abort

else $\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_f)} := *$
Supp$(\mathrm{PK}, \mathrm{SK}_S, f) \to (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)}$
StmtGen$(\mathrm{PK}, \tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)})$
$\to \{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)} =: X$
$W := \{w_{\mathrm{att}}\}_{\mathrm{att}\in S}$

$\boldsymbol{\Sigma}_f^1(X, W, \mathcal{T}_f, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_f)}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_f)})$
$\to ((\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)},$
$(\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$

$\mathrm{CHA} \leftarrow H(m \parallel (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$
$\mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_f)} := \mathrm{CHA}$

$\boldsymbol{\Sigma}_f^3(X, W, \mathcal{T}_f, (v_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{Nd}(\mathcal{T}_f)}, \mathrm{CHA}_{\mathrm{rt}(\mathcal{T}_f)},$
$(\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)},$
$(\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$
$\to ((\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$

Return $\sigma := (\tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)},$
$(\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)},$
$(\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)}))$

Vrfy$(\mathrm{PK}, (m, f), \sigma := (\tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)},$
$(\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)},$
$(\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)}))$:
Initialize inner state
$\mathrm{CHA} \leftarrow H(m \parallel (\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$
StmtGen$(\mathrm{PK}, \tau, (a_{\mathrm{att}})_{\mathrm{att}\in\mathrm{Att}(f)})$
$\to \{x_{\mathrm{att}}\}_{\mathrm{att}\in\mathrm{Att}(f)} =: X$

$\boldsymbol{\Sigma}_f^{\mathrm{vrfy}}(X, \mathcal{T}_f,$
$(\mathrm{CMT}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)},$
$\mathrm{CHA}, (\mathrm{CHA}_{\mathrm{nd}})_{\mathrm{nd}\in\mathrm{tNd}(\mathcal{T}_f)},$
$(\mathrm{RES}_{\mathrm{lf}})_{\mathrm{lf}\in\mathrm{Lf}(\mathcal{T}_f)})$
$\to b$, Return b

Figure 4: Our ABS scheme.

maximum number of hash queries issued by the adversary 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).*

$$\mathbf{Adv}^{eufcma}_{\mathcal{F},ABS}(\lambda) \leq q_H \mathbf{Adv}^{pa}_{\mathcal{B},ABID}(\lambda) + neg(\lambda). \quad (3)$$

6.3 Discussion

6.3.1 Attribute Privacy of our ABS.

As opposed to the case of our ABID, our ABS has attribute privacy defined in Section 2.5.2. Actually, more strongly, attribute privacy holds for $\mathcal{A}$ with unbounded computational ability because, for a fixed access policy f, the distribution of messages and signatures (m, σ) does not depend on secret keys SK_S where $f(S) = 1$.

6.3.2 Security Reduction.

Let q_H denote the maximum number of hash queries issued by the adversaries on ABS and $\mathrm{FS}_{\mathbf{\Sigma}}$. Combining the inequalities (3), (1) and (2), we obtain the following security reduction on advantages.

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

7. EFFICIENCY COMPARISON

The most efficient, previously known ABS scheme is the one by Okamoto and Takashima (OT11) [14]. We compare efficiency of our ABS scheme with their scheme in the length of a signature as well as underlying assumption, in the discrete-logarithm setting.

A prime of bit length λ (the security parameter) is denoted by p and we fix a cyclic group $\mathbb{G}_p$ of order p. We assume that an element of $\mathbb{G}_p$ is represented by 2λ bits. Let $l := |\mathrm{Lf}(\mathcal{T}_f)|$ denote the number of leaf nodes in an access tree. DLIN and CR hash mean the Decisional Linear assumption for pairing group [14] and the collision resistance of an employed hash function, respectively.

Then the lengths of a signature of the scheme OT11 and our ABS are as in the Table 1. OT11 scheme has advantages in the security-proof model and access formula, whereas our ABS realizes shorter length of signature.

Table 1: Efficiency Comparison.

Scheme	OT11[14]	Our ABS
Len. of Sig. (bit)	$(2\lambda)(9l + 11)$	$(2\lambda)l + (\lambda)(4l - 1)$
Sec.-Proof Model	Standard	Random Oracle
Assumption	DLIN $\wedge$ CR hash	DLog
Access Formula	non-monotone	monotone
Adaptive Target	Yes	Yes
Attribute Privacy	Yes	Yes

8. CONCLUSIONS

Our strategy is in the Fiat-Shamir paradigm; we first constructed a boolean proof system $\mathbf{\Sigma}_f$ from a given Σ-protocol $\mathbf{\Sigma}$. Then, combining it with a credential bundle scheme from $\mathrm{FS}_{\mathbf{\Sigma}}$, we constructed ABID of proof of knowledge. Finally, by applying the Fiat-Shamir transform to our ABID, we obtained our ABS. Our ABS can be constructed without pairings.

9. ACKNOWLEDGEMENTS

We appreciate sincere comments of anonymous reviewers, Dr. Emura and A.Prof. Hasegawa at SCIS2014, and Prof. Takagi and A.Prof. Morozov at Kyushu University. Concerning the third author, this work is partially supported by Grants-in-Aid for Scientific Research; Research Project Number:25540004.

10. 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.

[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.

[3] A. Beimel. *Secure Schemes for Secret Sharing and Key Distribution*. PhD thesis, Israel Institute of Technology, Technion, Haifa, Israel, 1996.

[4] M. Bellare and O. Goldreich. On Defining Proofs of Knowledge. In *CRYPTO '92*, volume 740 of *LNCS*, pages 390–420. Springer.

[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.

[6] 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.

[7] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty Computation from Threshold Homomorphic Encryption. In *EUROCRYPT 2001*, volume 2045 of *LNCS*, pages 280–300. Springer.

[8] I. Damgård. On σ-protocols. In Course Notes, http://www.daimi.au.dk/ ivan/Sigma.ps, 2004.

[9] 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.

[10] 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.

[11] S. Guo and Y. Zeng. Attribute-Based Signature Scheme. In *ISA '08*, pages 509–511. IEEE.

[12] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2008.

[13] H. K. Maji, M. Prabhakaran, and M. Rosulek. Attribute-Based Signatures. In *CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer.

[14] 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.

[15] C. P. Schnorr. Efficient Identification and Signatures for Smart cards. In *CRYPTO '89*, volume 435 of *LNCS*, pages 239–252. Springer.