



Research Article

F2P-ABS: A Fast and Secure Attribute-Based Signature for Mobile Platforms

Guofeng Lin ^{1,2}, Yunhao Xia,² Chun Ying,³ and Zhixin Sun ^{1,2}

¹Technology Research and Development Center of Postal Industry, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

²“Broadband Wireless Communication and Sensor Network Technology” Key Lab of Ministry of Education, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

³National Engineering Laboratory for Logistics Information Technology, Shanghai YTO Express (Logistics) Co., Ltd., Shanghai 201705, China

Correspondence should be addressed to Zhixin Sun; sunzx@njupt.edu.cn

Received 10 July 2019; Accepted 14 November 2019; Published 19 December 2019

Academic Editor: Angel M. Del Rey

Copyright © 2019 Guofeng Lin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Attribute-based signature (ABS) is a promising cryptographic primitive. It allows the signer to generate a signature with attributes satisfying the predicate without leaking more information, so as to provide message authenticity in an anonymous manner. However, drawbacks concerning security and efficiency hinder its applications for authentication in mobile platforms. Here, we present F2P-ABS, an escrow-free and pairing-free attribute-based signature supporting perfect signer privacy for mobile anonymous authentication. To enhance its adaptiveness to mobile platforms, a novel key extraction is proposed so that the key escrow problem is mitigated over the single authority setting. It also helps to remarkably reduce the size of the signing key. Different from existing schemes, F2P-ABS is free from pairing operations. It performs no pairing operation for verification. Without the loss of security, we prove that F2P-ABS achieves signer privacy in perfect sense. It is also proven to guarantee existential unforgeability under corrupted and adaptive chosen predicate and message attack, which is securer than existing schemes.

1. Introduction

Nowadays, with rapid development of computer network, electrical messaging services have been widely used in industry, e-commerce, medical treatment, and so on. In order to reduce management cost, cloud-based messaging is gradually getting prevailing [1]. As a result, more and more messages are transferred by off-premise infrastructure that some organizations jointly held. Such an open environment makes it highly possible to be under persistent threat of tamper attack [2, 3].

Attribute-based signature (ABS) [4] is a variation of digital signature, which is one of the most promising cryptographic primitives. ABS stems from the identity-based signature [5, 6] but describes the signer by a set of attributes

instead of a single unique identity. To endorse a message, the signer is allowed to generate a signature with his/her attributes satisfying the predicate without leaking other information. Providing data authenticity in this anonymous manner [7], ABS has been brought into much focus. It is widely accepted to build cloud-based messaging services [8–10], which make management of dynamic environment easier.

In terms of security, ABS inherently suffers from key escrow problem [11–14]. Having the absolute power to generate and issue signing keys, the attribute authority is required to be fully trusted. If got compromised, the attribute authority can forge signatures associated with any tampered messages. As a result, the whole system would be broken. Furthermore, electrical messaging services are now

being mobilized with the popularity of wireless networks and smart devices. But, the resource limitation of power and computation is still a bottleneck. Existing ABS schemes have to consume much resource. For one thing, large numbers of pairing computation are involved for verification. For example, performing one pairing operation based on 512-bit Tate pairing costs about 2 times of time than performing one exponentiation on a 1024-bit module [15]. For another thing, the size of the signing key is challenging the storage capacity of signing devices such as edge nodes in the Internet of Things. If existing ABS schemes are deployed in such a mobile platform, the cost of time and other resources would be unacceptable when a large universe of attributes was involved in.

1.1. Related Work. Sahai and Waters [16] first introduced a fuzzy identity-based encryption as an extension of identity-based cryptography [5]. The user is identified by a set of attributes and capable of decrypting a ciphertext if the distance between this attribute set and the one embedded into the ciphertext is close enough. Their construction indicated a novel cryptographic primitive called attribute-based encryption. Goyal et al. [17] proposed a key-policy attribute-based encryption (KP-ABE) using a tree-based access policy. For KP-ABE, the ciphertext is associated with an attribute set while the private key is associated with an access policy. Bethencourt et al. [18] proposed a ciphertext-policy attribute-based encryption (CP-ABE), in which the ciphertext is associated with an access policy while the private key is associated with an attribute set. To allow the data owner to flexibly define access policy before encryption, their construction is more complicated than [17]. Chow [11] indicated a major drawback comes when deploying public key cryptosystem is the key escrow problem that the single key generation center could decrypt any message addressed to a user by generating that user's private key. To address this problem in ABE, Chase and Chow [12] proposed a multi-authority scheme in which each authority secretly holds a unique secret. For key generation, each authority must communicate with the rest of authorities. Even if $n - 1$ authorities corrupted, where n is the number of authorities, it poses no threat to confidentiality of ciphertext at all. However, it results in $O(n^2)$ communication overhead for each time of key generation. Zhang et al. [19] provided an improvement of [12] to reduce the size of keys, and more importantly, it deployed only one subauthority to generate keys by its interactions with the key authority. Hur [13] introduced a two-party computational protocol (2PC) between the key authority and the data-storing center for key issuing. His construction was a novel solution but failed to show more details about how 2PC works. Lin et al. [9] presented a collaborative key management protocol that prevents ABE-based data sharing system from key escrow and key exposure, which relies on much bilinear computation.

ABS is a new attribute-based cryptographic primitive that stems from ABE. In ABS, a signer who possesses a set of attributes from the attribute authority can sign a

message with a predicate that is satisfied by his attributes. It can be noted that predicate in ABS generally indicates the access policy or access structure. Waters [20] transformed IBE to an identity-based signature (IBS) through a method proposed in [5]. After that, Yang et al. [21] first proposed a fuzzy identity-based signature (FIBE), in which the signer can endorse a message on behalf of an organization if his or her identity is similar enough to the claimed predicate. Yang et al.'s scheme is of some basic functions of ABS but does not guarantee signer privacy. Maji et al. [4] first formulated ABS with formal security definitions of existential unforgeability and signer privacy. Moreover, they built the predicate by monotone span program to provide flexible access control and the multi-authority construction to avoid leaking secrets when the attribute authority gets corrupted. However, they just proved security in the generic group model. It can be noted that those schemes stated above are generally considered as laying a theoretical foundation of ABS. Li et al. [22] proposed an ABS scheme with one central attribute authority and a group of distributed attribute authority, which, respectively, generate different components of a signing key. Although they provided provable security in the standard model, the construction is lack of expressiveness due to threshold predicate. To improve efficiency, a novel threshold ABS with a constant size signature was proposed [23]. For the (t, t) -ABS scheme, each signature contains just two elements from group \mathbb{G} . And, for the (t, n) -ABS scheme, the signature size is linear in the number of signing attributes. Su et al. [24] proposed an efficient ABS scheme for Internet of Things that provides flexible access control. It supports any predicate consists of AND and OR threshold gates. Li et al. [25] indicated that, to guarantee integrity of outsourced data, the server needs to respond to mass authentication requests when receiving data from large numbers of users. It sets a huge challenge for performance of the server due to large numbers of bilinear pairing operations in authentication. They proposed a novel ABS that supports verifying different signatures in a batch manner so as to improve verification efficiency. Cui et al. [14] separated the attribute authority into two independent parties, namely, the attribute certifying authority and the key generation center. The attribute certifying authority is responsible for authenticating users' attributes while the key generation center is known as taking over key generation. They designed an interactive protocol between the key generation center and the user based on zero-knowledge proof, by which only the user himself or herself knows the signing key after the interaction. As a result, the key generation center can by no means forge a valid signature even it gets corrupted. Rao and Dutta [26] proposed a novel key-policy ABS in which the predicate is built by linear secret sharing scheme (LSSS) to provide rich expressiveness. In addition, only 3 times of bilinear pairing operations are executed in verification. Rao [27] introduced a key-insulated mechanism for ABS to reduce the possibility of occurrence of key exposure when deploying ABS in performance-restrained platforms such as mobile devices. Due to periodical key

updating, even if the unauthorized user obtains the signing key, the valid signature cannot still be forged unless within the current time slot.

1.2. Our Contributions. In this paper, we focus on constructing a secure and efficient ABS scheme and try to mitigate problems that existing schemes [24–27] suffer. We try to realize the vital properties which include escrow-free, pairing-free, and perfect signer privacy. As is shown in Table 1, the ABS schemes [24, 26, 27] are not able to guarantee security if the attribute authority got compromised when it holds signing keys in escrow. The scheme in [25] mitigates the key escrow problem with dependency on multiauthority setting, which makes it cumbersome in computation and communications (see Tables 2 and 3).

The key extraction of our F2P-ABS is elegant and effective. It mitigates the key escrow problem on the condition of the single authority setting. With the participation of both attribute authority and data-storing center, anyone of them getting compromised will have no impact on security. By the key extraction of F2P-ABS, the size of the signing key is reduced by nearly half of that in [24–27] (see Table 2).

Since pairing computation is considered the most expensive operation, the number of pairing operations is a critical factor of feasibility in practical scenes. Most ABS schemes are dependent on pairing, so the signer has to execute pairing operations to validate the message. The required number of pairing operations in [24] proliferates linearly with the number of attributes in a minimum attribute set of a signing predicate. The required number of pairing operations in [25, 27] proliferates linearly with the number of attributes in a signing predicate. It demonstrates that a feasible ABS scheme which provides both strong reliability and high-efficiency scheme is far-fetched, especially for authentication in mobile platform. For our F2P-ABS, in contrast, it does not need any pairing operations for verifying (see Table 3). This is even less than that in [26], which is 3 pairing operations.

To enhance security, we provide a novel security model called existential unforgeability under corrupted and adaptive chosen predicate and message attack (EUF-ca-CPMA). And, our F2P-ABS is proven to be EUF-ca-CPMA secure assuming the hardness of discrete logarithm (DL) problem. It is stronger than existential unforgeability under a selective predicate/attribute set and adaptive chosen-message attack (EUF-sP-CMA/EUF-sA-CMA), which is met by [24–27]. Furthermore, it guarantees signer privacy in a perfect sense as in [26].

To conclude, the proposed F2P-ABS outperforms the existing schemes in terms of security and efficiency.

1.3. Organization. The rest of the paper is organized as follows. We review essential preliminaries in Section 2, show the architecture in Section 3, show the security requirements in Section 4, propose main construction of F2P-ABS in Section 5, provide security analysis in Section 6, present

performance comparison against similar schemes in Section 7, and draw conclusion and highlight our future work in Section 8.

2. Preliminaries

In this section, we will introduce some essential cryptographic preliminaries. Before it, we first describe some important notations, which is shown in Table 4. These are required for an unambiguous presentation of the paper, some of which will be interpreted in the following sections.

2.1. Predicate

Definition 1 (access structure). Let $\{P_1, P_2, \dots, P_n\}$ be a party of several participants. An access structure \mathcal{A} is defined as an arbitrary collection of nonempty subsets of $\{P_1, P_2, \dots, P_n\}$, namely, $\mathcal{A} \subseteq 2^{\{P_1, P_2, \dots, P_n\}} \setminus \{\emptyset\}$. We consider a set S authorized if $S \in \mathcal{A}$.

For two arbitrary sets B and C such that $B \subseteq C$, we say \mathcal{A} is monotonic if $C \in \mathcal{A}$. It can be noted that the term access structure mentioned in the following means monotonic access structure if there is no particular revelatory. For ABS, we note that a predicate generally indicates an access structure. From now on, to avoid confusion, we only use the term predicate.

Definition 2 (tree-based predicate). Let \mathcal{T} be a tree-based predicate, which is associated with a tree consisting of some leaf nodes and nonleaf nodes. Each nonleaf node represents a threshold gate, described by the number of its children and a threshold value. Each tree also defines an ordering among all nodes, and accordingly each index value is uniquely assigned to a node in an arbitrary manner. Without loss of generality, let x be an index of node. If num_x is the number of children of a node x and k_x is its threshold value, then $0 < k_x \leq \text{num}_x$. When $k_x = 1$, the threshold gate is an “OR” gate, and when $k_x = \text{num}_x$, it is an “AND” gate. Each leaf node x represents an attribute. For arbitrary attribute set S , we have $\mathcal{T}(S) = 1$ if it satisfies the predicate that we described; otherwise, we have $\mathcal{T}(S) = 0$.

Definition 3 (minimum attribute set). For all attribute set S such that $\mathcal{T}(S) = 1$, there exists a minimum attribute set $S_{\min} \subseteq S$ such that $\mathcal{T}(S_{\min}) = 1$ and $\mathcal{T}(S') = 0$ for all $S' \not\subseteq S_{\min}$. For instance, there is a tree-based predicate \mathcal{T} whose postfix expression is {“D,” “E,” “F,” “2 of 3,” “A,” “B,” “C,” “1 of 3,” “2 of 2”}. Each character represents a leaf node associated with an attribute. Each string like “ x of y ” represents a nonleaf node where x is its threshold and y is the number of its child nodes. A signer, whose attribute set S is {“A,” “B,” “D,” “E,” “F”}, sends a message signed through ABS with this predicate. Obviously, one of his minimum attribute sets S_{\min} is {“A,” “D,” “E”}.

2.2. Garbled Circuit

Definition 4 (garbled circuit). Garbled circuit allows two parties holding inputs x and y , respectively, to evaluate an arbitrary function $f(x, y)$ without leaking any

TABLE 1: Comparison in terms of functionality of ABS schemes.

Scheme	[24]	[25]	[26]	[27]	F2P-ABS
Authority	Single	Multiple	Single	Single	Single
Escrow-free	×	√	×	×	√
Pairing-free	×	×	×	×	√
Predicate	Tree	Threshold	LSSS	LSSS	Tree
Perfect signer privacy	×	×	√	×	√
Complexity assumption	CDH	CDH	n-CDHE	n-CDHE	DL
EUF-ca-CPMA	×	×	×	×	√

TABLE 2: Comparison in terms of public parameter size, secret key size, signing key size, and signature size.

Scheme	Param.	Sec.	Sk.	Sig.
[24]	$3 \mathbb{G} + \mathbb{G}_T $	$ Z_p^* $	$(2a_{sk} + 1) \mathbb{G} $	$(2a_u + 2) \mathbb{G} $
[25]	$k(k-1) Z_p^* + k \mathbb{G} + k \mathbb{G}_T $	$k Z_p^* $	$(2a_{sk} + 1) \mathbb{G} $	$(a_{sp} + 2) \mathbb{G} $
[26]	$(a_u + l + 2) \mathbb{G} + \mathbb{G}_T $	$ Z_p^* $	$(a_u + a_{sk}) \mathbb{G} $	$3 \mathbb{G} $
[27]	$(a_u + 1) \mathbb{G} + \mathbb{G}_T $	$ \mathbb{G} $	$(2a_{sk} + 3) \mathbb{G} $	$(a_{sp} + a_{ep} + 4) \mathbb{G} $
F2P-ABS	$(2a_u + 4) \mathbb{G} $	$(2a_u + 4) Z_p^* $	$(a_{sk} + 1) \mathbb{G} $	$(a_{sp} + 2) Z_p^* + a_{sp} \mathbb{G} $

TABLE 3: Comparison in terms of computation cost during signing and verifying.

Scheme	Signing			Verifying		
	Exp. over \mathbb{G}	Exp. over \mathbb{G}_T	Pair.	Exp. over \mathbb{G}	Exp. over \mathbb{G}_T	Pair.
[24]	$4a_{sp} + 2$	0	0	$n_{sp} + \phi_{sp} - 1$	0	$2\phi_{sp} + 2$
[25]	$(k+1)\phi_{sp} + ka_{sp} + 2$	0	0	0	0	$a_{sp} + 2$
[26]	$\phi_{sp} + a_{sp} + l + 1$	0	0	$\phi_{sp} + l$	0	3
[27]	$4a_{sp} + 2a_{ep} + 8$	1	0	$a_{sk} + 2\phi_{ep} + 1$	0	$a_{sp} + 5$
F2P-ABS	a_{sp}	0	0	$a_{sp} + n_{sp} + \phi_{sp} - 1$	0	0

TABLE 4: Notations.

Notation	Definition
att_i	Attribute
Ω	Attribute universe
S	Signer attribute set
\mathcal{T}	Tree-based predicate
S_{\min}	Minimum attribute set of S according to \mathcal{T}
Node x	A node of \mathcal{T} whose index is x
$\text{att}(x)$	Attribute associated with the leaf node x
\mathcal{T}_x	Tree-based subpredicate of \mathcal{T} rooted at node x
$\text{parent}(x)$	Parent node of node x
$\Lambda(x)$	Set of child nodes of node x
$\Lambda'(x)$	Set of child nodes at which subpredicates are rooted that S_{\min} cannot satisfy
$z \xleftarrow{R} X$	z is randomly chosen from X according to uniform distribution
GC_f	Garbled circuit of function f

information about their inputs beyond what is implied by the function output. The basic idea is that one party (the garbler) prepares an encrypted version of the function f ; the data transmitter and receiver then obliviously compute the output of the circuit without learning any intermediate values.

Starting with a Boolean circuit $f(b_i, b_j)$ where b_i and b_j are one-bit inputs, respectively, from input wires i and j . The circuit generator associates two random cryptographic keys w_i^0 and w_i^1 (w_j^0 and w_j^1) with each input wire i (j). Then, for each output $b_k = f(b_i, b_j)$, the generator computes

ciphertext $\text{Enc}_{w_i^{b_i}, w_j^{b_j}}(w_k^{b_k})$. The resulting four ciphertexts, in a random order, constitute a garbled table. Then, the garbled table is sent to the evaluator.

The evaluator also obtains the keys corresponding to each input. The circuit generator can first simply send $w_i^{b_i}$, which corresponds to the actual input from the circuit generator. Then, it uses 1-out-of-2 oblivious transfer (see Definition 5) to enable the circuit evaluator to obliviously obtain the key $w_j^{b_j}$ corresponding to its own input.

Given keys associated with both inputs from wires i and j , the evaluator can extract an output key $w_k^{b_k}$ by decrypting the appropriate ciphertext. Only one single decryption suffices the extraction [28]. Mapping from output keys to output bits, the evaluator can finally learn the actual output of f . If desired, the evaluator can share the output with the circuit generator. By using the framework proposed by Huang et al. [29], it is efficient to modularly build a secure protocol computing the function f .

Definition 5 (1-out-of- m oblivious transfer). A 1-out-of- m oblivious transfer protocol OT_1^m [30] refers to a protocol where at the beginning of the protocol, one party, Bob, has m inputs X_1, \dots, X_m and at the end of the protocol, the other party, Alice, learns one of the inputs X_i for an i ($1 \leq i \leq m$) of own choice, without learning anything about the other inputs and without allowing Bob to learn anything about i . An efficient OT_1^m was proposed by Tzeng [31].

2.3. Complexity Assumption

Definition 6 (discrete logarithm assumption). Given a multiplicative cyclic group \mathbb{G} with a prime order p . The discrete logarithm (DL) problem is given $g_1 \leftarrow_R \mathbb{G}$ and a generator g of \mathbb{G} , to find a unique $x \in Z_p^*$ which makes $g_1 = g^x$. For clarity, we denote DLP by $x \leftarrow (\mathbb{G}, g, g_1)$. If there is a polynomial-time algorithm \mathcal{A} extracting x with a probability that satisfies $\Pr[x \leftarrow (\mathbb{G}, g, g_1)] \geq \epsilon$, we denote the advantage to solve DLP by $\text{Adv}_{\mathcal{A}}^{\text{DLP}} = \epsilon$. The DL assumption says that there is no such an algorithm which can solve DL problem with a nonnegligible advantage.

It is well known that this problem is considered to be intractable. The DL assumption has been used to create many cryptosystems, including the ElGamal cryptosystem. This intractable assumption has also been used to create signature schemes [32].

3. F2P-ABS Architecture Overview

3.1. Definition of F2P-ABS. Following the definition in [26], our F2P-ABS consists of six algorithms:

- (1) $gparam \leftarrow \text{GSetup}(1^\lambda)$: the GSetup algorithm is run with a security parameter λ , which outputs a global parameter $gparam$.
- (2) $\{dparam, dsk\} \leftarrow \text{DSetup}(gparam)$: the DSetup algorithm takes $gparam$ as input and generates the data-storing center parameter $dparam$ and the data-storing center secret key dsk . Finally, it publishes $dparam$ and stores dsk .
- (3) $\{aparam, ask\} \leftarrow \text{ASetup}(gparam)$: the ASetup algorithm takes $gparam$ as input and generates the attribute authority parameter $aparam$ and the authority secret key ask . Finally, it publishes $aparam$ and stores ask .
- (4) $sk \leftarrow \text{KeyExtract}(gparam, dparam, aparam, ask, S)$: the KeyExtract algorithm takes $gparam$, $dparam$, $aparam$, ask , and dsk as input, as well as an attribute set S . Finally, it outputs a signing key sk .
- (5) $\sigma \leftarrow \text{Sign}(gparam, dparam, aparam, sk, \mathcal{T}, m)$: the Sign algorithm takes $gparam$, $dparam$, $aparam$, and sk as input to endorse a message m with respect to the predicate \mathcal{T} . Then, a valid signature σ is generated.
- (6) $1/0 \leftarrow \text{Verify}(gparam, dparam, aparam, m, \mathcal{T}, \sigma)$: the Verify algorithm takes $gparam$, $dparam$, $aparam$, m , \mathcal{T} , and σ as input. It outputs 1 if σ is a valid signature of m corresponding to \mathcal{T} ; otherwise, it outputs 0.

3.2. System Description. As is shown in Figure 1, the framework of an anonymous message authentication system based on F2P-ABS consists of the following four entities:

- (1) *Attribute Authority (AA)*. This is a semitrusted entity that generates $aparam$ and ask . By collaborating

with the data-storing center (DSC), it issues sk to the signer. All communications between the AA, the signer, and the DSC are secured by secure shell (SSH) protocol.

- (2) *Data-Storing Center (DSC)*. This is a semitrusted entity that stores messages and their corresponding signature. It takes part in issuing sk for the signer to avoid put sk in escrow in the AA. But, it also does not obtain any part of sk . Besides, it is responsible for transferring messages and signatures to corresponding verifiers by SSH protocol.
- (3) *Signer*. This is who wants to share authentic messages in public environments, such as in social network. The signer defines a predicate that exhibits the authenticity of the signer to the public. Then, the message m will be signed by sk , and a signature σ is generated. As is shown in Figure 1, the predicate can be described by a postfix expression {"Professor," "Associate Professor," "Lecturer," "1 of 3," "Department of Informatics," "University A," "3 of 3"}. Alice whose attribute set is {"University A," "Department of Informatics," "Professor"} can sign the message with this predicate. Finally, m and σ will be outsourced to the DSC.
- (4) *Verifier*. This is who receives m and wants to verify it. By computation with m and σ , the verifier will know whether the message is valid. As shown in Figure 1, Bob and Carol can verify that message comes from the signer satisfies that predicate but do not know who he/she is.

4. Security Requirements

For F2P-ABS, we adopt a semitrusted (also known as honest-but-curious) threat model, where attribute authorities and data-storing center are assumed to follow the protocol but may attempt to learn additional information from the protocol transcript. Although this is a bit weak, it is a standard security model for escrow-free computation. Studying F2P-ABS in the semitrusted setting is relevant for two reasons:

- (1) There may be instances where a semitrusted model is appropriate: (1) when parties are legitimately trusted but are prevented from divulging information for legal reasons, or want to protect against future break-in; (2) where it would difficult for parties to change the software attestation is used or due to internal controls in place (for example, when parties represent corporations or government agencies).
- (2) Protocols for the semitrusted setting are an important first step toward constructing protocols with stronger security guarantees. There exist generic ways to give full security against malicious authorities.

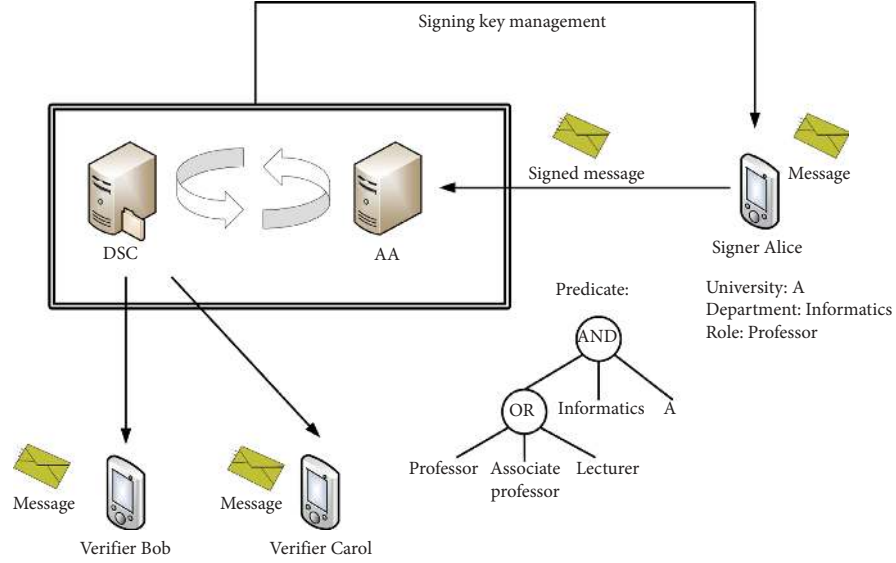


FIGURE 1: The mobile anonymous authentication system based on F2P-ABS.

4.1. Correctness

Definition 7 (correctness of F2P-ABS). We define a valid signature to be the one that can be verified with success probability 1. We say F2P-ABS scheme guarantees correctness if, for all pairs of attribute set S and predicate \mathcal{T} satisfying $\mathcal{T}(S) = 1$, all signing keys associated with S can always be used to generate a valid signature of message m with predicate \mathcal{T} .

4.2. Perfect Signer Privacy

Definition 8 (perfect signer privacy of F2P-ABS). Our F2P-ABS is perfectly private if, for all $gparam \leftarrow GSetup(1^\lambda)$, all $\{dparam, dsk\} \leftarrow DSetup(gparam)$, all $\{aparam, ask\} \leftarrow ASetup(gparam)$, all messages m , all attribute sets S_1 and S_2 , all $sk_1 \leftarrow KeyExtract(gparam, dparam, aparam, ask, dsk, S_1)$ and $sk_2 \leftarrow KeyExtract(gparam, dparam, aparam, ask, dsk, S_2)$, all predicates \mathcal{T} such that $\mathcal{T}(S_1) = 1$ and $\mathcal{T}(S_2) = 1$, and the distributions of $\sigma_1 \leftarrow Sign(gparam, dparam, aparam, sk_1, \mathcal{T}, m)$ and $\sigma_2 \leftarrow Sign(gparam, dparam, aparam, sk_2, \mathcal{T}, m)$ are equal.

4.3. Existential Unforgeability. Based on the definition of existential unforgeability provided in [33], we build the definition of existential unforgeability under corrupted and adaptive chosen predicate and message attack (EUF-ca-CPMA).

Definition 9 (EUF-ca-CPMA of F2P-ABS). We define ϵ to be the success probability of an adversary \mathcal{A} in the following game with a challenger \mathcal{C} :

- (1) *Initial Phase.* \mathcal{A} submits a bit $c_0 \in \{0, 1\}$, which will be used to forge a signature over either the corrupted AA or the corrupted DSC.
- (2) *Setup Phase.* \mathcal{C} publishes all parameters independent with attributes to \mathcal{A} with the following steps:

- (1) \mathcal{C} runs $gparam \leftarrow GSetup(1^\lambda)$ and gives $gparam$ to \mathcal{A} .
 - (2) \mathcal{C} runs $\{dparam, dsk\} \leftarrow DSetup(gparam)$ and gives \mathcal{A} part of $dparam$ that is independent of attributes.
 - (3) \mathcal{C} runs $\{aparam, ask\} \leftarrow ASetup(gparam)$ and gives \mathcal{A} part of $aparam$ that is independent to attributes
 - (4) If $c_0 = 0$, \mathcal{C} gives \mathcal{A} part of ask that is independent to attributes. Otherwise, \mathcal{C} gives \mathcal{A} part of dsk that is independent to attributes.
- (3) *Query Phase.* \mathcal{A} adaptively issues a polynomially bounded number of queries to the following oracles:
- (1) *Attribute Oracle.* \mathcal{A} submits an attribute att_i . If $c_0 = 0$, \mathcal{C} returns to \mathcal{A} part of ask and $dparam$ that correspond to attributes. Otherwise, \mathcal{C} returns to \mathcal{A} part of $aparam$ and dsk that correspond to attributes.
 - (2) *Key Extracting Oracle.* \mathcal{A} submits an attribute set S and obtains the corresponding signing key $sk \leftarrow KeyExtract(gparam, dparam, aparam, ask, dsk, S)$.
 - (3) *Signing Oracle.* \mathcal{A} submits a message m and a predicate \mathcal{T} . \mathcal{C} selects an attribute set S such that $\mathcal{T}(S) = 1$ computes a signing key $sk \leftarrow KeyExtract(gparam, dparam, aparam, ask, dsk, S)$ and returns the corresponding signature $\sigma \leftarrow Sign(gparam, dparam, aparam, sk, \mathcal{T}, m)$.
 - (4) *Forge Phase.* \mathcal{A} forges a signature σ^* of a message m^* with a predicate \mathcal{T}^* and succeeds in this game if m^* and \mathcal{T}^* are never queried to the signing oracle; \mathcal{T}^* does not accept any attribute set queried to key extracting oracle; \mathcal{C} eventually obtains $1 \leftarrow Verify(gparam, dparam, aparam, m^*, \mathcal{T}^*, \sigma^*)$.

We say F2P-ABS is $(t, q_{\text{att}}, q_{\text{sk}}, q_{\text{sig}}, \varepsilon)$ -EUF-CPMA secure if, for any probabilistic polynomial-time adversary \mathcal{A} running in time at most t that makes at most q_{att} attribute queries, q_{sk} key extracting queries, and q_{sig} signing queries, the \mathcal{A} 's advantage $\text{Adv}_{\mathcal{A}}^{\text{EUF-CPMA}}$ is at most ε .

5. F2P-ABS Main Construction

In this section, the main construction of F2P-ABS and the details of how our F2P-ABS proceeds among the AA, the DSC, the signer, and the verifier are presented.

5.1. Global Setup. To initialize the system, the AA first calls the GSetup algorithm. It can be noted that the GSetup algorithm can be run by any other trusted party ahead of system initialization. It takes a security parameter λ as input and proceeds the following steps:

- (1) It selects a cyclic group \mathbb{G} of λ -bit prime order p with generator g
- (2) It generates an attribute universe $\Omega = \{\text{att}_1, \text{att}_2, \dots, \text{att}_n\}$, which is a large universe that contains attributes of all authorized users
- (3) It selects a hash function $H : \{0, 1\}^* \rightarrow Z_p^*$
- (4) Outputs the parameter $gparam = \langle p, \mathbb{G}, g, \Omega, H \rangle$

5.2. DSC Setup. It is also necessary for the DSC to participate in the system initialization. After the global setup, the DSC calls DSetup algorithm with $gparam$ and runs the following steps:

- (1) It selects $\alpha_1, \alpha_2 \xleftarrow{R} Z_p^*$ and computes $A_1 = g^{\alpha_1}$ and $A_2 = g^{\alpha_2}$
- (2) For each attribute $\text{att}_i \in \Omega$, it selects $\varepsilon_{1,i}, \varepsilon_{2,i} \xleftarrow{R} Z_p^*$ and computes $T_{1,i} = g^{\varepsilon_{1,i}}$ and $T_{2,i} = g^{\varepsilon_{2,i}}$
- (3) It generates a garbled circuit GC_f of the function $f(x_1, x_2, y_1, y_2) = x_1 + y_1/x_2 + y_2$
- (4) It published the DSC parameter $dparam = \{A_1, A_2, C, \{\forall \text{att}_i \in \Omega : T_{1,i}, T_{2,i}\}\}$ and stores the DSC secret key $dsk = \{\alpha_1, \alpha_2, \{\forall \text{att}_i \in \Omega : \varepsilon_{1,i}, \varepsilon_{2,i}\}\}$

5.3. AA Setup. As the final procedure of system initialization, the AA calls the ASetup algorithm which takes $gparam$ as input. Then, it runs the following steps:

- (1) It selects $\beta_1, \beta_2 \xleftarrow{R} Z_p^*$ and computes $B_1 = g^{\beta_1}$ and $B_2 = g^{\beta_2}$
- (2) For each $\text{att}_i \in Z_p^*$, it selects $\gamma_{1,i}, \gamma_{2,i} \xleftarrow{R} Z_p^*$ and computes $U_{1,i} = g^{\gamma_{1,i}}$ and $U_{2,i} = g^{\gamma_{2,i}}$
- (3) It publishes the AA parameter $aparam = \{B_1, B_2, \{\forall \text{att}_i \in \Omega : U_{1,i}, U_{2,i}\}\}$ and stores the AA secret key $ask = \{\beta_1, \beta_2, \{\forall \text{att}_i \in \Omega : \gamma_{1,i}, \gamma_{2,i}\}\}$

5.4. Key Extraction. When the signer asks for authorization with an attribute set S , the AA and the DSC jointly calls the KeyExtract algorithm, which takes as input $gparam$, $dparam$, $aparam$, dsk , ask , and the attribute set S . The execution of the KeyExtract algorithm is as follows:

- (1) The AA selects $r \xleftarrow{R} Z_p^*$
- (2) $D_1(\alpha_1, \alpha_2, \beta_1 + r, \beta_2)$ is generated by using GC_f between the AA and the DSC
- (3) For each $\text{att}_i \in \Omega$, $D_{2,i} = f(\varepsilon_{1,i}, \varepsilon_{2,i}, \gamma_{1,i} - r, \gamma_{2,i})$ is generated by using GC_f between the AA and the DSC
- (4) The signing key $sk = \{S, D_1, \{\forall \text{att}_i \in S : D_{2,i}\}\}$ returns to the signer

5.5. Signing. To sign a message m with predicate \mathcal{T} , the signer calls the Sign algorithm. It takes as input the predicate \mathcal{T} , the message m , and the signing key sk . Then, it proceeds as follows:

- (1) It randomly selects a secret $s \xleftarrow{R} Z_p^*$ and then assigns a random polynomial for each node x of \mathcal{T} as follows:
 - (1) If node x is the root node and its threshold is k_{root} , set a random d_{root} -degree polynomial q_{root} such that $q_{\text{root}}(0) = s$, where $d_{\text{root}} = k_{\text{root}} - 1$.
 - (2) If node x is neither the root node nor a leaf node and its threshold is k_x , set a random d_x -degree polynomial q_x such that $q_x(0) = q_{\text{parent}(x)}(x)$, where $d_x = k_x - 1$.
 - (3) If node x is a leaf node, set a zero-degree polynomial $q_x = q_{\text{parent}(x)}(x)$.
- (2) It selects another secret $s' \xleftarrow{R} Z_p^*$ and assigns another polynomial for each node x of \mathcal{T} as follows:
 - (1) If node x is the root node, set a random 2-degree polynomial q'_{root} such that $q'_{\text{root}}(0) = s'$.
 - (2) If node x is neither the root node nor a leaf node, set a random 2-degree polynomial q'_x such that $q'_x(0) = q'_{\text{parent}(x)}(x)$.
 - (3) If node x is a leaf node, set a 0-degree polynomial $q'_x = q'_{\text{parent}(x)}(x)$.
- (3) The signer chooses a minimum attribute set S_{min} . For each leaf node x in \mathcal{T} such that $\text{att}(x) = \text{att}_i$, the following computation will be executed:
 - (1) If $\mathcal{T}_x(S_{\text{min}}) = 1$, compute $\sigma_{1,x} = (T_{1,i}U_{1,i})^{q_x(0)}$ and $\sigma_{2,x} = D_{2,i}q_x(0)$.
 - (2) If $\mathcal{T}_x(S_{\text{min}}) = 0$, compute $\sigma_{1,x} = (T_{1,i}U_{1,i})^{q'_x(0)}$ and $\sigma_{2,x} = D_{2,i}q'_x(0)$.
- (4) Finally, it computes $\sigma_3 = D_1(s + s')$ and $\sigma_4 = H((A_1B_1)^{(s+s')} \| m)$ and the signature is presented as $\sigma = \{\mathcal{T}, \{\sigma_{1,x}, \sigma_{2,x}\}, \sigma_3, \sigma_4\}$.

Let $\Lambda(x)$ denote the set of the all child nodes of node x . Let $\Lambda'(x)$ denote the set of some child nodes of node x such that S_{min} cannot satisfy the predicate rooted at them, namely,

$\Lambda'(x) = \{z \mid z \in \Lambda(x) \cap \mathcal{F}_z(S_{\min}) = 0\}$. For F2P-ABS, there is only one restriction on predicate \mathcal{F} which is $|\Lambda'(x)| \geq 2$.

5.6. *Verifying.* To validate this message, the verifier calls for the Verify algorithm. It takes as input the message m and the corresponding signature σ . Then, it proceeds as follows:

(1) For each node x of \mathcal{T} , it recursively calls the function $\text{VerifyNode}(x)$.

(1) If x is a leaf node, it computes as equation (1). For generality, we define that, for each node x such that $\mathcal{F}_x(S_{\min}) = 1$, it has a $q'_z(0) = 0$.

$$\begin{aligned} \text{VerifyNode}(x) &= \frac{(T_{2,i}U_{2,i})^{\sigma_{2,x}}}{\sigma_{1,x}} \\ &= \begin{cases} g^{-r(q_x(0)+q'_x(0))}, & \text{if } \mathcal{F}_x(S_{\min}) = 1, \\ g^{-rq'_x(0)}, & \text{if } \mathcal{F}_x(S_{\min}) = 0. \end{cases} \end{aligned} \quad (1)$$

(2) If x is a nonleaf node, we denote the computation result of its arbitrary child node z by F_z . Then, it computes as equation (2). It is assured that the result $g^{-r(s+s')}$ of the root node can be computed with finite recursion, if $\mathcal{F}(S) = 1$ holds:

$$\begin{aligned} \text{VerifyNode}(x) &= \prod_{z \in \Lambda(x)} F_z^{\Lambda_z(x)(0)} \\ &= \prod_{z \in \Lambda(x), \mathcal{F}_z(S_{\min})=1} g^{-r(q_z(0)+q'_z(0))} \\ &\quad \cdot \prod_{z \in \Lambda(x), \mathcal{F}_z(S_{\min})=0} g^{-rq'_z(0)} \\ &= \begin{cases} g^{-r(q_x(0)+q'_x(0))}, & \text{if } \mathcal{F}_x(S_{\min}) = 1, \\ g^{-rq'_x(0)}, & \text{if } \mathcal{F}_x(S_{\min}) = 0. \end{cases} \end{aligned} \quad (2)$$

(2) It computes $Z = H(((A_2B_2)^{\sigma_3} g^{(-r)(s+s')}) \| m)$ and output 1 if $Z = \sigma_4$ holds. That is, the message is indeed from a legal signer. Otherwise, it outputs 0 to indicate the signature is invalid.

6. Security Analysis

In this section, we will discuss how our F2P-ABS meets the security requirements mentioned in previous section.

6.1. Correctness

Theorem 1. *F2P-ABS guarantees correctness.*

Proof. Considering the Verify algorithm, the result $g^{-r(q_{\text{root}}(0)+q'_{\text{root}}(0))} = g^{-r(s+s')}$ of the root node can be computed with finite recursion if $\mathcal{F}(S) = 1$ holds. If the message is integrated, accordingly, equation (3) will always hold:

$$\begin{aligned} &H\left(\left((A_2B_2)^{\sigma_3} g^{-r(s+s')}\right) \| m\right) \\ &= H\left(\left(g^{\alpha_2} g^{\beta_2}\right)^{(\alpha_1+\beta_1+r)(\alpha_2+\beta_2)} g^{-r(s+s')} \| m\right) \\ &= H\left(\left(g^{(\alpha_1+\beta_1+r)} g^{-r(s+s')}\right) \| m\right) \\ &= H\left(g^{(\alpha_1+\beta_1)(s+s')} \| m\right) \\ &= \sigma_4. \end{aligned} \quad (3)$$

In this case, the Verify algorithm outputs 1 with success probability 1. \square

6.2. Perfect Signer Privacy

Theorem 2. *F2P-ABS guarantees perfect signer privacy.*

Proof. Considering the Sign algorithm, the following assertion always holds:

- (1) Since the secret numbers s and s' are randomly selected, both $\sigma_3 = D_1(s+s')$ and $\sigma_4 = H((A_1B_1)^{(s+s')} \| m)$ are uniformly distributed in Z_p^*
- (2) For an arbitrary leaf node x such that $\mathcal{F}_x(S_{\min}) = 1$, we have $\sigma_{1,x} = (T_{1,i}U_{1,i})^{q_x(0)}$ and $\sigma_{2,x} = D_{2,i}q_x(0)$, which are uniformly distributed in \mathbb{G} because $q_x(0)$ is from a random polynomial
- (3) For an arbitrary leaf node x such that $\mathcal{F}_x(S_{\min}) = 0$, we have $\sigma_{1,x} = (T_{1,i}U_{1,i})^{q'_x(0)}$ and $\sigma_{2,x} = D_{2,i}q'_x(0)$, which are also uniformly distributed in \mathbb{G} because $q'_x(0)$ is also from a random polynomial

Therefore, for all $gparam \leftarrow \text{GSetup}(1^\lambda)$, all $\{dparam, dsk\} \leftarrow \text{DSetup}(gparam)$, all $\{aparam, ask\} \leftarrow \text{ASetup}(gparam)$, all messages m , all attribute sets S_1 and S_2 , all signing keys $sk_1 \leftarrow \text{KeyExtract}(gparam, dparam, aparam, ask, dsk, S_1)$ and $sk_2 \leftarrow \text{KeyExtract}(gparam, dparam, aparam, ask, dsk, S_2)$, all predicates \mathcal{F} such that $\mathcal{F}(S_1) = 1$ and $\mathcal{F}(S_2) = 1$, and the distributions of $\sigma_1 \leftarrow \text{Sign}(gparam, dparam, aparam, sk_1, \mathcal{F}, m)$ and $\sigma_2 \leftarrow \text{Sign}(gparam, dparam, aparam, sk_2, \mathcal{F}, m)$ are equal. \square

6.3. Existential Unforgeability

Theorem 3. *F2P-ABS is EUF-ca-CPMA secure in the random oracle assuming the hardness of DL problem.*

Proof. Suppose there exists an adversary \mathcal{A} that can break F2P-ABS by chosen predicate and message attacks. We can build a simulator \mathcal{B} that can solve DL problem in a challenge game. The challenge game proceeds as follows:

- (1) *Initial Phase.* \mathcal{A} sends a challenge bit $c_0 \in \{0, 1\}$ to \mathcal{B} .
- (2) *Setup Phase.* The challenger \mathcal{C} generates $g_1 \in \mathbb{G}$ and sends g and g_1 to \mathcal{B} . Then, a group of parameters are generated by \mathcal{B} as follows:

- (1) \mathcal{B} selects a random oracle $H : \{0, 1\}^* \rightarrow Z_p^*$ and sends p, \mathbb{G}, g , and H to \mathcal{A} .
 - (2) \mathcal{B} randomly selects $\alpha_1, \alpha_2 \in Z_p^*$ and computes $A_1 = g^{\alpha_1}$ and $A_2 = g^{\alpha_2}$. Then, it checks c_0 and sends α_1 and α_2 to \mathcal{A} if $c_0 = 1$. Otherwise, it sends A_1 and A_2 .
 - (3) \mathcal{B} randomly selects $\beta_1, \beta_2 \in Z_p^*$ and computes $B_1 = g^{\beta_1}$ and $B_2 = g^{\beta_2}$. Then, it checks c_0 and sends β_1 and β_2 to \mathcal{A} if $c_0 = 0$. Otherwise, it sends B_1 and B_2 to \mathcal{A} .
- (3) *Query Phase.* \mathcal{A} adaptively issues a polynomially bounded number of queries to the following oracles:
- (1) *Attribute Oracle.* \mathcal{B} maintains a list $L_{\text{att}} = \left\{ \left\{ \text{att}_i, \varepsilon_{1,i}, \varepsilon_{2,i}, \gamma_{1,i}, \gamma_{2,i}, T_{1,i}, T_{2,i}, U_{1,i}, U_{2,i} \right\} \right\}$ and scans it when \mathcal{A} makes a query for parameter associated with an attribute att_i . It returns the corresponding $T_{1,i}, T_{2,i}, U_{1,i}$, and $U_{2,i}$ if att_i is in L_{att} . Otherwise, it randomly chooses $\varepsilon_{1,i}, \varepsilon_{2,i}, \gamma_{1,i}, \gamma_{2,i} \in Z_p^*$ and $c_1 \in 0, 1$ under the Bernoulli distribution with $\theta \in (0, 1)$, that is, $\Pr[c_1 = 0] = \theta$ and $\Pr[c_1 = 1] = 1 - \theta$. If $c_1 = 0$, \mathcal{B} computes $T_{1,i} = g^{\varepsilon_{1,i}}, T_{2,i} = g^{\varepsilon_{2,i}}, U_{1,i} = g^{\gamma_{1,i}}$, and $U_{2,i} = g^{\gamma_{2,i}}$. Otherwise, it computes $T_{1,i} = g_1^{\varepsilon_{1,i}}, T_{2,i} = g_1^{\varepsilon_{2,i}}, U_{1,i} = g_1^{\gamma_{1,i}}$, and $U_{2,i} = g_1^{\gamma_{2,i}}$. Then, \mathcal{B} checks c_0 and sends $\varepsilon_{1,i}, \varepsilon_{2,i}, U_{1,i}$, and $U_{2,i}$ to \mathcal{A} . Otherwise, it sends $T_{1,i}, T_{2,i}, \gamma_{1,i}$, and $\gamma_{2,i}$ to \mathcal{A} . Finally, it adds the new tuple $\left\{ \text{att}_i, \varepsilon_{1,i}, \varepsilon_{2,i}, \gamma_{1,i}, \gamma_{2,i}, T_{1,i}, T_{2,i}, U_{1,i}, U_{2,i} \right\}$ into L_{att} .
 - (2) *Key Extracting Oracle.* \mathcal{B} maintains a list $L_{\text{sk}} = \left\{ \left\{ S, r, D_1, \forall \text{att}_i \in S : D_{2,i} \right\} \right\}$ and scans L_{att} when \mathcal{A} issues a query to the key extracting oracle with an attribute set S . \mathcal{B} will abort the game if there exists an attribute att_i that is not in L_{att} because it is unable to coherently answer the query. We denote this abortion by E_1 . If not, \mathcal{B} scans L_{sk} and returns $S, r, D_1, \forall \text{att}_i \in S : D_{2,i}$ if S is in L_{sk} . Otherwise, \mathcal{B} randomly selects $r, R \in Z_p^*$ and computes $D = R$ and $D_{2,i} = (\varepsilon_{1,i} + \gamma_{1,i} - r) / (\varepsilon_{2,i} + \gamma_{2,i})$ for each attribute att_i in S . Then, \mathcal{B} returns $\left\{ S, D_1, \forall \text{att}_i \in S : D_{2,i} \right\}$ to \mathcal{A} if $c_0 = 0$, otherwise, returns $\left\{ S, r, D_1, \forall \text{att}_i \in S : D_{2,i} \right\}$ to \mathcal{A} . Finally, \mathcal{B} adds the new tuple $\left\{ S, r, D_1, \forall \text{att}_i \in S : D_{2,i} \right\}$ into L_{sk} .
 - (3) *Signing Oracle.* \mathcal{B} maintains a list $L_{\text{sig}} = \left\{ \left\{ m, \mathcal{T}, s, s', \left\{ \sigma_{1,x}, \sigma_{2,x} \right\}, \sigma_3, \sigma_4 \right\} \right\}$ and scans L_{att} when \mathcal{A} makes a signing query on a message m with predicate \mathcal{T} . The game will be aborted if there exists an attribute att_i of \mathcal{T} that is not in L_{att} . We denote this abortion by E_2 . Otherwise, \mathcal{B} checks c_1 for each attribute of \mathcal{T} . The game will be aborted, which is denoted by E_3 , if there exists an attribute att_i such that $c_1 = 1$. If not, \mathcal{B} generates a random attribute set S and runs the key extracting oracle to obtain $S, D_1, \forall \text{att}_i$

$\in S : D_{2,i}$. Then, it computes $\sigma_{1,x} = (T_{1,i} U_{1,i})^{q_x(0)}$ and $\sigma_{2,x} = D_{2,i} q_x(0)$ for each leaf node x , where $q_x(0)$ is generated as the real Sign algorithm. Then, it randomly chooses $s, s' \in Z_p^*$ to compute $\sigma_3 = R(s + s')$ and $Z = (A_1 B_1)^{(s+s')}$. Meanwhile, $\sigma_4 = H((A_1 B_1)^{(s+s')} \| m)$ is computed by the random oracle H . Finally, \mathcal{B} adds the new tuple $\left\{ m, \mathcal{T}, s, s', \left\{ \sigma_{1,x}, \sigma_{2,x} \right\}, \sigma_3, \sigma_4 \right\}$ into L_{sig} and sends $\left\{ \mathcal{T}, \left\{ \sigma_{1,x}, \sigma_{2,x} \right\}, \sigma_3, \sigma_4 \right\}$ to \mathcal{A} .

- (4) *Forge Phase.* \mathcal{A} forges a signature $\sigma^* = \left\{ \mathcal{T}^*, \left\{ \sigma_{1,x}^*, \sigma_{2,x}^* \right\}, \sigma_3^*, \sigma_4^* \right\}$ of message m^* . \mathcal{B} will abort the game if there exist attributes of \mathcal{T}^* that is not in L_{att} . We denote this abortion by E_4 . Otherwise, \mathcal{B} checks c_1 for each attribute that every leaf node is associated with. If there exists a $c_1 = 0$, \mathcal{B} will abort the game. We denote this abortion by E_5 . If not, \mathcal{B} runs the verifying oracles as follows:

(1) If x is a leaf node, it runs $\text{VerifyNode}(x)$ as

$$\text{VerifyNode}(x) = \frac{(T_{2,i} U_{2,i})^{\sigma_{2,x}^*}}{\sigma_{1,x}^*} = \begin{cases} g_1^{-r(q_x(0)+q_x'(0))}, & \text{if } \mathcal{T}_x^*(S_{\min}) = 1, \\ g_1^{-r q_x'(0)}, & \text{if } \mathcal{T}_x^*(S_{\min}) = 0. \end{cases} \quad (4)$$

- (2) If x is a nonleaf node, we denote the computation result of its arbitrary child node z by F_z . It runs $\text{VerifyNode}(x)$ as equation (5). By those finite recursions, \mathcal{B} can validate the forged signature as the real F2P-ABS. We say \mathcal{A} wins the game if $Z = \sigma_4^*$. In this case, \mathcal{B} chooses a tuple $\left\{ m, \mathcal{T}, s, s', \left\{ \sigma_{1,x}, \sigma_{2,x} \right\}, \sigma_3, \sigma_4 \right\}$ from L_{sig} and a tuple $\left\{ S, r, D_1, \forall \text{att}_i \in S : D_{2,i} \right\}$ and finally returns $(\alpha_1 + \beta_1 + r) / \sigma_3^* (\alpha_2 + \beta_2)$ to \mathcal{C} as the answer to DL problem:

$$\text{VerifyNode}(x) = \prod_{z \in \Lambda(x)} F_z^{\Delta_{z,\Lambda(x)}(0)} = \prod_{z \in \Lambda(x), \mathcal{T}_z(S_{\min})=1} g_1^{-r(q_z(0)+q_z'(0))} \cdot \prod_{z \in \Lambda(x), \mathcal{T}_z(S_{\min})=0} g_1^{-r q_z'(0)} = \begin{cases} g^{-r(q_x(0)+q_x'(0))}, & \text{if } \mathcal{T}_x^*(S_{\min}) = 1, \\ g^{-r q_x'(0)}, & \text{if } \mathcal{T}_x^*(S_{\min}) = 0. \end{cases} \quad (5)$$

Lemma 1. *If E_1, E_2 , and E_3 do not happen, all results output by \mathcal{B} in the query phase are valid and indistinguishable.*

Lemma 1 holds because

- (1) For any query to the attribute oracle with an attribute att_i , \mathcal{B} selects random $\varepsilon_{1,i}, \varepsilon_{2,i}, \gamma_{1,i}, \gamma_{2,i} \in Z_p^*$ and a random $c_1 \in \{0, 1\}$. No matter what the value of c_1 is, \mathcal{A} will be obtained. We note that $\varepsilon_{1,i}, \varepsilon_{2,i}, U_{1,i}$, and $U_{2,i}$ are all randomly chosen, which are valid and indistinguishable from those generated by the real F2P-ABS. Likewise, \mathcal{A} will obtain $T_{1,i}, T_{2,i}, \gamma_{1,i}$, and $\gamma_{2,i}$ if $c_0 = 1$, and those are also randomly chosen and indistinguishable.
- (2) For any query to the key extracting oracle, \mathcal{B} selects two random numbers $r, R \in Z_p^*$ if E_1 does not happen. It generates a tuple $S, D_1, \{\forall \text{att}_i \in S : D_{2,i}\}$ where $D_1 = R$ and $D_{2,i} = (\varepsilon_{1,i} + \gamma_{1,i} - r) / (\varepsilon_{2,i} + \gamma_{2,i})$. If $c_0 = 0$, there always exist $\beta'_1, \beta'_2, r', \gamma'_{1,i}, \gamma'_{2,i} \in Z_p^*$ such that $D_1 = (\alpha_1 + \beta'_1 + r') / (\alpha_2 + \beta'_2)$ and $D_{2,i} = (\varepsilon_{1,i} + \gamma'_{1,i} - r') / (\varepsilon_{2,i} + \gamma'_{2,i})$. If $c_0 = 1$, there always exist $\alpha'_1, \alpha'_2, \varepsilon'_{1,i}, \varepsilon'_{2,i} \in Z_p^*$ such that $D_1 = (\alpha'_1 + \beta_1 + r) / (\alpha'_2 + \beta_2)$ and $D_{2,i} = (\varepsilon'_{1,i} + \gamma_{1,i} - r) / (\varepsilon'_{2,i} + \gamma_{2,i})$. Therefore, the output is valid and indistinguishable from real signing keys.
- (3) For any query to the signing oracle, each leaf node of predicate has a corresponding attribute in L_{att} if E_2 does not happen. And, if E_3 does not happen, \mathcal{B} will generate a tuple $S, D_1, \{\forall \text{att}_i \in S : D_{2,i}\}$. Then, it computes the tuple $\{\mathcal{T}, \{\sigma_{1,x}, \sigma_{2,x}\}, \sigma_3, \sigma_4\}$ in a way of the real F2P-ABS. So, the output of the signing oracle is valid and indistinguishable from real signatures.

Lemma 2. *If E_4 and E_5 do not happen, \mathcal{B} can probably solve DLP when \mathcal{A} successfully breaks the F2P-ABS.*

Lemma 2 holds because if E_4 and E_5 do not happen, each att_i has corresponding $T_{1,i} = g_1^{\varepsilon_{1,i}}, T_{2,i} = g_1^{\varepsilon_{2,i}}, U_{1,i} = g_1^{\gamma_{1,i}}$, and $U_{2,i} = g_1^{\gamma_{2,i}}$ in L_{att} . In this case, \mathcal{B} can output the answer to the DL problem if it finds the proper r in L_{sk} and proper s and s' in L_{sig} .

Suppose \mathcal{A} makes at most q_{att} queries to the attribute oracle, q_{sk} queries to the key extracting oracle and q_{sig} queries to the signing oracle. Let t be the time required for \mathcal{A} to break F2P-ABS. Let t_a and t_m be the time required for one-time addition and scalar multiplication operation, respectively, in Z_p^* . Let t_e be the time required for one-time exponentiation in \mathbb{G} . The total time t' required for \mathcal{B} to solve DL problem satisfies

$$t' \leq t + (3q_{sk} + q_{\text{sig}})t_a + (q_{sk} + (n+1)q_{\text{sig}})t_m + (4q_{\text{att}} + (n+1)q_{\text{sig}})t_e. \quad (6)$$

The probability that E_1, E_2 , and E_4 do not happen is $\Pr[\overline{E_1} \wedge \overline{E_2} \wedge \overline{E_4}] = (q_{\text{att}}/2^l)^{nq_{sk} + nq_{\text{sig}} + 1}$. The probability that E_3 and E_5 do not happen is $\Pr[\overline{E_3} \wedge \overline{E_5}] = \theta^{nq_{sk}} (1 - \theta)^n$ which achieves its upper bound when $\theta = nq_{sk} / (1 + nq_{sk})$. That is, $\Pr[\overline{E_3} \wedge \overline{E_5}] \leq (nq_{sk} / (1 + nq_{sk}))^{nq_{sk}} (1 / (1 + nq_{sk}))^n$. Considering all situations mentioned above, the advantage ε' of \mathcal{B} to solve DL problem satisfies

$$\varepsilon' \leq \left(\frac{q_{\text{att}}}{2^l}\right)^{(nq_{sk} + nq_{\text{sig}} + 1)} \left(\frac{nq_{sk}}{1 + nq_{sk}}\right)^{nq_{sk}} \left(\frac{1}{1 + nq_{sk}}\right)^n \varepsilon. \quad (7)$$

7. Performance Analysis

In this section, we analyze the performance of our F2P-ABS scheme against current ABS schemes [24–27] in Tables 2 and 3 in terms of public parameter size, secret key size and signing key size, signature size, and computation overhead (number of required exponentiation and pairing computation, which are time-consuming operations). For clarity, we list some essential notations for analysis in Table 5.

As is shown in Table 2, the size of the public parameter in [25] increases fast with the square of the number k of authorities, which is less practical for initializing the system. The size of public parameter in [26, 27] is linear to the size of attribute universe. To mitigate key escrow problem, we assign part of public parameters to the AA and the DSC, respectively. As a result, our F2P-ABS has a group of public parameters two times the size of that in [26, 27]. The secret key in F2P-ABS has $2a_u + 4$ group elements, which is linear to the size of attribute universe. Although F2P-ABS has big-sized public parameters and secret key, it hardly causes performance dilemma considering significant storage capacity of the AA and the DSC.

We achieve a short signing key size compared to existing schemes [24–27]. The signing key size in [26] is linear to the size of attribute universe, which is a huge challenge to the storage capacity of signer when large numbers of attributes are engaged. The signing key of F2P-ABS consists of $a_{sk} + 1$ group elements, while that for [24, 25] is $2a_{sk} + 1$. Meanwhile, signing key in [27] consists of $2a_{sk} + 3$ group elements. Hence, we successfully reduce the number of group elements in signing key by half. The signature of F2P-ABS totally has $2a_{sp} + 2$ group elements, which is slightly less than that for [27].

Since pairing computation is considered the most expensive operation, number of pairing operations is a critical factor of feasibility in practical scenes. We note that there is a signcryption in [27], which is a mixture of ABE and ABS. It requires $a_{sp} + 3$ pairings for verification and 2 pairings to recover the message plaintext. The number of pairings in [24, 25] is also linear to the size of the minimum signing attribute set and the size of signing predicate, respectively. As clearly seen in Table 3, F2P-ABS requires no pairing for verification. This is even less than that for [26], which is 3 pairing operations. In addition, it achieves light computation at the cost of large signing key size being linear to the size of attribute universe. It is worth noting that garbled circuit operations become the only performance bottleneck of F2P-ABS. Although we are unsatisfied with existing garbled circuit implementation, we believe there will be better ones in future which help to achieve better performance in practical scenes. Hence, F2P-ABS is efficient from computation point of view.

To sum up, our F2P-ABS achieves a better performance, in terms of signing key size, signature size, and computation cost, than existing schemes. Furthermore, it achieves other

TABLE 5: Notations.

Notation	Definition
a_u	Size of the attribute universe
a_{sk}	Size of the signer attribute set
a_{sp} (a_{ep})	Size of the signing (encrypting) predicate
n_{ep}	Number of nonleaf nodes of a signing predicate
ϕ_{sp} (ϕ_{ep})	Size of the minimum signing (encrypting) attribute set
k	Number of attribute authorities
$ \mathbb{G} $ ($ \mathbb{G}_T $)	bit size of an element of a group \mathbb{G} (\mathbb{G}_T)
$ Z_p^* $	bit size of an element of a group Z_p^*
Param.	Size of the public parameter
Sec.	Size of the secret key
Sig.	Size of the signature
Exp.	Number of exponentiation operations
Pair.	Number of pairing operations

desirable goals such as expressive predicate, authenticity, and signer privacy.

8. Conclusion

ABS is a promising cryptographic primitive for anonymous authentication. However, existing ABS approaches have drawbacks with respect to security and efficiency, which hinders their application for authentication in mobile platforms. We focus on constructing an ABS scheme that is elegantly free from key escrow problem and heavy pairing operations. In addition, the size of the signing key is much less than existing schemes. Without the loss of security, it achieves perfect signer privacy. It is proven to guarantee existential unforgeability under corrupted and adaptive chosen predicate and message attack, which is securer than existing schemes. Therefore, it outperforms existing schemes and shows more feasibility for anonymous authentication in mobile platforms.

To further its application, we are exploiting a new method to enhance efficiency. Considering its performance bottleneck, our future work will build on the preliminary construction in this work to develop the proposed scheme by improving the performance of garbled circuit operations.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This study was supported by the National Natural Science Foundation of China under grants 61972208, 61373135, 61672299, and 61170276.

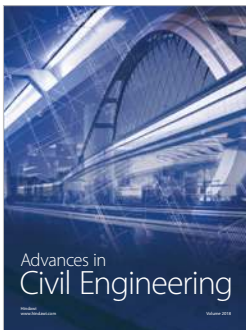
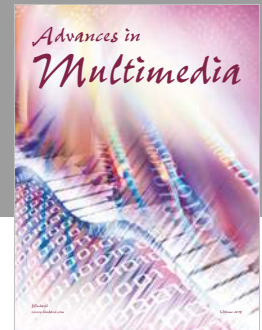
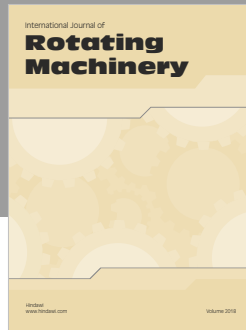
References

- [1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: vision, hype,

and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

- [2] N. Santos, K. P. Gummadi, and R. Rodrigues, "Towards trusted cloud computing," Edited by S. Sahu and P. J. Shenoy, Eds., in *Proceedings of the 2009 Conference on Hot Topics in Cloud Computing, HotCloud '09*, vol. 3, USENIX, Berkeley, CA, USA, June 2009.
- [3] L. Aniello, R. Baldoni, E. Gaetani, F. Lombardi, A. Margheri, and V. Sassone, "A prototype evaluation of a tamper-resistant high performance blockchain-based transaction log for a distributed database," in *Proceedings of European Dependable Computing Conference (EDCC 2017)*, pp. 151–154, IEEE, Geneva, Switzerland, September 2017.
- [4] H. K. Maji, M. Prabhakaran, and M. Rosulek, "Attribute-based signatures," in *Topics in Cryptology—CT-RSA 2011 Lecture Notes in Computer Science*, A. Kiayias, Ed., vol. 6558, pp. 376–392, Springer, Berlin, Germany, 2011.
- [5] D. Boneh and M. K. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology—CRYPTO 2001 Lecture Notes in Computer Science*, J. Kilian, Ed., vol. 2139, pp. 213–229, Springer, Berlin, Germany, 2001.
- [6] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proceedings of the 15th ACM Conference on Computer and Communications Security, CCS '08*, P. Ning, P. F. Syverson, and S. Jha, Eds., pp. 417–426, ACM, New York, NY, USA, October 2008.
- [7] Z. Liu, H. Yan, and Z. Li, "Server-aided anonymous attribute-based authentication in cloud computing," *Future Generation Computer Systems*, vol. 52, pp. 61–66, 2015.
- [8] J. Liu, X. Huang, and J. K. Liu, "Secure sharing of personal health records in cloud computing: ciphertext-policy attribute-based signcryption," *Future Generation Computer Systems*, vol. 52, pp. 67–76, 2015.
- [9] G. Lin, H. Hong, and Z. Sun, "A collaborative key management protocol in ciphertext policy attribute-based encryption for cloud data sharing," *IEEE Access*, vol. 5, pp. 9464–9475, 2017.
- [10] G. Lin, L. You, B. Hu et al., "A coordinated ciphertext policy attribute-based PHR access control with user accountability," *KSII Transactions on Internet and Information Systems*, vol. 12, no. 4, pp. 1832–1853, 2018.
- [11] S. S. M. Chow, "Removing escrow from identity-based encryption," in *Public Key Cryptography—PKC 2009 Lecture Notes in Computer Science*, S. Jarecki and G. Tsudik, Eds., vol. 5443, pp. 256–276, Springer, Berlin, Germany, 2009.
- [12] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS '09*, E. Al-Shaer, S. Jha, and A. D. Keromytis, Eds., pp. 121–130, ACM, New York, NY, USA, November 2009.
- [13] J. Hur, "Improving security and efficiency in attribute-based data sharing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 10, pp. 2271–2282, 2013.
- [14] H. Cui, G. Wang, R. H. Deng, and B. Qin, "Escrow free attribute-based signature with self-revealability," *Information Sciences*, vol. 367–368, pp. 660–672, 2016.
- [15] L. Chen, Z. Cheng, and N. P. Smart, "Identity-based key agreement protocols from pairings," *International Journal of Information Security*, vol. 6, no. 4, pp. 213–241, 2007.
- [16] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances in Cryptology—EUROCRYPT 2005 Lecture Notes in*

- Computer Science*, R. Cramer, Ed., vol. 3494, pp. 457–473, Springer, Berlin, Germany, 2005.
- [17] V. Goyal, O. Pandey, A. Sahai et al., “Attribute-based encryption for fine-grained access control of encrypted data,” in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS ’06)*, A. Juels, R. N. Wright, and S. D. C. di Vimercati, Eds., pp. 89–98, ACM, New York, NY, USA, October 2006.
- [18] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in *Proceedings of 2007 IEEE Symposium on Security and Privacy (SP ’07)*, pp. 321–334, IEEE, Washington, DC, USA, May 2007.
- [19] G. Zhang, L. Liu, and Y. Liu, “An attribute-based encryption scheme secure against malicious KGC,” in *Proceedings of 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom 2012)*, G. Min, Y. Wu, L. C. Liu, X. Jin, S. A. Jarvis, and A. Y. Al-Dubai, Eds., pp. 1376–1380, IEEE, Liverpool, UK, June 2012.
- [20] B. Waters, “Efficient identity-based encryption without random oracles,” in *Advances in Cryptology—EUROCRYPT 2005 Lecture Notes in Computer Science*, R. Cramer, Ed., vol. 3494, pp. 114–127, Springer, Berlin, Germany, 2005.
- [21] P. Yang, Z. Cao, and X. Dong, “Fuzzy identity based signature,” *IACR Cryptology ePrint Archive*, vol. 2008, p. 2, 2008.
- [22] J. Li, M. H. Au, W. Susilo, D. Xie, and K. Ren, “Attribute-based signature and its applications,” in *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS ’10)*, D. Feng, D. A. Basin, and P. Liu, Eds., pp. 60–69, ACM, Beijing, China, April 2010.
- [23] M. Gagne, S. Narayan, and R. Safavi-Naini, “Short pairing-Efficient threshold-attribute-based signature,” in *Pairing-Based Cryptography—Pairing 2012 Lecture Notes in Computer Science*, M. Abdalla and T. Lange, Eds., vol. 7708, pp. 295–313, Springer, Berlin, Germany, 2012.
- [24] J. Su, D. Cao, B. Zhao, X. Wang, and I. You, “EPASS: an expressive attribute-based signature scheme with privacy and an unforgeability guarantee for the Internet of things,” *Future Generation Computer Systems*, vol. 33, pp. 11–18, 2014.
- [25] J. Li, X. Chen, and X. Huang, “New attribute-based authentication and its application in anonymous cloud access service,” *International Journal of Web and Grid Services*, vol. 11, no. 1, pp. 125–141, 2015.
- [26] Y. S. Rao and R. Dutta, “Efficient attribute-based signature and signcryption realizing expressive access structures,” *International Journal of Information Security*, vol. 15, no. 1, pp. 81–109, 2016.
- [27] Y. S. Rao, “Signature-policy attribute-based key-insulated signature,” *IET Information Security*, vol. 11, no. 1, pp. 23–33, 2017.
- [28] D. Malkhi, N. Nisan, B. Pinkas et al., “Fairplay-secure two-party computation system,” in *Proceedings of the 13th Conference on USENIX Security Symposium (SSYM ’04)*, M. Blaze, Ed., pp. 287–302, USENIX, San Diego, CA, USA, August 2004.
- [29] Y. Huang, D. Evans, J. Katz et al., “Faster secure two-party computation using garbled circuit,” in *Proceedings of the 20th USENIX Conference on Security (SEC ’11)*, pp. 331–335, USENIX, Berkeley, CA, USA, November 2011.
- [30] G. Brassard, C. Crepeau, and J. Robert, “All-or-nothing disclosure of secrets,” in *Advances in Cryptology—CRYPTO ’86 Lecture Notes in Computer Science*, A. M. Odlyzko, Ed., vol. 263, pp. 234–238, Springer, Berlin, Germany, 1986.
- [31] W. Tzeng, “Efficient 1-out-of-n oblivious transfer schemes with universally usable parameters,” *IEEE Transactions on Computers*, vol. 53, no. 2, pp. 232–240, 2004.
- [32] E. Van Heyst and T. P. Pedersen, “How to make efficient fail-stop signatures,” in *Advances in Cryptology—EUROCRYPT ’92 Lecture Notes in Computer Science*, R. A. Rueppel, Ed., vol. 658, pp. 366–377, Springer, Berlin, Germany, 1993.
- [33] T. Okamoto and K. Takashima, “Decentralized attribute-based signatures,” in *Public-Key Cryptography—PKC 2013 Lecture Notes in Computer Science*, K. Kurosawa and G. Hanaoka, Eds., vol. 7778, pp. 125–142, Springer, Berlin, Germany, 2013.



Hindawi

Submit your manuscripts at
www.hindawi.com

