Received 26 September 2023; accepted 12 October 2023. Date of publication 17 October 2023; date of current version 2 November 2023.

Digital Object Identifier 10.1109/OJCOMS.2023.3325106

Two-Party Adaptor Signature Scheme Based on IEEE P1363 Identity-Based Signature

XINJIE ZHU[1,2], DEBIAO HE[1,3] (Member, IEEE), ZIJIAN BAO[1], CONG PENG[1], AND MIN LUO[1,4]

[1]School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China

[2]Institute of Information Technology, Shenzhen Institute of Information Technology, Shenzhen 518172, China

[3]Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center, Qilu University of Technology (Shandong Academy of Sciences), Jinan 250014, China

[4]Shanghai Key Laboratory of Privacy-Preserving Computation, Matrix Elements Technologies, Shanghai 201204, China

CORRESPONDING AUTHORS: D. HE AND Z. BAO (e-mail: hedebiao@163.com; baozijian@whu.edu.cn)

This work was supported in part by the Major Research Plan of Hubei Province under Grant 2023BAA027; in part by the Shandong Provincial Key Research and Development Program under Grant 2021CXGC010107; in part by the National Natural Science Foundation of China under Grant 62202339, Grant 62172307, and Grant U21A20466; in part by the New 20 Project of Higher Education of Jinan under Grant 202228017; in part by the Special Project on Science and Technology Program of Hubei Province under Grant 2020AEA013 and Grant 2021BAA025; and in part by the Fundamental Research Funds for the Central Universities under Grant 2042023KF0203.

ABSTRACT Adaptor signature is extended from the standard digital signature, which conceals a secret value in the "pre-signature". The one who knows the secret value can transform it into a completed signature. Adaptor signatures are useful in addressing blockchain scalability issues, and have been used to build blockchain scaling protocols, such as payment channels or atomic swaps. Recently, someone propose two-party adaptor signature to enhance the privacy in application of adaptor signatures. However, we notice that there is no identity-based two-party adaptor signature scheme so far. Based on IEEE P1363 standard identity-based signature scheme, a secure two-party adaptor signature is constructed in this paper. Moreover, under the random oracle model, we analyze the security of our proposed P1363-based two-party adaptor signature. Finally, we make a comparison between our scheme and other adaptor signature schemes in computation and communication overheads, the result shows the costs of our scheme are acceptable.

INDEX TERMS Two-party signature, adaptor signature, IEEE P1363, payment channel.

I. INTRODUCTION

THE EVOLVING blockchain technology has attracted a surge of interest from academia, industry and government agencies to enable secure payments in a decentralized system. However, most of the blockchain applications need to face the problem of scalability of the blockchain platform [1]. For example, Bitcoin can only process about 10 transactions per second, While the throughput of VisaNet is almost 1700 transactions. This situation severely limits the development of blockchain applications and degrades the user experience.

Adaptor signature is extended from the standard digital signature. It was first proposed by Poelstra [2] to help build blockchain scaling protocols, such as Payment Channel Networks (PCNs) [3], and was later formalized by Aumayr et al. [4]. In adaptor signatures, the signer creates a pre-signature with a hidden hard relation, then the one who knows the hard relation witness can convert the pre-signature to a completed one. Meanwhile, we can verify the completed signature with standard signature verification algorithms. Anyone is able to extract the secret witness with the pre-signature and the completed signature.

Some blockchain scaling protocols aim to handle the transactions offchain and only put the final state onchain, such as payment channels [5]. With adaptor signatures, the transaction sender can pre-sign a transaction to lock some coins with a hard relation such as hash pre-image, then the witness can be selled, and the buyer can adapt the

pre-signature and make a transaction to spend the coins. While the hard relation behind the pre-signature can be set with multiple hash values and different hash algorithms, the related blockchain applications can be more flexible and efficient, such as payment channels, payment routing in PCNs [6] or atomic swaps [7], [8], [9]. Many applications behind it are used to address blockchain scalability issues, which shows huge potential of adaptor signatures in blockchain scalability.

A. MOTIVATION

While applying adaptor signature to blockchain application, the verification of the constructed payment transaction consist of two signatures, one for adaptor signature and another for transaction signature, which can be used to distinguish it from a normal payment transaction. Some [10], [11] suggest using two-party adaptor signatures to solve this drawback, the main idea of which is to replace the two verification steps of adaptor signature and transaction signature by one verification. With two-party adaptor signature, the transactions in the application such as payment channels are generated collaboratively by the two users in the channel, and acted the same as normal transactions. Therefore the application of two-party adaptor signature can prevent this kind of transactions from censorship.

Meanwhile, we notice that there is no identity-based two-party adaptor signature scheme so far. With identity-based cryptosystem, people can replace public key with other's identity (e.g., email address). It is more user-friendly than public-key cryptosystem, and simplifies key management. Recently, there have been a number of efforts trying to apply identity-based cryptosystems to blockchain that have achieved positive effects, such as combination with the Internet-of-Things (IoT) [12], [13], which removes the dependency of public key infrastructure (PKI), and improves efficiency and transparency of the systems. HIBEChain [14] is an good example that shows blockchain system based on identity-based cryptosystem can achieve high scalability and accountability, which demonstrates the potential of identity-based blockchain systems for large-scale industrial IoT applications. Observing the trend towards the application of identity-based cryptosystems in the blockchain, we believe it is necessary to build a two-party identity-based adaptor signature scheme, so we make the following contributions.

B. OUR CONTRIBUTIONS

In this paper, we construct a two-party identity-based adaptor signature protocol for the IEEE P1363 standard [15]. Specifically, the pre-signature in our proposed protocol is generated with the communication of two parties. We also implement our protocol on a device and compare its performance with other schemes. The main contributions are summarized as below:

- We propose a secure two-party adaptor signature scheme based on IEEE P1363 standard identity-based signature scheme. The distributed pre-signature generation phase works with the communication of two parties.
- We give a security proof of our adaptor signature scheme under the random oracle model. The result shows that our scheme meets the security properties of the two-party adaptor signature.
- We evaluate the performance of our scheme, and make a comparison with other adaptor signature schemes.

C. ORGANIZATION

The remaining of this paper is structured as follows. In Section II, we review the related work of adaptor signature schemes. In Section III, we introduce some preliminaries. In Section IV, we give basic concepts. The details of the proposed two-party adaptor signature are presented in Section V, and we analyze its security in Section VI. Findings from the performance evaluation are shown in Section VII. Finally, Section VIII concludes this paper.

II. RELATED WORK

Poelstra [2] first introduced a concept named "scriptless scripts", which gave the notion of adaptor signature, and proved it was useful in blockchain scaling applications such as PCNs, payment channel hubs or atomic swaps. Then Aumayr et al. [4] formalized the adaptor signature as a standalone primitive, the schemes in their work are constructed with Schnoor and ECDSA signatures. Fournier [16] also tried to give a formal definition of adaptor signature at the same time, but his definition was weaker, and not suitable for some applications. Concurrently, Thyagarajan and Malavolta [17] proposed a scheme named lockable signature, which is similar to adaptor signature, and can be built with any signature scheme. Lockable signature is a weaker primitive than adaptor signature, as it needs an honestly created partial signature (e.g., through MPC protocols) and the witness of the hard relation must be known while generating signatures.

Malavolta et al. [10] presented two-party threshold adaptor signature protocols with Schnoor and ECDSA signatures, and built an anonymous multi-hop lock mechanism based on them. However, they did not formalize the two-party adaptor signature, therefore a formal security proof for these schemes is missing. Then Erwig et al. [11] gave the definition of two-party adaptor signatures, and showed a generic way to construct adaptor signature schemes with identification schemes. Klamti and Hasan [18] constructed a two-party adaptor signature scheme, which is built with code theory, thus has a huge costs of computation and big pre-signature size.

Additionally, Qin et al. [19] also gave a generic approach to construct adaptor signature from identification schemes, but their construction is more practical for the lattice-based instantiation. Meanwhile, they proposed adaptor blind signature and linkable ring adaptor signature. After that, Dai et al. [20] gave a formalization definition for unlinkability of adaptor signatures, and designed a generic method to construct unlinkable adaptor signatures. Their approach

allowed more Flexible instantiations and minimal assumptions. Moreno-Sanchez et al. [21] constructed an adaptor signature scheme for the linkable ring signature in Monero, which change the transaction script logic and improve its efficiency in largescale applications. Tairi et al. [22] proposed a primitive named anonymous atomic locks, which can be used to achieve three-party conditional transactions in payment channel hubs, and give an instantiation using adaptor signatures.

With the development of quantum computers, many digital signature schemes are under threat of being broken, and adaptor signature is under the same situation. Therefore, it is important to build secure adaptor signature schemes that can resist quantum attacks. Esgin et al. [23] designed the first lattice-based adaptor signature scheme LAS, which relies on standard lattice assumptions, namely SIS and LWE. However, the LAS protocol has some shortcuts in communication overhead and privacy protection, which limits its application. Then Tairi et al. [24] designed an adaptor signature scheme based on isogenies named IAS, which needs lower storage space than LAS, but with higher computation cost. Further more, Gilchrist [25] introduced a post-quantum adaptor signature using SQISign signature, which is based on Isogeny and therefore require less storage.

III. PRELIMINARIES

In this section, we briefly review the related preliminaries and basic security assumptions used in this paper.

A. BILINEAR MAPS

Let (G_1, G_2, G_T) be a set of three cycle groups with the same order q, then $e : G_1 \times G_2 \to G_T$ is a bilinear map [26] with following three properties:

1. Bilinear: For all $g_1 \in G_1, g_2 \in G_2, \forall a, b \in \mathbb{Z}_q, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$;
2. Nondegenerate: $\exists g_1 \in G_1, g_2 \in G_2, e(g_1, g_2) \neq 1$;
3. Computable: Given the elements $g_1 \in G_1, g_2 \in G_2$, the $e(g_1, g_2)$ can be computed efficiently.

B. DIGITAL SIGNATURE

Generally, a digital signature [27] scheme $\Sigma = (\text{Gen}, \text{Sign}, \text{Vrfy})$ includes three algorithms:

- $\text{Gen}(1^\lambda)$: Given a security parameter λ as input, this algorithm outputs a pair of private key sk and public key pk.
- $\text{Sign}_{sk}(m)$: Given private key sk and message m as input, this algorithm outputs a signature σ.
- $\text{Vrfy}_{pk}(m, \sigma)$: Given public key pk, message m and signature σ as input, this algorithm outputs a verify result $b \in \{0, 1\}$. If $b = 1$, then the signature is valid; otherwise, it is invalid.

A secure digital signature scheme must meet the following two properties:

1) *correctness*: For all message m and valid key pair (sk, pk), it holds that $\text{Vrfy}_{pk}(m, \text{Sign}_{sk}(m)) = 1$.
2) *(strong) existential unforgeability under chosen message attack* (EUF-CMA or SUF-CMA): In EUF-CMA, for a public key pk, a $\mathcal{PPT}$ adversary is not able to generate a valid signature on a fresh message m even with ability to access a signing oracle. In SUF-CMA, for arbitrary message m, it is impossible to generate a new valid signature on it with a $\mathcal{PPT}$ adversary.

C. IEEE STANDARD FOR IDENTITY-BASED SIGNATURE

We briefly introduce the IEEE standard for identity-based signature scheme [28]. The scheme consists of the following four algorithms:

- Setup: Given a security parameter λ, this algorithm generates public parameters PP as follows:
  a) Generates the cylic groups (G_1, G_2, G_T) with the same order q, and a bilinear map $e : G_1 \times G_2 \to G_T$.
  b) Chooses two random generators $Q_1 \in G_1$ and $Q_2 \in G_2$.
  c) Selects a random secret value s in $\mathbb{Z}_q^*$ as the secret key of server, then computes $P_{pub} = s \cdot Q_2$ and sets $g = e(Q_1, Q_2)$.
  d) Sets $PP = \{P_{pub}, g, Q_1, Q_2, G_1, G_2, G_T, e\}$.
- Extract: Given the public parameters PP, the server's secret s and the user's ID, this algorithm computes the user's identity $h_{ID} = H_1(ID)$, where H_1 is a hash function of $\{0, 1\}^* \to \mathbb{Z}_q^*$, then it outputs the user's private key $d_{ID} = (s + h_{ID})^{-1} \cdot Q_1$.
- Sign: Given the public parameters PP, the user's private key d_{ID}, and message m, this algorithm randomly chooses $r \in \mathbb{Z}_q$ and computes $u = g^r, h = H_2(m, u)$ and $S = (r + h) \cdot d_{ID}$, where H_2 is a hash function of $\{0, 1\}^* \times G_T \to \mathbb{Z}_q^*$, then it returns $\sigma = (h, s)$ as the signature of m.
- Verify: Given the public parameters PP, the user's public key h_{ID}, a message m and signature σ, this algorithm computes $u' = e(S, h_{ID} \cdot Q_2 + P_{pub}) \cdot g^{-h}$, then it returns 1 if $h = H_2(m, u')$; otherwise, returns 0.

D. HARD RELATION

Let R be a binary relation, L_R be the language for describing the relation, then we have $L_R := \{Y | \exists y \ s.t. \ (Y, y) \in R\}$. If L_R meets the following three properties, then it is a hard relation:

1. There exists a $\mathcal{PPT}$ algorithm $\text{GenR}(1^\lambda)$ that can take security parameter as input and output a pair $(Y, y) \in R$.
2. The relation R is decidable in poly-time.
3. For any $\mathcal{PPT}$ adversary, the probability of extracting a valid witness y for $Y \in L_R$ is negligible.

E. NON-INTERACTIVE ZERO-KNOWLEDGE PROOF

A non-interactive zero-knowledge (NIZK) [29] proof scheme consists of two algorithms: proof generate scheme $\pi \leftarrow \text{Prove}(Y, y)$ and proof verify scheme $\{0, 1\} \leftarrow \text{Verify}(Y, \pi)$.

With a NIZK protocol, the prover can convince the verifier that there exists a witness y for the statement Y without leaking any additional information. The NIZK proof used in our work needs to have the following three properties:

1. Completeness: $\text{Verify}(Y, Prove(Y, y)) = 1$ for all $(Y, y) \in R$.
2. Soundness: For any $(Y, y) \notin R$, $\text{Verify}(Y, Prove(Y, y)) = 0$ except with negligible probability.
3. Zero-knowledge: There exists a $\mathcal{PPT}$ simulator S that can output a proof π for any $(Y, y) \in R$.

IV. BASIC CONCEPTS

A. FORMAL DEFINITION

Basically, adaptor signature scheme has two steps: the signer generates a pre-signature for a message and a commitment of a secret value, then the person with the secret value has ability to convert the pre-signature into a completed signature. Different from the single-party adaptor signatures, in the two-party identity-based adaptor signatures, pre-signature is generated with the cooperation of two parties. Meanwhile, both the pre-signature and the signature should be valid under the user's identity h_{ID}. Here is the formal definition of our two-party identity-based adaptor signature [11].

Definition 1 (Two-Party Identity-Based Adaptor Signature): Given a hard relation R and a two-party digital signature $SIG_2 = (\text{Setup}, \text{Gen}, \Pi_{\text{sign}}, \text{Verify})$, a two-party adaptor signature is run between parties A_0, A_1 and consists of the following algorithms: (Setup, Gen, GenR, pSign, pVerify, Adapt, Ext). The details are described as follows:

- $\text{Setup}(1^\lambda)$: Given a system security parameter λ, this algorithm outputs a public parameter set PP.
- $\text{Gen}(PP)$: Given a public parameter set PP, this algorithm generates the user's private keys $D_{ID}^{A_1}$ and $D_{ID}^{A_2}$, which can be stored on two devices A_1 and A_2.
- $\text{GenR}(1^\lambda)$: Given a system security parameter λ, this algorithm generates a hard relation statement/witness pair $(z, Y) \in R$ and a zero-knowledge proof π.
- $\text{pSign}_{\langle D_{ID}^{A_1}, D_{ID}^{A_2}\rangle}(m, z, \pi)$: Given a message $m \in \{0,1\}^*$, the user's private keys $D_{ID}^{A_1}, D_{ID}^{A_2}$, a statement $z \in L_R$ and a zero-knowledge proof π, this algorithm generates a pre-signature $\tilde{\sigma}$ with the cooperation of two devices A_1 and A_2.
- $\text{pVerify}_{\text{h}_{\text{ID}}}(m, z, \tilde{\sigma})$: Given a message $m \in \{0,1\}^*$, the user's identity h_{ID}, a statement $z \in L_R$ and a pre-signature $\tilde{\sigma}$, this algorithm outputs a bit $b \in \{0,1\}$. If the pre-signature is valid, then $b = 1$; otherwise, $b = 0$.
- $\text{Adapt}(\tilde{\sigma}, Y)$: Given a pre-signature $\tilde{\sigma}$ and a witness Y, this algorithm generates a completed signature σ.
- $\text{Ext}(\sigma, \tilde{\sigma}, z)$: Given a signature σ, a pre-signature $\tilde{\sigma}$ and a hard relation statement z, this algorithm outputs a witness Y that satisfies $(z, Y) \in R$.

B. SECURITY MODEL

Here we give the correctness and security properties of the two-party identity-based adaptor signature [11].

Definition 2 (Two-Party Pre-Signature Correctness): A two-party adaptor signature satisfies two-party pre-signature correctness if for any message $m \in \{0,1\}^*$ and hard relation statement/witness pair $(z, Y) \in R$, it holds that:

$$\Pr\left[\begin{array}{c|l} \text{pVerify}_{\text{h}_{\text{ID}}}(m, & PP \leftarrow \text{Setup}(1^\lambda) \\ z; \tilde{\sigma}) = 1 \wedge & (D_{ID}^{A_1}, D_{ID}^{A_2}) \leftarrow \text{Gen}(PP) \\ \text{Verify}_{h_{ID}}(m; \sigma) = 1 & (z, Y, \pi) \leftarrow \text{GenR}(1^\lambda) \\ \wedge & \tilde{\sigma} \leftarrow \\ & \Pi_{\text{pSign}_{\langle D_{ID}^{A_1}, D_{ID}^{A_2}\rangle}}(m, z, \pi) \\ (z, Y') \in \text{R} & \sigma := \text{Adapt}(\tilde{\sigma}, Y) \\ & Y' := \text{Ext}(\sigma, \tilde{\sigma}, z) \end{array}\right] = 1. \quad (1)$$

Then we give the definition of the security properties for two-party adaptor signature. A two-party adaptor signature should satisfy unforgeability, similar to the definition of EUF-CMA, this property means that if there exists a malicious party, the party cannot generate a valid signature without help of the other party. Compared to EUF-CMA, the adversary in adaptor signature can additionally access pre-signing oracles to obtain pre-signature and hard relation statement, and it is hard for the adversary to forge a legitimate signature of message m. Here we give the definition of 2-aEUF-CMA security, where "2" represents "two-party" and "a" represents "adaptor".

Definition 3 (2-aEUF-CMA Security): A two-party adaptor signature scheme $aSIG_2$ satisfies 2-aEUF-CMA security if for any $\mathcal{PPT}$ adversary $\mathcal{A}$, the probability of winning the $\text{aSigForge}^b_{\mathcal{A},\text{aSIG}_2}$ experiment is negligible, means that $Pr[\text{aSigForge}^b_{\mathcal{A},\text{aSIG}_2}(\lambda) = 1] < \varepsilon(\lambda)$. The $b \in \{0,1\}$ marks which party is being corrupted by the adversary, and the detail of experiment $\text{aSigForge}^b_{\mathcal{A},\text{aSIG}_2}$ is defined as follows:

1) $\mathcal{Q} := \emptyset$: The challenger $\mathcal{C}$ creates an empty message query list $\mathcal{Q}_m$.
2) $PP \leftarrow \text{Setup}(1^\lambda)$: The challenger $\mathcal{C}$ executes $\text{Setup}(1^\lambda)$ phase to output public parameters PP.
3) $(D_{ID}^{A_{1-b}}, D_{ID}^{A'_b}) \leftarrow \text{Gen}(PP)$: The challenger $\mathcal{C}$ obtains a key pair $(D_{ID}^{A_{1-b}}, D_{ID}^{A'_b})$ by executing $\text{Gen}(PP)$, then uses the $D_{ID}^{A_{1-b}}$ to simulate the honest party $\mathcal{P}_{1-b}$
4) $(D_{ID}^{A_b}) \leftarrow \mathcal{A}(PP)$: The challenger $\mathcal{C}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D_{ID}^{A_b}$, therefore simulating the malicious Party $\mathcal{P}_b$.
5) $(z, Y, \pi) \leftarrow \text{GenR}(1^\lambda)$: The challenger $\mathcal{C}$ creates a hard relation statement/witness pair $(z, Y) \in R$, then it generates a zero-knowledge proof π for the relation.
6) $m^* \leftarrow \mathcal{A}^{\mathcal{O}^b_{\Pi_S}, \mathcal{O}^b_{\Pi_{pS}}}(h_{ID}, D_{ID}^{A_b})$: Adversary $\mathcal{A}$ has the ability to access signing and pre-signing oracles $\mathcal{O}^b_{\Pi_S}$ and $\mathcal{O}^b_{\Pi_{pS}}$, where it can obtain the corresponding signature and pre-signature. Then $\mathcal{A}$ selects a message m^* that is not in the list $\mathcal{Q}_m$.

7) $\tilde{\sigma} \leftarrow \Pi^{\mathcal{A}}_{\mathrm{pSign}\langle D_{ID}^{A_{1-b}},\cdot\rangle.}(m^*, z, \pi)$: Adversary $\mathcal{A}$ obtains the pre-signature $\tilde{\sigma}$ of message m^*.
8) $\sigma^* \leftarrow \mathcal{A}^{\mathcal{O}^b_{\Pi_S},\mathcal{O}^b_{\Pi_{pS}}}(\tilde{\sigma}, z)$: With ability to call signing oracle and pre-signing oracle, adversary $\mathcal{A}$ outputs a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z.
9) Outputs $\{0, 1\}$: If the forged signature σ^* satisfies $\mathrm{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q}_m$, then adversary $\mathcal{A}$ wins the experiment, returns 1;otherwise, returns 0.

For an adaptor signature, there has a requirement that any valid pre-signature (even from a malicious adversary) can be converted to a completed one, this property is defined next.

Definition 4 (Two-Party Pre-Signature Adaptability): A two-party adaptor signature scheme $aSIG_2$ satisfies two-party pre-signature adaptability if for any statement/witness pair $(z, Y) \in R$ and message $m \in \{0, 1\}^*$, the pre-signature $\tilde{\sigma}$ meets $\mathrm{pVerify}_{h_{ID}}(m, z, \tilde{\sigma}) = 1$ and we have $Pr[\mathrm{Verify}_{h_{ID}}(m, \mathrm{Adapt}(\tilde{\sigma}, Y))] = 1$.

Below is the definition of *witness extractability*, which means a $\mathcal{PPT}$ adversary can extract the witness Y for the message/statement (m, z) with a legitimate signature/pre-signature pair $(\sigma, \tilde{\sigma})$.

Definition 5 (Two-Party Witness Extractability): A two-party adaptor signature scheme $aSIG_2$ satisfies Witness Extractability if for any $\mathcal{PPT}$ adversary $\mathcal{A}$, the probability of winning the $\mathrm{aWitExt}_{\mathcal{A},\mathrm{aSIG}_2}$ experiment is negligible, means that $Pr[\mathrm{aWitExt}_{\mathcal{A},\mathrm{aSIG}_2}(\lambda) = 1] < \varepsilon(\lambda)$, the experiment $\mathrm{aWitExt}_{\mathcal{A},\mathrm{aSIG}_2}$ is defined as follows:

1) $\mathcal{Q} := \emptyset$: The challenger $\mathcal{C}$ executes $\mathrm{Setup}(1^\lambda)$ phase to generate public parameters PP.
2) $PP \leftarrow \mathrm{Setup}(1^\lambda)$: The challenger $\mathcal{C}$ executes $\mathrm{Setup}(1^\lambda)$ phase to output public parameters PP.
3) $(D_{ID}^{A_{1-b}}, D_{ID}^{A'_b}) \leftarrow \mathrm{Gen}(PP)$: The challenger $\mathcal{C}$ obtains a key pair $(D_{ID}^{A_{1-b}}, D_{ID}^{A'_b})$ by executing $\mathrm{Gen}(PP)$, then uses the $D_{ID}^{A_{1-b}}$ to simulate the honest party $\mathcal{P}_{1-b}$
4) $(D_{ID}^{A_b}) \leftarrow \mathcal{A}(PP)$: The challenger $\mathcal{C}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D_{ID}^{A_b}$, therefore simulating the malicious Party $\mathcal{P}_b$.
5) $(m^*, z^*, \pi) \leftarrow \mathcal{A}^{\mathcal{O}^b_{\Pi_S},\mathcal{O}^b_{\Pi_{pS}}}(h_{ID}, D_{ID}^{A_b})$: Adversary $\mathcal{A}$ has the ability to access signing and pre-signing oracles $\mathcal{O}^b_{\Pi_S}$ and $\mathcal{O}^b_{\Pi_{pS}}$, where it can obtain the corresponding signature and pre-signature. Then $\mathcal{A}$ selects a message m^* that is not in the list $\mathcal{Q}$ and generate a proof π for the statement z^*.
6) $\tilde{\sigma} \leftarrow \Pi^{\mathcal{A}}_{\mathrm{pSign}\langle D_{ID}^{A_{1-b}},\cdot\rangle.}(m^*, z^*)$: Adversary $\mathcal{A}$ obtains the pre-signature $\tilde{\sigma}$ of message m^*.
7) $\sigma^* \leftarrow \mathcal{A}^{\mathcal{O}^b_{\Pi_S},\mathcal{O}^b_{\Pi_{pS}}}(\tilde{\sigma}, z^*, \pi)$: With ability to call signing oracle and pre-signing oracle, adversary $\mathcal{A}$ outputs a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z.
8) $Y' := \mathrm{Ext}(\sigma^*, \tilde{\sigma}, z^*)$: The challenger $\mathcal{C}$ extracts witness Y' with $\sigma^*, \tilde{\sigma}$ and z^*.
9) Outputs $\{0, 1\}$: If the signature σ^* and witness Y' satisfies $\mathrm{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q} \wedge (z^*, Y') \notin R$, then adversary $\mathcal{A}$ wins the experiment, returns 1;otherwise, returns 0.

Definition 6 (Security of Two-Party Adaptor Signature): If a two-party adaptor signature scheme $aSIG_2$ satisfies 2-aEUF-CMA security, two-party pre-signature adaptability and two-party witness extractability, we say it is a secure two-party adaptor signature scheme.

V. PROPOSED TWO-PARTY ADAPTOR SIGNATURE PROTOCOL

In this section, we construct a two-party identity-based adaptor signature scheme for the IEEE P1363 standard. Our work is based on the two-party P1363 signature scheme given in [30]. The details of our proposed protocol (as shown in Fig. 1) are given below.

- Setup: Given a security parameter λ, this algorithm generates public parameters PP as follows:
  a) Generates the cyclic groups (G_1, G_2, G_T) with the same order q, and a bilinear map $e : G_1 \times G_2 \rightarrow G_T$.
  b) Chooses two random generators $Q_1 \in G_1$ and $Q_2 \in G_2$.
  c) Selects a random secret value s in $\mathbb{Z}_q^*$ as the secret key of server, then computes $P_{pub} = s \cdot Q_2$ and sets $g = e(Q_1, Q_2)$.
  d) Sets $PP = \{P_{pub}, g, Q_1, Q_2, G_1, G_2, G_T, e\}$.
- Gen: Given a user's ID and KGC's secret key s, this algorithm generates the user's private keys $D_{ID}^{A_1}$ and $D_{ID}^{A_2}$ as follows:
  a) KGC generates a random number d_1 in $\mathbb{Z}_q^*$, computes $D_{ID}^{A_1} = d_1 \cdot Q_1$.
  b) KGC computes $d_2 = \frac{s}{s+H_1(ID)} d_1^{-1} (mod\ q)$ and $g_1 = g^{d_1^{-1}}$, then $D_{ID}^{A_2} = (d_2, g_1)$.

  Note that the two private keys $D_{ID}^{A_1}$ and $D_{ID}^{A_2}$ will be stored on two devices A_1 and A_2 respectively. The user's private key $D_{ID} = d_2 D_{ID}^{A_1}$.
- GenR: Given a security parameter λ, this algorithm generates a hard relation statement/witness pair (z, Y) and NIZK proof π as follows:
  a) Select a random value $Y \in G_1$, then calculate $z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})$, where $H_1(ID_A)$ is the identity h_{ID} of the user.
  b) Set the hard relation $I_z := \{(z, Y) : z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})\}$.
  c) Generate a zero-knowledge proof $\pi = Prove(z, Y)$.
  d) Output the hard relation statement/witness pair z, Y and the proof π.
- pSign: Given the user's two private key $D_{ID}^{A_1}$, $D_{ID}^{A_2}$, a message $m \in \{0, 1\}^*$ and a hard relation state/proof pair (z, π), this algorithm generates a pre-signature $\tilde{\sigma}$ as follows:

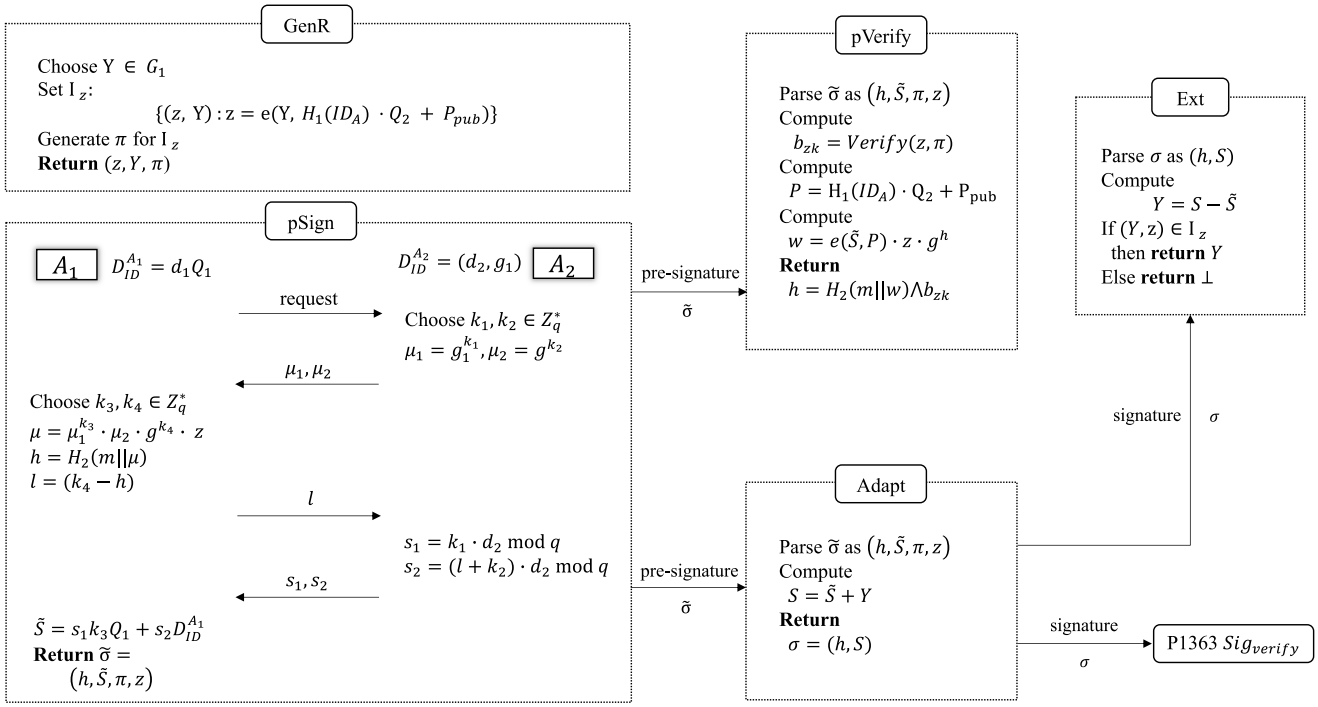

FIGURE 1. The design of two-party P1363-based adaptor signature protocol.

1. $A_1 \rightarrow A_2 : \{request\}$
   A_1 sends a signature request to A_2.
2. $A_2 \rightarrow A_1 : \{\mu_1, \mu_2\}$
   i) While receiving the signature request from A_1, A_2 selects two random numbers $k_1, k_2 \in \mathbb{Z}_q^*$.
   ii) A_2 calculates $\mu_1 = g_1^{k_1}$, $\mu_2 = g^{k_2}$.
   iii) A_2 sends $\{\mu_1, \mu_2\}$ to A_1.
3. $A_1 \rightarrow A_2 : \{l\}$
   i) While receiving the message from A_2, A_1 selects two random numbers $k_3, k_4 \in \mathbb{Z}_q^*$.
   ii) A_1 calculates $\mu = \mu_1^{k_3} \cdot \mu_2 \cdot g^{k_4} \cdot z$, $h = H_2(m||\mu)$ and $l = (k_4 - h)$.
   iii) A_1 sends $\{l\}$ to A_2.
4. $A_2 \rightarrow A_1 : \{s_1, s_2\}$
   i) While receiving the message from A_1, A_2 calculates $s_1 = k_1 \cdot d_2(mod\ q)$ and $s_2 = (l + k_2) \cdot d_2(mod\ q)$.
   ii) A_2 sends $\{s_1, s_2\}$ to A_1.
5. A_1 outputs the pre-signature $\tilde{\sigma}$
   i) While receiving the message from A_2, A_1 calculates $\tilde{S} = s_1k_3Q_1 + s_2D_{ID}^{A_1}$.
   ii) A_1 outputs the pre-signature $\tilde{\sigma} = (h, \tilde{S}, \pi, z)$.

- pVerify: Given a user's public key h_{ID}, a pre-signature $\tilde{\sigma}$, a message m, and a statement z, this algorithm returns 1 if the pre-signature is valid; otherwise, it returns 0. The pre-signature is verified as follows:
  a) Parse $\tilde{\sigma}$ as $(h, \tilde{S}, \pi, z)$.
  b) Compute $b_{zk} = Verify(z, \pi)$.
  c) Compute $P = h_{ID} \cdot Q_2 + P_{pub}$.
  d) Compute $w = e(\tilde{S}, P) \cdot z \cdot g^h$.
  e) If $(h = H_2(m||w)) \wedge b_{zk} = 1)$, then return 1;otherwise, return 0.
- Adapt: Given a pre-signature $\tilde{\sigma}$ and the witness Y, this algorithm output the completed signature as follows:
  a) Parse $\tilde{\sigma}$ as $(h, \tilde{S}, \pi, z)$.
  b) Compute $S = \tilde{S} + Y$.
  c) Return completed signature $\sigma = (h, S)$.
- Ext: Given a pre-signature $\tilde{\sigma}$, a signature σ and a relation statement z, this algorithm computes the witness Y as follows:
  a) Compute $Y = S - \tilde{S}$.
  b) Check whether $(z, Y) \in I_z$.
  c) If yes, return Y; otherwise, return $\perp$.

VI. SECURITY ANALYSIS

Here we analyze the security properties of our proposed protocol, and prove that it achieves the security goals of two-party adaptor signature.

Lemma 1: The two-party P1363 identity-based adaptor signature scheme $aSIG_2^{P1363}$ achieves two-party pre-signature adaptability.

Proof: For any message $m \in \{0, 1\}$, hard relation $I_z : (z, Y)$, we know that $z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})$ and $S = \tilde{S} + Y$, if $\text{pVerify}_{\text{h}_{\text{ID}}}(m, z, \tilde{\sigma}) = 1$, then we have:

$$\begin{aligned} h &= H_2(m, w) \\ &= H_2\left(m, \tilde{e}\left(\tilde{S}, h_{ID} \cdot Q_2 + P_{pub}\right) \cdot z \cdot g^h\right) \end{aligned}$$

$$= H_2\Big(m, \tilde{e}\Big(\tilde{S} + Y, h_{ID} \cdot Q_2 + P_{pub}\Big) \cdot g^h\Big)$$
$$= H_2\Big(m, \tilde{e}\big(S, h_{ID} \cdot Q_2 + P_{pub}\big) \cdot g^h\Big)$$

Obviously, the signature σ converted from pre-signature $\tilde{\sigma}$ is a valid P1363 signature. ■

Lemma 2: The two-party P1363 identity-based adaptor signature scheme $aSIG_2^{P1363}$ achieves pre-signature correctness.

Proof: For an arbitrary user ID, we know that $D_{ID} = d_2 D_{ID}^{A_1} = (s + h_{ID})^{-1} \cdot Q_1$, $h_{ID} = H_1(ID)$ and $z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})$. For the pre-signature $\tilde{\sigma} = (h, \tilde{S})$ generated by $\text{pSign}_{\langle D_{ID}^{A_1}, D_{ID}^{A_2}\rangle}(m, z, \pi)$, we need to check $h = H_2(m, \mu)$, then we have:

$$\begin{aligned}
h &= H_2(m, w) \\
&= H_2\Big(m, e\Big(\tilde{S}, P\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, e\Big(\tilde{S}, h_{ID} \cdot Q_2 + P_{pub}\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, \tilde{e}\Big(k_3 \cdot s_1 Q_1 + s_2 D_{ID}^{A_1}, h_{ID} \cdot Q_2 \\
&\quad + s \cdot Q_2\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, \tilde{e}\Big(k_1 \cdot k_3 \cdot d_2 Q_1 + (k_4 - h + k_2) \cdot d_2 D_{ID}^{A_1}, \\
&\quad (s + h_{ID}) \cdot Q_2\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, \tilde{e}\Big(k_1 \cdot k_3 \cdot d_2 \cdot d_1^{-1} D_{ID} + (k_4 - h + k_2) D_{ID}, \\
&\quad (s + h_{ID}) \cdot Q_2\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, \tilde{e}\Big(\Big(k_1 \cdot k_3 \cdot d_1^{-1} + k_2 + k_4 - h\Big) D_{ID}, \\
&\quad (s + h_{ID}) \cdot Q_2\Big) \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, \tilde{e}\Big(\Big(k_1 \cdot k_3 \cdot d_1^{-1} + k_2 + k_4 - h\Big) Q_1, Q_2\Big) \\
&\quad \cdot z \cdot g^h\Big) \\
&= H_2\Big(m, g^{k_1 \cdot k_3 \cdot d_1^{-1} + k_2 + k_4 - h} \cdot g^h \cdot z\Big) \\
&= H_2(m, \mu)
\end{aligned}$$

After checking the correctness of NIZK, we have $\text{pVerify}_{\text{hID}}(m, z, \tilde{\sigma}) = 1$. As the Lemma 1 is correct, the check $\text{Verify}_{h_{ID}}^{P1363}(m; \sigma) = 1$ achieves for the signature σ that $\text{Adapt}(\tilde{\sigma}, Y)$ outputs. At last, we know that $Y = S - \tilde{S}$, so the return value of algorithm Ext is the witness of I_z, which completes the proof. ■

Lemma 3: Suppose that IEEE P1363 identity-based signature scheme is SUF-CMA secure and R is a hard relation, then our two-party P1363 identity-based adaptor signature scheme $aSIG_2^{P1363}$ satisfies 2-aEUF-CMA security.

Proof: Let $\mathcal{A}$ be the adversary who plays the $\text{aSigForge}_{\mathcal{A},\text{aSIG}_2}^b$ game with simulator $\mathcal{S}$. We assume $\mathcal{S}$ can access the signing oracle $\mathcal{O}_{\Pi_S}(\cdot)$, and $\mathcal{S}$ has control of hash oracle $\mathcal{H}$ and pre-signing oracle $\mathcal{O}_{\Pi_{pS}}$. Then $\mathcal{A}$ makes signing oracle queries($\mathcal{O}_{\Pi_S}$), pre-signing oracle queries($\mathcal{O}_{\Pi_{pS}}$) and hash queries($\mathcal{H}$) to $\mathcal{S}$. Then we give the proof by considering of two cases: adversary corrupts device A_1 or device A_2. ■

Adversary corrupts A_1 ($b = 0$): As the definition of $\text{aSigForge}_{\mathcal{A},\text{aSIG}_2}^0$ experiment, adversary $\mathcal{A}$ interacts with simulator $\mathcal{S}$ as follows:

1) The simulator $\mathcal{S}$ creates an empty message query list $\mathcal{Q}_m$ and an empty hash query list $\mathcal{Q}_H$.
2) The simulator $\mathcal{S}$ executes $\text{Setup}(1^\lambda)$ phase to output public parameters PP.
3) The simulator $\mathcal{S}$ obtains a key pair $(D_{ID}^{A_2}, D_{ID}^{A_1'})$ by executing $\text{Gen}(PP)$, and uses the $D_{ID}^{A_2}$ to simulate the honest device A_2.
4) The simulator $\mathcal{S}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D_{ID}^{A_1}$. Then $\mathcal{A}$ simulates the malicious device A_1.
5) The simulator $\mathcal{S}$ creates a hard relation statement/witness pair $(z, Y) \in R$, and computes a NIZK proof π for $I_z := \{(z, Y) : z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})\}$, then sends z, π to adversary $\mathcal{A}$.
6) The adversary $\mathcal{A}$ makes signing oracle queries($\mathcal{O}_{\Pi_S}$), pre-signing oracle queries($\mathcal{O}_{\Pi_{pS}}$) and hash queries($\mathcal{H}$) to $\mathcal{S}$. As for the message $m_i \in \{0, 1\}^*$ chosen by $\mathcal{A}$, $\mathcal{S}$ responses the queries as follows:
   Hash query $\mathcal{H}(x)$: While querying the hash value of x, firstly $\mathcal{S}$ queries list $\mathcal{Q}_H$ to see if $H(x)$ exists; if exists, then return it; otherwise, select a random $H(x)$ in $\mathbb{Z}_q^*$, then stroes $(x, H(x))$ in list $\mathcal{Q}_H$ and return $H(x)$.
   Sign query $\mathcal{O}_{\Pi_S}^0(m_i)$: While querying the signature value of m_i, $\mathcal{S}$ make a signing query to oracle $\mathcal{O}_{\Pi_S}$ to get signature σ, then return σ.
   The signature is distributively generated in the way given in [30]. As the device A_1 is corrupted by $\mathcal{A}$, $\mathcal{S}$ needs to simulate the operations that $\mathcal{A}$ executes during the distributed signature generation with ability to invoke adversary $\mathcal{A}$. On receiving the query of message m_i, the sign oracle $\mathcal{O}_{\Pi_S}^0$ works as follows:
   a) For the query of message m_i, $\mathcal{S}$ calls original P1363 $\text{Sign}(m_i)$ to get signature (h, S). Then $\mathcal{A}$ interacts with $\mathcal{S}$ as follows.
      Case 1:
      i) For the first message sent by $\mathcal{S}$, if $\mu_1 = g_1^{k_1} = g^{k_1 d_1^{-1}}$ and $\mu_2 = g_{k_2}$, $\mathcal{A}$ computes $\mu = \mu_1^{k_3} \cdot \mu_2 \cdot g^{k_4}$ and generates h', then $\mathcal{A}$ replies $\mathcal{S}$ with (μ, k_3).
      ii) While getting the message from $\mathcal{A}$, $\mathcal{S}$ sets $s_1 = k_3^{-1} \cdot S$ and replies (s_1, k_3^{-1}) to $\mathcal{A}$.
      Case 2:
      i) The steps described in A_1 are not executed by $\mathcal{A}$ and replaced by the following computation: $\mu = g^x, h = H_2(m_i, \mu)$. Then (μ, x) is sent to $\mathcal{S}$.
      ii) While getting the message from $\mathcal{A}$, $\mathcal{S}$ selects two random numbers $s_1, s_2 \in \mathbb{Z}_q^*$ and sends (s_1, s_2) to $\mathcal{A}$.

b) $\mathcal{A}$ finally computes signature σ^*, then the simulation will be terminated by $\mathcal{S}$ and outputs signature σ^*.

Pre-sign query $\mathcal{O}^0_{\Pi_{pS}}(m_i, z, \pi)$*:* While querying the pre-signature of m_i and z, $\mathcal{S}$ responses as follows:

a) Queries to $\mathcal{O}^0_{\Pi_S}(m_i)$ and gets the signature σ.
b) Computes $\tilde{S} = S - Y$.
c) Updates the list $\mathcal{Q}_m \coloneqq \mathcal{Q}_m \cup \{m_i\}$.
d) Outputs the pre-signature $\tilde{\sigma} = (h, \tilde{S}, \pi, z)$.

7) $\mathcal{A}$ chooses a challenge message $m^* \in \{0,1\}^*$ and makes a query to $\mathcal{S}$ about pre-signature of (m^*, z, π), $\mathcal{S}$ responses with $\tilde{\sigma} = \mathcal{O}^0_{\Pi_{pS}}(m^*, z, \pi)$.
8) $\mathcal{A}$ generates a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z, then outputs it.
9) On receiving the σ^*, $\mathcal{S}$ checks whether the signature σ^* satisfies $\text{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q}_m$. If it is valid, returns 1; otherwise, returns 0.

While σ^* is legitimate and the message m^* is not in list $\mathcal{Q}_m$, $\mathcal{S}$ can use the message-signature pair (m^*, σ^*) to attack the strong unforgeability of the P1363 signature scheme.

Advantage Analysis: While simulator returns 1, adversary can get witness Y by running $\text{Ext}(\sigma, \tilde{\sigma}, z)$, but the probability is negligible. The advantage of adversary $\mathcal{A}$ is $Adv^{\mathcal{A}}_{aSigForge} = Adv^{\mathcal{A}}_{P1363-SUF-CMA} + \varepsilon(\lambda)$, which is also negligible.

Adversary corrupts A_2 $(b = 1)$*:* In this case most steps are the same as the case of $b = 0$, but there is something different. During the simulation of sign query $\mathcal{O}^1_{\Pi_S}(m_i)$, the signature is finally generated by $\mathcal{S}$, so adversary $\mathcal{A}$ cannot forge a valid signature without knowing the private key of $\mathcal{S}$. Therefore we let $\mathcal{S}$ abort the simulation for some random points. $\mathcal{S}$, the details are given in the description of $\mathcal{O}^1_{\Pi_S}(m_i)$.

As the definition of $\text{aSigForge}^1_{\mathcal{A},\text{aSIG}_2}$ experiment, adversary $\mathcal{A}$ interacts with simulator $\mathcal{S}$ as follows:

1) The simulator $\mathcal{S}$ creates an empty message query list $\mathcal{Q}_m$ and an empty hash query list $\mathcal{Q}_H$.
2) The simulator $\mathcal{S}$ executes $\text{Setup}(1^\lambda)$ phase to output public parameters PP.
3) The simulator $\mathcal{S}$ obtains a key pair $(D^{A_1}_{ID}, D^{A'_2}_{ID})$ by executing $\text{Gen}(PP)$, and uses the $D^{A_1}_{ID}$ to simulate the honest device A_1.
4) The simulator $\mathcal{S}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D^{A_1}_{ID}$. Then $\mathcal{A}$ simulates the malicious device A_2.
5) The simulator $\mathcal{S}$ creates a hard relation statement/witness pair $(z, Y) \in R$, and computes a NIZK proof π for $I_z \coloneqq \{(z, Y) : z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})\}$, then sends z, π to adversary $\mathcal{A}$.
6) The adversary $\mathcal{A}$ makes signing oracle queries$(\mathcal{O}_{\Pi_S})$, pre-signing oracle queries$(\mathcal{O}_{\Pi_{pS}})$ and hash queries$(\mathcal{H})$ to $\mathcal{S}$. As for the message $m_i \in \{0,1\}^*$ chosen by $\mathcal{A}$, $\mathcal{S}$ responses the queries as follows:
Hash query $\mathcal{H}(x)$*:* While querying the hash value of x, firstly $\mathcal{S}$ queries list $\mathcal{Q}_H$ to see if $H(x)$ exists; if exists, then return it; otherwise, select a random $H(x)$ in $\mathbb{Z}^*_q$, then stroes $(x, H(x))$ in list $\mathcal{Q}_H$ and return $H(x)$.
Sign query $\mathcal{O}^1_{\Pi_S}(m_i)$*:* While querying the signature value of m_i, $\mathcal{S}$ make a signing query to oracle $\mathcal{O}_{\Pi_S}$ to get signature σ, then return σ.
As the device A_2 is corrupted by $\mathcal{A}$, $\mathcal{S}$ needs to simulate the operations that $\mathcal{A}$ executes during the signature generation phase with ability to invoke adversary $\mathcal{A}$. In the simulation of sign oracle $\mathcal{O}^1_{\Pi_S}$, $\mathcal{S}$ selects a random number $i \in \{1, \ldots, t(n)+1\}$, where $t(\cdot)$ represents how many times that adversary $\mathcal{A}$ runs this protocol. If i is chosen correctly, then S can perform a successful simulation of $\mathcal{O}^1_{\Pi_S}$. On receiving the query of message m_i, the sign oracle $\mathcal{O}^1_{\Pi_S}$ works as follows:

a) For the query of message m_i, $\mathcal{S}$ calls original P1363 $\text{Sign}(m_i)$ to get signature (h, S). Then $\mathcal{A}$ interacts with $\mathcal{S}$ as below.
   i) For the first message sent by $\mathcal{A}$, if $\mu_1 = g_1^{k_1} = g^{k_1 d_1^{-1}}$ and $\mu_2 = g_{k_2}$, then $\mathcal{S}$ computes $\mu = g^{k_1 d_1^{-1} k_3 + k_2}$ and generates h', then $\mathcal{S}$ sends h to $\mathcal{A}$. Otherwise, the simulation will be terminated by $\mathcal{S}$.
   ii) For another query of message m_2, if it is the $i - th$ time that $\mathcal{A}$ runs the protocol, then $\mathcal{S}$ terminates and returns $\perp$. Otherwise, the simulation continues.
b) $\mathcal{S}$ finally computes signature σ^*. Then the simulation will be terminated by $\mathcal{S}$ and outputs signature σ^*.

While $i \in \{1, \ldots, t(n) + 1\}$, adversary $\mathcal{A}$ will obtain $t(n)$ correct signatures. Therefore we get the following equation:

$$\begin{aligned}\mu^* &= \frac{e\left(S^*, H_1(ID)Q_2 + P_{pub}\right)}{g^h} \\ &= \frac{e(S^*, (H_1(ID) + s)Q_2)}{g^h} \\ &= \frac{e\left(\left(k_1 k_3 d_1^{-1} + k_2 + h\right) d_1 d_2 Q_1, (H_1(ID) + s)Q_2\right)}{g^h} \\ &= \frac{g^{\left(k_1 k_3 d_1^{-1} + k_2 + h\right)}}{g^h} = g^{k_1 k_3 d_1^{-1} + k_2}\end{aligned}$$

Pre-sign query $\mathcal{O}^1_{\Pi_{pS}}(m_i, z, \pi)$*:* While querying the pre-signature of m_i and z, $\mathcal{S}$ responses as follows:

a) Queries to $\mathcal{O}^1_{\Pi_S}(m_i)$ and gets the signature σ.
b) Computes $\tilde{S} = S - Y$.
c) Updates the list $\mathcal{Q}_m \coloneqq \mathcal{Q}_m \cup \{m_i\}$.
d) Outputs the pre-signature $\tilde{\sigma} = (h, \tilde{S}, \pi, z)$.

7) $\mathcal{A}$ chooses a challenge message $m^* \in \{0,1\}^*$ and makes a query to $\mathcal{S}$ about pre-signature of (m^*, z, π), $\mathcal{S}$ responses with $\tilde{\sigma} = \mathcal{O}^1_{\Pi_{pS}}(m^*, z, \pi)$.
8) $\mathcal{A}$ generates a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z, then outputs it.

9) On receiving the σ^*, $\mathcal{S}$ checks whether the signature σ^* satisfies $\text{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q}_m$. If it is valid, returns 1; otherwise, returns 0.

While σ^* is legitimate and the message m^* is not in list $\mathcal{Q}_m$, $\mathcal{S}$ can use the message-signature pair (m^*, σ^*) to attack the strong unforgeability of the P1363 signature scheme.

Advantage Analysis: While simulator returns 1, adversary can get witness Y by running $\text{Ext}(\sigma, \tilde{\sigma}, z)$, but the probability is negligible. Additionally, the simulator only terminates in the sign oracle $\mathcal{O}^1_{\Pi_S}$, while the probability is also negligible. Thus we get the advantage of adversary $\mathcal{A}$: $Adv^{\mathcal{A}}_{aSigForge} = Adv^{\mathcal{A}}_{P1363-SUF-CMA} + \varepsilon(\lambda)$, which is negligible.

Lemma 4: Suppose that IEEE P1363 identity-based signature scheme is SUF-CMA secure and R is a hard relation, then our two-party P1363 identity-based adaptor signature scheme $aSIG_2^{P1363}$ satisfies two-Party Witness Extractability.

Proof: Let $\mathcal{A}$ be the adversary who plays the $\text{aWitExt}_{\mathcal{A},\text{aSIG}_2}$ game with simulator $\mathcal{S}$. The proof we give here is in common with the previous proof for Lemma 3, but the hard relation in $\text{aSigForge}^b_{\mathcal{A},\text{aSIG}_2}$ is generated by simulator $\mathcal{S}$, while in the $\text{aWitExt}_{\mathcal{A},\text{aSIG}_2}$, the witness Y is only known to $\mathcal{A}$.

We assume $\mathcal{S}$ can access the signing oracle $\mathcal{O}_{\Pi_S}(\cdot)$, and $\mathcal{S}$ has control of hash oracle $\mathcal{H}$ and pre-signing oracle $\mathcal{O}_{\Pi_{pS}}$. Then $\mathcal{A}$ makes signing oracle queries($\mathcal{O}_{\Pi_S}$), pre-signing oracle queries($\mathcal{O}_{\Pi_{pS}}$) and hash queries($\mathcal{H}$) to $\mathcal{S}$. Then we give the proof by considering of two cases: adversary corrupts device A_1 or device A_2.

Adversary corrupts A_1 ($b = 0$): As the definition of $\text{aSigForge}^0_{\mathcal{A},\text{aSIG}_2}$ experiment, adversary $\mathcal{A}$ interacts with simulator $\mathcal{S}$ as follows:

1) The simulator $\mathcal{S}$ creates an empty message query list $\mathcal{Q}_m$ and an empty hash query list $\mathcal{Q}_H$.
2) The simulator $\mathcal{S}$ executes $\text{Setup}(1^\lambda)$ phase to output public parameters PP.
3) The simulator $\mathcal{S}$ obtains a key pair $(D^{A_2}_{ID}, D^{A'_1}_{ID})$ by executing $\text{Gen}(PP)$, and uses the $D^{A_2}_{ID}$ to simulate the honest device A_2.
4) The simulator $\mathcal{S}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D^{A_1}_{ID}$. Then $\mathcal{A}$ simulates the malicious device A_1. Additionally, $\mathcal{A}$ generates a hard relation statement/witness pair $(z, Y) \in R$, and computes a NIZK proof π for $I_z := \{(z, Y) : z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})\}$.
5) The adversary $\mathcal{A}$ makes signing oracle queries($\mathcal{O}_{\Pi_S}$), pre-signing oracle queries($\mathcal{O}_{\Pi_{pS}}$) and hash queries($\mathcal{H}$) to $\mathcal{S}$. As for the message $m_i \in \{0,1\}^*$ chosen by $\mathcal{A}$, $\mathcal{S}$ responses the queries as follows:
   Hash query $\mathcal{H}(x)$: While querying the hash value of x, firstly $\mathcal{S}$ queries list $\mathcal{Q}_H$ to see if $H(x)$ exists; if exists, then return it; otherwise, select a random $H(x)$ in $\mathbb{Z}^*_q$, then stroes $(x, H(x))$ in list $\mathcal{Q}_H$ and return $H(x)$.
   Sign query $\mathcal{O}^0_{\Pi_S}(m_i)$: While querying the signature value of m_i, $\mathcal{S}$ make a signing query to oracle $\mathcal{O}_{\Pi_S}$ to get signature σ, then return σ.
   The way that the sign oracle $\mathcal{O}^0_{\Pi_S}$ works is described in the proof of Lemma 3.
   Pre-sign query $\mathcal{O}^0_{\Pi_{pS}}(m_i, z, \pi)$: While querying the pre-signature of m_i and z, $\mathcal{S}$ responses as follows:
   a) Gets the witness Y throuth the witness extractor of the NIZK scheme; if $(z, Y) \notin R$, terminates and return $\perp$.
   b) Queries to $\mathcal{O}^0_{\Pi_S}(m_i)$ and gets the signature σ.
   c) Computes $\tilde{S} = S - Y$.
   d) Updates the list $\mathcal{Q}_m := \mathcal{Q}_m \cup \{m_i\}$.
   e) Outputs the pre-signature $\tilde{\sigma} = (h, \tilde{S}, \pi, z)$.
6) $\mathcal{A}$ chooses a challenge message $m^* \in \{0,1\}^*$ and makes a query to $\mathcal{S}$ about pre-signature of (m^*, z, π), $\mathcal{S}$ responses with $\tilde{\sigma} = \mathcal{O}^1_{\Pi_{pS}}(m^*, z, \pi)$.
7) $\mathcal{A}$ generates a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z, then outputs it.
8) Simulator $\mathcal{S}$ extracts witness Y' throuth $\text{Ext}(\sigma^*, \tilde{\sigma}, z^*)$.
9) On receiving the σ^*, $\mathcal{S}$ checks whether the signature σ^* and witness Y' satisfies $\text{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q} \wedge (z^*, Y') \notin R$. If it is valid, returns 1; otherwise, returns 0.

While σ^* is legitimate and the message m^* is not in list $\mathcal{Q}_m$, $\mathcal{S}$ can use the message-signature pair (m^*, σ^*) to attack the strong unforgeability of the P1363 signature scheme.

Advantage Analysis: As the simulator only terminates during the step a) of pre-sign query, and the probability is negligible. Thus we get the advantage of adversary $\mathcal{A}$ is $Adv^{\mathcal{A}}_{aWitExt} = Adv^{\mathcal{A}}_{P1363-SUF-CMA} + \varepsilon(\lambda)$, which is also negligible.

Adversary corrupts A_2 ($b = 1$): In this case most steps are the same as the case of $b = 0$, and we show the differences in the previous proof of Lemma 3. As the definition of $\text{aWitExt}^1_{\mathcal{A},\text{aSIG}_2}$ experiment, adversary $\mathcal{A}$ interacts with simulator $\mathcal{S}$ as follows:

1) The simulator $\mathcal{S}$ creates an empty message query list $\mathcal{Q}_m$ and an empty hash query list $\mathcal{Q}_H$.
2) The simulator $\mathcal{S}$ executes $\text{Setup}(1^\lambda)$ phase to output public parameters PP.
3) The simulator $\mathcal{S}$ obtains a key pair $(D^{A_1}_{ID}, D^{A'_2}_{ID})$ by executing $\text{Gen}(PP)$, and uses the $D^{A_1}_{ID}$ to simulate the honest device A_1.
4) The simulator $\mathcal{S}$ forwards PP to the adversary $\mathcal{A}$ to make it generates its key $D^{A_1}_{ID}$. Then $\mathcal{A}$ simulates the malicious device A_2. Additionally, $\mathcal{A}$ generates a hard relation statement/witness pair $(z, Y) \in R$, and computes a NIZK proof π for $I_z := \{(z, Y) : z = e(Y, H_1(ID_A) \cdot Q_2 + P_{pub})\}$.
5) The adversary $\mathcal{A}$ makes signing oracle queries($\mathcal{O}_{\Pi_S}$), pre-signing oracle queries($\mathcal{O}_{\Pi_{pS}}$) and hash queries($\mathcal{H}$) to $\mathcal{S}$. As for the message $m_i \in \{0,1\}^*$ chosen by $\mathcal{A}$, $\mathcal{S}$ responses the queries as follows:
   Hash query $\mathcal{H}(x)$: While querying the hash value of x, firstly $\mathcal{S}$ queries list $\mathcal{Q}_H$ to see if $H(x)$ exists; if exists, then return it; otherwise, select a random $H(x)$ in $\mathbb{Z}^*_q$, then stroes $(x, H(x))$ in list $\mathcal{Q}_H$ and return $H(x)$.

Sign query $\mathcal{O}^1_{\Pi_S}(m_i)$*:* While querying the signature value of m_i, $\mathcal{S}$ make a signing query to oracle $\mathcal{O}_{\Pi_S}$ to get signature σ, then return σ.
The way that the sign oracle $\mathcal{O}^1_{\Pi_S}$ works is described in the proof of Lemma 3.
Pre-sign query $\mathcal{O}^1_{\Pi_{pS}}(m_i, z, \pi)$*:* While querying the pre-signature of m_i and z, $\mathcal{S}$ responses as follows:

a) Gets the witness Y throuth the witness extractor of the NIZK scheme; if $(z, Y) \notin R$, terminates and return $\perp$.
b) Queries to $\mathcal{O}^1_{\Pi_S}(m_i)$ and gets the signature σ.
c) computes $\tilde{S} = S - Y$.
d) updates the list $\mathcal{Q}_m := \mathcal{Q}_m \cup \{m_i\}$.
e) outputs the pre-signature $\tilde{\sigma} = (h, \tilde{S}, \pi, z)$.

6) $\mathcal{A}$ chooses a challenge message $m^* \in \{0,1\}^*$ and makes a query to $\mathcal{S}$ about pre-signature of (m^*, z, π), $\mathcal{S}$ responses with $\tilde{\sigma} = \mathcal{O}^1_{\Pi_{pS}}(m^*, z, \pi)$.
7) $\mathcal{A}$ generates a forged signature σ^* under the signature $\tilde{\sigma}$ and relation statement z, then outputs it.
8) Simulator $\mathcal{S}$ extracts witness Y' throuth $\text{Ext}(\sigma^*, \widetilde{\sigma}, z^*)$.
9) On receiving the σ^*, $\mathcal{S}$ checks whether the signature σ^* satisfies $\text{Verify}_{h_{ID}}(m^*, \sigma^*) = 1 \wedge m^* \notin \mathcal{Q}_m$. If it is valid, returns 1; otherwise, returns 0.

While σ^* is legitimate and the message m^* is not in list $\mathcal{Q}_m$, $\mathcal{S}$ can use the message-signature pair (m^*, σ^*) to attack the strong unforgeability of the P1363 signature scheme.

Advantage Analysis: The simulator terminates in two cases: step a) of pre-sign query and step ii) of sign oracle $\mathcal{O}^1_{\Pi_S}$. Both of these two cases have a negligible probability. Thus we get the advantage of adversary $\mathcal{A}$ is $Adv^{\mathcal{A}}_{aWitExt} = Adv^{\mathcal{A}}_{P1363-SUF-CMA} + \varepsilon(\lambda)$, which is also negligible.

Lemma 5: Suppose that two-party IEEE P1363 identity-based adaptor signature scheme satisfies 2-aEUF-CMA security, two-party pre-signature adaptability and two-party witness extractability, then the two-party adaptor signature scheme is secure.

Proof: According to Lemma 1-4 and Definition 6, the two-party P1363 identity-based adaptor signature scheme is secure. ■

VII. EXPERIMENTAL EVALUATION

We implemented our protocol with the MIRACL library, including the core algorithms of the scheme and NIZK proof. The operating environment is a macOS desktop computer with Intel Core i5-1038NG7 16.00GB RAM. Then we analyze the performance of our 2-P1363-based adaptor signature scheme and make a comparison with 2-Schnoor-based, ECDSA-based and SM2-based adaptor signatures, where "2" represents "two-party". We first give a complexity analysis of main algorithm in our protocol, then we make a timing benchmark to show the computation times, finally we analyze the communication cost.

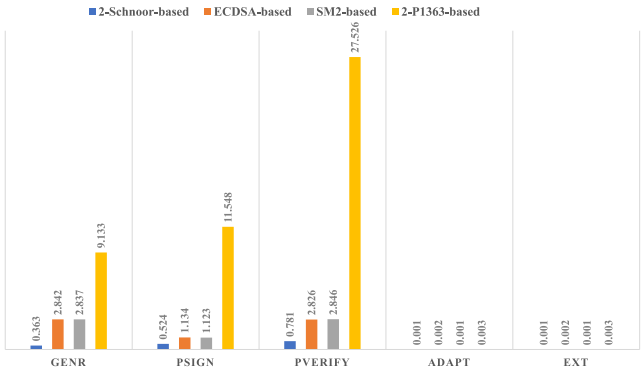

FIGURE 2. Comparison of different schemes in execution time (milliseconds).

A. COMPLEXITY ANALYSIS

We make a complexity analysis of five phases in these four protocols, including GenR, pSign, pVerify, Adapt and Ext. The Setup and Gen phases are the same as the original signature scheme, which can be ignored. Result of the comparison is shown in Table 1. We use M_1, M_2, M_T, M_q to represent the multiplication operation in G_1, G_2, G_T and $\mathbb{Z}^*_q$ respectively, the addition operation in G_1, G_2 and $\mathbb{Z}^*_q$ is denoted as A_1, A_2 and A_q. We also need exponentiation operation in G_T and modular inversion in $\mathbb{Z}^*_q$, which is represented by E_T and M_{inv}. The pairing operation is denoted as P, and the H is a hash operation.

Meanwhile, we use G_π and V_π to represent NIZK proof generation and verification. As for the NIZK proof scheme used in our evaluation, we choose the Σ-protocol for ECDSA-based, SM2-based and our 2-P1363-based adaptor signature, which is efficient and practical for our application.

In the pSign phase of our protocol, the generation of signature is distributed, so we spilt it into 4 steps, step 1 refers to the computation by A_2 after receiving a request from A_1, step 2 refers to the computation by A_1 after receiving first message from A_2, step 3 refers to the computation by A_2 after receiving the message from A_1, and step 4 refers to the computation by A_1 after getting all messages from A_2. Due to the use of bilinear pairs, the complexity of our scheme in GenR phase and pVerify phase is obviously higher. Further more, the distributed pSign phase also requires more computation, especially the exponentiation operation in G_T.

B. TIMING BENCHMARK

Fig. 2 shows the execution time of each procedures, note that we execute each entry for 10000 times and compute the average value. We choose BN Curve over $GF(p)$ (256-bit modulus p, $k = 12$) for implemention.

The result shows the execution time of our protocol in GenR, pSign and pVerify phases are higher, due to the extra expensive pairing and exponentiation operation in G_T and multiplication operation in G_2. The cost of GenR phase is about three times than ECDSA-based and SM2-based schemes. The pVerify phase in our protocol needs almost ten times of the execution time than ECDSA-based and

TABLE 1. Complexity analysis of algorithms.

Algorithm	Computation costs				
	2-P1363-based		2-Schnoor-based	SM2-based	ECDSA-based
GenR	$G_\pi + P + A_2 + M_2 + H$		M_1	$G_\pi + M_1$	$G_\pi + M_1$
pSign	step1	$2E_T$	$A_1 + M_1 + H + M_q$	$A_1 + 2M_1 + H + 2M_q$	$2M_1 + H + M_{inv} + M_q$
	step2	$2E_T + 3M_T + H + A_q$			
	step3	$2M_q + A_q$			
	step4	$2M_1 + A_1 + M_q$			
pVerify	$V_\pi + P + A_2 + M_2 + 2M_T + E_T + H$		$2A_1 + 2M_1 + H$	$V_\pi + A_1 + 2M_1 + H + 2M_q$	$V_\pi + A_1 + 2M_1 + H + M_{inv} + 2M_q$
Adapt	A_1		A_q	A_q	$M_{inv} + M_q$
Ext	A_1		A_q	A_q	$M_{inv} + M_q$

TABLE 2. Communication costs of different schemes.

Schemes	Pre-signature size/bytes
2-Schnoor-based	$2\|\mathbb{Z}_q^*\| = 64$
ECDSA-based	$4\|\mathbb{Z}_q^*\| + \|G_1\| = 192$
SM2-based	$4\|\mathbb{Z}_q^*\| + \|G_1\| = 192$
2-P1363-based	$2\|\mathbb{Z}_q^*\| + \|G_1\| + 3\|G_T\| = 320$

SM2-based schemes. The pSign phase is distributed in our scheme, so the comparison is not important here. Finally, the performance in Adapt and Ext phases are actually the same in all these four schemes, and the execution time can be ignored.

C. COMMUNICATION COST

As for the communication cost, we mainly consider the pre-signature size here, as the key size and completed signature size of these schemes are the same as the original signature schemes. Note that the elements in $\mathbb{Z}_q^*$ is 32 bytes, and the size for elements in G_1 and G_T is 64 bytes.

The pre-signature size of ECDSA-based and SM2-based schemes are $4|\mathbb{Z}_q^*|+|G_1|$, while the NIZK proof size is $2|\mathbb{Z}_q^*|$. As the 2-Schnoor-based scheme does not need a proof, the pre-signature size of it is $2|\mathbb{Z}_q^*|$. The pre-signature in our scheme is $(h, \tilde{S}, \pi, z)$, and $(h, \tilde{S}, z)$ are elements in $\mathbb{Z}_q^*$, G_1 and G_T respectively. Additionally, the NIZK proof size in our protocol is $|\mathbb{Z}_q^*|+2|G_T|$, so the pre-signature size in our protocol is $2|\mathbb{Z}_q^*|+|G_1|+3|G_T|$. We give the comparison result in Table 2. With the additional NIZK proof, our protocol is surely with a larger pre-signature size than 2-schnoor-based protocol. While our protocol is designed for identity-based application, we think the extra costs is acceptable.

VIII. CONCLUSION

In this paper, we propose a secure two-party adaptor signature based on IEEE P1363 standard identity-based signature scheme. Specifically, the pre-signature in our protocol is generated with interactions of two devices, and cannot be computed from only one device (lost or stolen). We formally prove that our protocol satisfies 2-aEUF-CMA security, two-party pre-signature adaptability and two-party witness extractability. The performance evaluation shows that the overhead of our protocol is acceptable. As for future work, we will try to design adaptor blind signature and ring adaptor signature based on our protocol.

DATA AVAILABILITY

The experimental data used to support the findings of this study are available from the corresponding author upon request.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, and Y. Liu, "A survey on the scalability of blockchain systems," *IEEE Netw.*, vol. 33, no. 5, pp. 166–173, Sep./Oct. 2019.

[2] A. Poelstra. "Scriptless scripts." 2017. Accessed: Aug. 10, 2023. [Online]. Available: http://diyhpl.us/wiki/transcripts/layer2-summit/2018/scriptless-scripts/

[3] G. Malavolta, P. Moreno-Sanchez, A. Kate, M. Maffei, and S. Ravi, "Concurrency and privacy with payment-channel networks," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2017, pp. 455–471.

[4] L. Aumayr et al., "Generalized channels from limited blockchain scripts and adaptor signatures," IACR, Bellevue, WA, USA, Rep. 2020/476, 2020. [Online]. Available: https://eprint.iacr.org/2020/476.

[5] C. Decker and R. Wattenhofer, "A fast and scalable payment network with bitcoin duplex micropayment channels," in *Proc. Symp. Self-Stabil. Syst.*, 2015, pp. 3–18.

[6] R. Yu, G. Xue, V. T. Kilari, D. Yang, and J. Tang, "CoinExpress: A fast payment routing mechanism in blockchain-based payment channel networks," in *Proc. 27th Int. Conf. Comput. Commun. Netw. (ICCCN)*, 2018, pp. 1–9.

[7] M. Herlihy, "Atomic cross-chain swaps," in *Proc. ACM Symp. Princ. Distrib. Comput.*, 2018, pp. 245–254.

[8] A. Deshpande and M. Herlihy, "Privacy-preserving cross-chain atomic swaps," in *Proc. Int. Conf. Financ. Cryptogr. Data Security*, 2020, pp. 540–549.

[9] S. A. Thyagarajan, G. Malavolta, and P. Moreno-Sanchez, "Universal atomic swaps: Secure exchange of coins across all blockchains," in *Proc. IEEE Symp. Security Privacy (SP)*, 2022, pp. 1299–1316.

[10] G. Malavolta, P. Moreno-Sanchez, C. Schneidewind, A. Kate, and M. Maffei, "Anonymous multi-hop locks for blockchain scalability and interoperability," IACR, Bellevue, WA, USA, Rep. 2018/472, 2018.

[11] A. Erwig, S. Faust, K. Hostáková, M. Maitra, and S. Riahi, "Two-party adaptor signatures from identification schemes," in *Proc. IACR Int. Conf. Public-Key Cryptogr.*, 2021, pp. 451–480.

[12] P. Sharma, N. R. Moparthi, S. Namasudra, V. Shanmuganathan, and C.-H. Hsu, "Blockchain-based IoT architecture to secure healthcare system using identity-based encryption," *Expert Syst.*, vol. 39, no. 10, 2022, Art. no. e12915.

[13] D. Pavithran, J. N. Al-Karaki, and K. Shaalan, "Edge-based blockchain architecture for event-driven IoT using hierarchical identity based encryption," *Inf. Process. Manage.*, vol. 58, no. 3, 2021, Art. no. 102528.
[14] Z. Wan, W. Liu, and H. Cui, "HIBEChain: A hierarchical identity-based blockchain system for large-scale IoT," *IEEE Trans. Dependable Secure Comput.*, vol. 20, no. 2, pp. 1286–1301, Mar./Apr. 2023.
[15] *IEEE Standard Specifications for Public-Key Cryptography*, IEEE Standard 1363-2000, 2000.
[16] L. Fournier. "One-time verifiably encrypted signatures A.K.A. adaptor signatures." 2020. Accessed: Aug. 10, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:212743751
[17] S. Thyagarajan and G. Malavolta, "Lockable signatures for blockchains: Scriptless scripts for all signatures," in *Proc. IEEE Symp. Security Privacy*, 2021, pp. 937–954.
[18] J. B. Klamti and M. A. Hasan, "Post-quantum two-party adaptor signature based on coding theory," *Cryptography*, vol. 6, no. 1, p. 6, 2022.
[19] X. Qin, H. Cui, and T. H. Yuen, "Generic adaptor signature," IACR, Bellevue, WA, USA, Rep. 2021/161, 2021.
[20] W. Dai, T. Okamoto, and G. Yamamoto, "Stronger security and generic constructions for adaptor signatures," in *Proc. Int. Conf. Cryptol.*, 2022, pp. 52–77.
[21] P. Moreno-Sanchez, A. Blue, D. V. Le, S. Noether, B. Goodell, and A. Kate, "DLSAG: Non-interactive refund transactions for interoperable payment channels in monero," in *Proc. Int. Conf. Financ. Cryptogr. Data Security*, 2020, pp. 325–345.
[22] E. Tairi, P. Moreno-Sanchez, and M. Maffei, "A2L: Anonymous atomic locks for scalability in payment channel hubs," in *Proc. IEEE Symp. Security Privacy (SP)*, 2021, pp. 1834–1851.
[23] M. F. Esgin, O. Ersoy, and Z. Erkin, "Post-quantum adaptor signatures and payment channel networks," in *Proc. Eur. Symp. Res. Comput. Security*, 2020, pp. 378–397.
[24] E. Tairi, P. Moreno-Sanchez, and M. Maffei, "Post-quantum adaptor signature for privacy-preserving off-chain payments," in *Proc. Int. Conf. Financ. Cryptogr. Data Security*, 2021, pp. 131–150.
[25] V. Gilchrist, "An isogeny-based adaptor signature using SQIsign," M.S. thesis, Dept. Combinatorics Optim., Univ. Waterloo, Waterloo, ON, Canada, 2022.
[26] R. Dutta, R. Barua, and P. Sarkar, "Pairing-based cryptographic protocols: A survey," IACR, Bellevue, WA, USA, Rep. 2004/064, 2004.
[27] J. Katz, "Digital signatures: Background and definitions," in *Digital Signatures*. Berlin, Germany: Springer, 2010.
[28] P. S. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Proc. 11th Int. Conf. Theory Appl. Cryptol. Inf. Security*, Chennai, India, Dec. 2005, pp. 515–532.
[29] M. Blum, P. Feldman, and S. Micali, "Non-interactive zero-knowledge and its applications," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*. New York, NY, USA: ACM, 2019, pp. 329–349.
[30] D. He, Y. Zhang, D. Wang, and K.-K. R. Choo, "Secure and efficient two-party signing protocol for the identity-based signature scheme in the IEEE P1363 standard for public key cryptography," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 5, pp. 1124–1132, Sep./Oct. 2020.

XINJIE ZHU received the bachelor's degree from the School of Cyber Engineering, Xidian University, Xi'an, China, in 2020. He is currently pursuing the master's degree with the Key Laboratory of Aerospace Information Security and Trusted Computing Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China. His research interests include cryptographic protocols and blockchain.

DEBIAO HE (Member, IEEE) received the Ph.D. degree in applied mathematics from the School of Mathematics and Statistics, Wuhan University, Wuhan, China, in 2009, where he is currently a Professor with the School of Cyber Science and Engineering. He has authored or coauthored more than 100 research papers in refereed international journals and conferences, such as IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and Usenix Security Symposium. His main research interests include cryptography and information security, in particular, cryptographic protocols. He is on the editorial board of several international journals, such as *ACM Distributed Ledger Technologies: Research & Practice*, *Frontiers of Computer Science*, and IEEE TRANSACTIONS ON COMPUTERS.

ZIJIAN BAO received the M.S. degree in computer application technology from the School of Computer Science and Engineering, Northeastern University, Shenyang, China, in 2019. He is currently pursuing the Ph.D. degree with the Key Laboratory of Aerospace Information Security and Trusted Computing Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan, China. His research interests include cryptographic protocols.

CONG PENG received the Ph.D. degree in applied mathematics from the School of Mathematics and Statistics, Wuhan University, Wuhan, China, in 2021, where he is currently an Associate Professor with the School of Cyber Science and Engineering. His research interests mainly include applied cryptography and data security.

MIN LUO received the Ph.D. degree in computer science from Wuhan University, Wuhan, China, in 2003, where he is currently a Professor with the School of Cyber Science and Engineering. He has authored or coauthored more than 50 research papers in refereed international journals and conferences, such as IEEE Symposium on Security and Privacy and IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY. His research interests include cryptography and information security, in particular, blockchain security.