Efficient and Provably Secure Distributed Signing Protocol for Mobile Devices in Wireless Networks

Yudi Zhang, Debiao He, Sherali Zeadally, Ding Wang, and Kim-Kwang Raymond Choo, *Senior Member, IEEE*

***Abstract*—Rapid advances in wireless communications, hardware/software, and Internet technologies have contributed to an exponential growth in the number of users accessing the Internet using mobile, wearable or other Internet of Things devices. Identity-based signature schemes have been widely applied to enforce user authorization and validate user messages in mobile wireless networks. However, the user's private key used to generate signatures is prone to leakage because the key is being stored on the mobile device. Several (t, n) threshold secret sharing schemes have been proposed to address the issue. One limitation is that the private keys in most of those schemes have to be recovered on a single device when generating signatures, so that the user who holds the device can sign any message without the participation of other users. To address the recovery limitation, we propose an efficient and secure two-party distributed signing protocol for the identity-based signature scheme in the IEEE P1363 Standard, where two users can generate a valid signature without recovering the whole private key. We formally prove its security under a nonstandard assumption. We also implemented our proposed protocol using the MIRACL Cryptographic software development kit. The experimental results obtained show that the time it takes for two general Android devices to generate a signature is about 709.53 ms.**

***Index Terms*—Distributed signing, IEEE P1363 Standard, mobile device, provable security.**

I. Introduction

TECHNOLOGIES, such as smart mobile devices and Internet of Things (IoT) devices have dramatically changed how we communicate (wired or wireless) in today's increasingly interconnected society. Unlike a wireless network, it is easier to ensure the security of the wired network. According to Statista [1], the number of connected products will triple from 2018 to 2025. For 2020, the installed base of IoT devices is expected to grow to almost 31 billion worldwide. Typically, devices with any intelligence and data collection capability come from different vendors and they often rely on different connection standards and have different network interfaces. However, as shown in Fig. 1, the concept of a smart home assumes every device and sensor can work together, and this highly heterogeneous environment can be integrated into a unified system. Boddy and Shattuck [2] reported that IoT attacks grew 280% from the prior six-month reporting period, with a large number of these attacks attributed to the Mirai malware that infects IoT devices and turns them into bots. Robles *et al.* [3] and Komninos *et al.* [4] showed that additional challenges and issues also exist in smart home.

Fig. 1. Smart home architecture.

In 1984, Shamir [5] proposed the first identity-based cryptography, in an identity-based signature scheme, wherein the user's public key can be obtained from user's ID or e-mail address. After Boneh and Franklin [6] proposed identity-based encryption from the Weil pairing, many other identity-based schemes were proposed [7]–[10]. Identity-based

Manuscript received March 17, 2018; revised May 28, 2018 and July 26, 2018; accepted August 7, 2018. Date of publication August 14, 2018; date of current version January 16, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802500, in part by the National Natural Science Foundation of China under Grant 61572370, Grant 61572379, Grant 61802006, and Grant 61501333, and in part by the open fund of the Guangxi Key Laboratory of Trusted Software under Grant kx201529. The work of K.-K. R. Choo was supported by the Cloud Technology Endowed Professorship. *(Corresponding author: Debiao He.)*

Y. Zhang and D. He are with the Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430000, China, and also with the Guangxi Key Laboratory of Trusted Software, Guilin University of Electronic Technology, Guilin 541004, China (e-mail: zhangyudi007@gmail.com; hedebiao@163.com).

S. Zeadally is with the College of Communication and Information, University of Kentucky, Lexington, KY 40506 USA (e-mail: szeadally@uky.edu).

D. Wang is with the School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: wangdingg@pku.edu.cn).

K.-K. R. Choo is with the Department of Information Systems and Cyber Security, University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: raymond.choo@fulbrightmail.org).

Digital Object Identifier 10.1109/JIOT.2018.2865247

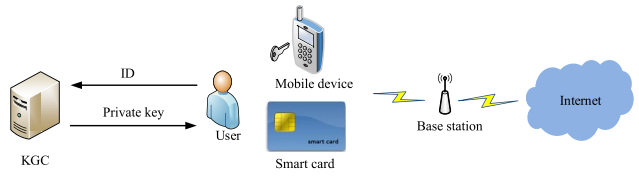


Fig. 2. Typical identity-based signature architecture.

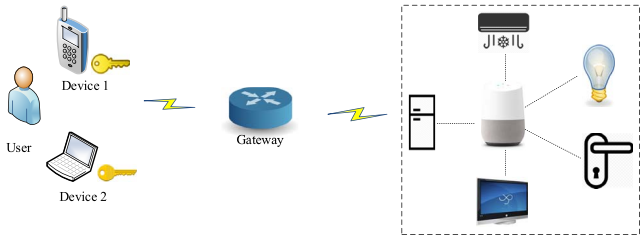


Fig. 3. Remote control smart home.

signature schemes have been used in many practical applications [11]–[13]. Fig. 2 shows a typical identity-based signature architecture for the wireless environment, wherein the user's private key must be used when signing. In Fig. 2, KGC and ID, respectively, denote the key generation center and the user's identity. When a user registers with the network, he/she needs to prove the validation of his/her identity to the server. Similarly, the validation of his/her message should also be verified.

Thus, it is important to be able to authenticate a user and the message. To authenticate an individual, we generally rely on the user "proving" the ownership of the corresponding private signing key by some means [14], [15]. Such private keys are normally stored in the mobile device or smart card, which may be remotely compromised if physically acquired by an attacker [16], [17].

One common approach is the (t, n)-threshold secret sharing scheme [18]–[20], which extends Shamir's secret sharing [21]. The threshold secret sharing scheme has been used in many applications [22]–[24]. In the (t, n)-threshold secret sharing scheme, a private key is shared among n parties. Any information about the private key cannot be obtained from $t-1$ or fewer shares, and with a subset of t or more shares, the whole private key can be recovered. Thus, threshold cryptography provides a high level security for the private key because by corrupting less than $t-1$ parties or devices, the adversary will obtain nothing about the secret.

However, the (t, n)-threshold secret sharing scheme has a key limitation. Specifically, any party who holds the recovered private key can sign any document without the participation of other parties. Moreover, the recovered private key is normally stored in a single device that can be compromised. Several two-party protocols have also been designed to mitigate such a reconstruction limitation [22], [25], [26]. MacKenzie and Reiter [22] presented an S-DSA protocol which allows two-party sign a message m, and they proved the security in the random oracle model. Gennaro *et al.* [25] proposed an efficient and optimal threshold DSA scheme. In their scheme, multiple parties can generate a DSA signature on a message m without reconstructing the private key. Lindell [26] proposed a fast and secure two-party ECDSA signing scheme, which is faster than previous schemes. Compared with the conventional secret sharing scheme, in a two-party protocol, two parties interact with each other and output a signature without recovering the private key. In addition, the existing two-party protocols are designed for traditional public-key cryptography, which have a problem of public key certificate management. To make up for this deficiency, we design a two-party protocol for the identity-based scheme in IEEE P1363 Standard.

The IEEE P1363 project [27] is well-known for issuing standard specifications for public-key cryptography through a series of IEEE standards documents. The IEEE Standard 1363-2000 consists of the following parts [28].

1) Traditional public-key cryptography (1363-2000 & 1363a-2004).
2) Lattice-based public-key cryptography (P1363.1).
3) Password-based public key cryptography (P1363.2).
4) Identity-based public key cryptography using pairings (P1363.3).

The BLMQ [29] signature scheme is the identity-based signature scheme in the IEEE P1363 Standard, and has been widely used in many practical applications [30]–[32]. To enhance security, in this paper, we present an efficient and secure two-party distributed signing protocol for the identity-based signature scheme described in the IEEE P1363 Standard.

A. Application Case

A smart home system will control lighting, temperature, entertainment systems, and appliances. It may also include home security that includes alarm systems. When connected with the Internet, home devices become important constituents of the IoT ecosystem. A home automation system typically connects controlled devices, such as Google Home, Amazon Echo to a central hub. The user can remotely control these IoT devices by using a tablet or desktop computers or a mobile phone application.

However, the user's private key is usually stored on a single device, such as a mobile phone or a tablet. If the device is lost or attacked by malicious applications, others can use the private key to access the central hub and control the IoT devices in the house.

Our scheme can address this major drawback. As shown in Fig. 3, in our scheme, a user's private key is split between two devices. Therefore, even if one of them has been attacked, the adversary cannot get the full private key.

B. Our Research Contributions

We present a two-party distributed signing protocol for the identity-based signature scheme described in the IEEE P1363 Standard, and we implement it on two personal computers (PCs) and two Android devices. We demonstrate that our protocol is more secure, efficient, and practical in a wireless

environment. We summarize the main research contributions of this paper as follows.

1) We propose the design of a novel two-party distributed signing protocol, which is a fast threshold cryptography protocol for the identity-based signature scheme in IEEE P1363. The proposed protocol can generate a valid signature without recovering the private key. Meanwhile, a valid signature cannot be generated if one of the participants is not involved.
2) We present the security analysis of our proposed protocol. The analysis shows that our protocol can satisfy the security requirements when running on two devices. Moreover, we prove the security under the nonstandard assumption, and present the zero-knowledge proof analysis.
3) We implement our protocol by using the MIRACL Cryptographic software development kit (SDK) on two PCs and two Android devices. The evaluation results demonstrate that our protocol achieves good performance with a sign operation for the Barreto–Naehrig (BN) curve, which takes approximately 709.53 ms to execute.

C. Organization of This Paper

The rest of this paper is organized as follows. In Section II, we describe the notations, identity-based signature scheme in IEEE P1363 Standard, zero-knowledge proof and the homomorphic property of the Paillier cryptosystem. In Section III, we present the proposed protocol for two-party identity-based signature generation, and describe our construction. In Section IV, we present the security analysis and the zero-knowledge proof. We present our evaluation results in Section V, before concluding this paper in the last section.

II. Preliminaries

A. Notations

Let D denote a random distribution or set, and $x \xleftarrow{r} D$ denote that x is selected from D randomly. The security parameter is n, and a function $\mu(n)$ is negligible if for any polynomial p, $\mu(n) = O(1/p(n))$. PPT denotes a probabilistic-polynomial time algorithm, and KGC denotes a trusted key generation center. H_1 and H_2 are two secure hash functions, such that $H_1 : \{0,1\}^* \to \mathbb{Z}_p$, and $H_2 : \{0,1\}^* \to \mathbb{Z}_q$.

B. Bilinear Pairing

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic additive groups, $\mathbb{G}_3$ be a multiplicative group, and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$ denotes a bilinear map that satisfies the following properties [29].

1) For all $a_1, a_2 \in \mathbb{G}_1$ and $b_1, b_2 \in \mathbb{G}_2$, $e(a_1 + a_2, b_1) = e(a_1, b_1)e(a_2, b_1)$ and $e(a_1, b_1 + b_2) = e(a_1, b_1)e(a_1, b_2)$.
2) For all $0 \neq a \in \mathbb{G}_1$, there exists $b \in \mathbb{G}_2$ such that $e(a, b) \neq 1$.
3) For all $0 \neq b \in \mathbb{G}_2$, there exists $a \in \mathbb{G}_1$ such that $e(a, b) \neq 1$.

C. Review of the Identity-Based Signature Scheme in IEEE P1363 Standard

Next, we briefly review the identity-based signature scheme in the IEEE P1363 Standard [28], [29], which consists of the following four algorithms.

1) *Setup:* Given a security parameter n, KGC executes the following steps to produce the system parameters params.
   a) Choose $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3$ and a pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_3$.
   b) Pick a random generator Q_2 of $\mathbb{G}_2$, calculate $Q_1 = \phi(Q_2) \in \mathbb{G}_1$.
   c) Generate a random number $s \in \mathbb{Z}_p$ as the master secret key, and calculate $R = sQ_2$ and $g = e(Q_1, Q_2)$.
   d) Set the system parameters params $= (R, g, Q_1, Q_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, e)$ available.
2) *Extract:* Given a user's identity ID, KGC executes the following steps to produce the user's private key.
   a) Compute the identity element $h_{\mathrm{ID}} = H_1(\mathrm{ID})$ in $\mathbb{Z}_p$.
   b) Output $K_{\mathrm{ID}} = (h_{\mathrm{ID}} + s)^{-1} Q_1$.
3) *Sign:* Given a message m, the user with the identity ID, the signer executes the following steps to generate the signature σ.
   a) Compute $u = g^r$, where r is a random integer.
   b) Compute $h = H_2(m, u)$ and $S = (r + h)K_{\mathrm{ID}}$.
   c) Output $\sigma = (h, S)$.
4) *Verify:* Given the signature $\sigma = (h, S)$, the identity ID and the message m, the verifier executes the following steps to verify the validation of the signature.
   a) Compute $h_{\mathrm{ID}} = H_1(\mathrm{ID})$.
   b) Compute $u = ([e(S, h_{\mathrm{ID}}Q_2 + R)]/[e(Q_1, Q_2)^h])$.
   c) If $h = H_2(m, u)$, then output 1, otherwise output 0.

D. Zero-Knowledge Proof

The ideal zero knowledge functionality is $\mathcal{F}_{\mathrm{zk}}$, and the standard ideal zero-knowledge functionality is defined by $((x, w), \lambda) \to (\lambda, (x, R(x, w)))$, where λ denotes the empty string.

Definition 1: The zero knowledge functionality $\mathcal{F}_{\mathrm{zk}}^R$ for relation R: upon receiving $(\mathsf{prove}, \mathrm{sid}, x, w)$ from P_i $(i \in \{1, 2\})$: if $(x, w) \notin R$ or sid has been previously used, then ignore the message. Otherwise, send $(\mathsf{proof}, \mathrm{sid}, x)$ to P_{3-i}. The noninteractive zero-knowledge proof of knowledge satisfying $\mathcal{F}_{\mathrm{zk}}$ [33] can be achieved in random oracle model.

E. Paillier Encryption

In our proposed protocol, we use the Paillier cryptosystem [34] for encryption. The Paillier cryptosystem is defined as follows.

1) *Key Generation:*
   a) Choose two equivalent length large prime numbers p and q randomly.
   b) Compute $g = n + 1$, $\lambda = \phi(n)$ and $\mu = (\phi(n))^{-1} \bmod n$, where $\phi(n) = (p - 1)(q - 1)$.

c) The public key is pk $= (n, g)$, the private key is sk $= (\lambda, \mu)$.

2) *Encryption:*
   a) Select a random number r, where $r \in \mathbb{Z}_n^*$.
   b) Compute ciphertext $c = \mathsf{Enc}_{\mathrm{pk}}(m) = g^m \cdot r^n \bmod n^2$, where $0 \le m < n$.
3) *Decryption:*
   a) Decrypt the ciphertext as $m = \mathsf{Dec}_{\mathrm{sk}}(c) = L(c^{\lambda} \bmod n^2) \cdot \mu \bmod n$, where $L(x) = ([x-1]/n)$.

In this paper, $\mathsf{Enc}_{\mathrm{pk}}(\cdot)$ denotes the encrypt operation using public key pk, and $\mathsf{Dec}_{\mathrm{sk}}(\cdot)$ denotes the decrypt operation using private key sk. In the Paillier cryptosystem, there is a notable feature which is its homomorphic properties.

1) $\mathsf{Dec}_{\mathrm{sk}}(\mathsf{Enc}_{\mathrm{pk}}(m_1) \cdot \mathsf{Enc}_{\mathrm{pk}}(m_2)) = m_1 + m_2$.
2) $\mathsf{Dec}_{\mathrm{sk}}(\mathsf{Enc}_{\mathrm{pk}}(m_1)^{m_2}) = m_1 m_2$.

Let $c_1 = \mathsf{Enc}_{\mathrm{pk}}(m_1)$, $c_2 = \mathsf{Enc}_{\mathrm{pk}}(m_2)$, then $c_1 \oplus c_2 = \mathsf{Enc}_{\mathrm{pk}}(m_1 + m_2)$, $m_2 \otimes c_1 = \mathsf{Enc}_{\mathrm{pk}}(m_1)^{m_2}$.

III. Proposed Two-Party Distributed Signing Protocol

In this section, we present our proposed two-party distributed signing protocol for identity-based signature in IEEE P1363 Standard. We show the detailed two key phases, i.e., distributed key generation phase (see Section III-A) and the distributed signature generation phase (see Section III-B).

A. Distributed Key Generation

In the distributed key generation phase, the main algorithms are processed by KGC. As shown in Fig. 4, KGC distributes the private key K_{ID} into two parts.

The steps of distributed key generation are shown in Phase 1 and Fig. 4.

Phase 1 (Distributed Key Generation):

1) KGC computes the user ID's identity element $h_{\mathrm{ID}} = H_1(\mathrm{ID})$.
2) KGC chooses $t_1 \xleftarrow{r} \mathbb{Z}_q$, and computes $t_2 = t_1^{-1} \cdot (h_{\mathrm{ID}} + s)^{-1}$.
3) KGC sets $K_{\mathrm{ID}}^{(1)} = t_1 Q_1$, and $K_{\mathrm{ID}}^{(2)} = t_2$.
4) KGC generates a Paillier key-pair (pk, sk) for P_1.
5) KGC sends $(K_{\mathrm{ID}}^{(1)}, \mathrm{pk}, \mathrm{sk})$ to P_1, and sends $(K_{\mathrm{ID}}^{(2)}, \mathrm{pk})$ to P_2.
6) P_1 stores $(\mathrm{ID}, K_{\mathrm{ID}}^{(1)}, \mathrm{pk}, \mathrm{sk})$ and the public parameter params, and P_2 stores $(\mathrm{ID}, K_{\mathrm{ID}}^{(2)}, \mathrm{pk})$ and the public parameter params.

It is worth noting that we can check the correctness of the equation $K_{\mathrm{ID}}^{(2)} K_{\mathrm{ID}}^{(1)} = (h_{\mathrm{ID}} + s)^{-1} Q_1$.

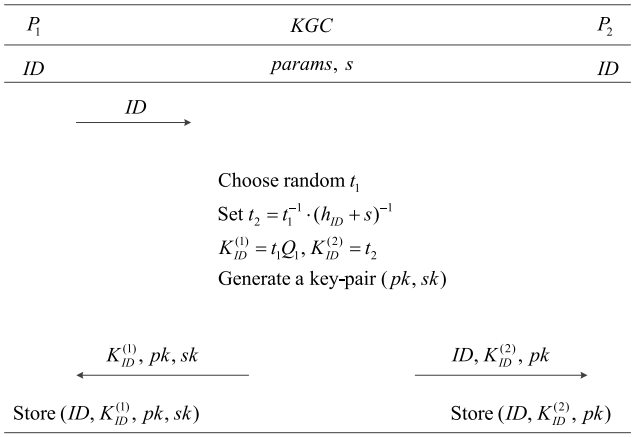


Fig. 4. Distributed key generation.

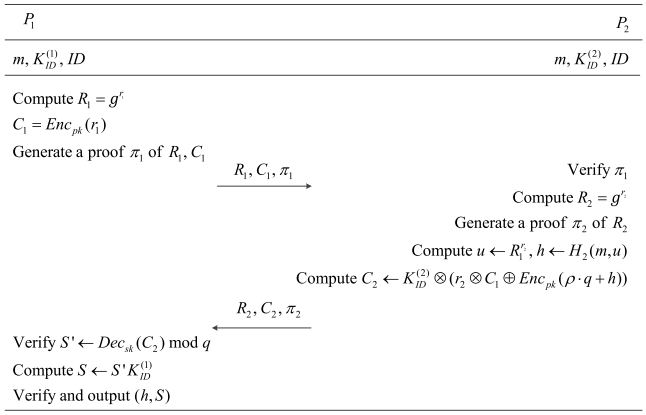


Fig. 5. Distributed signature generation.

B. Distributed Signature Generation

In the distributed signature generation phase, P_1 and P_2 select r_1 and r_2, respectively, where $r_1 r_2 = r$, and $u = g^{r_1 r_2}$. P_1 encrypts r_1 under pk (i.e., $C_1 = \mathrm{Enc}(r_1)$), computes $R_1 = g^{r_1}$, and sends R_1, C_1 to P_2. After P_2 receives R_1, C_1, it is trivial for P_2 to compute a ciphertext of $S'' = (r_1 r_2 + h) \cdot K_{\mathrm{ID}}^{(2)}$ under P_1's public key. In order to prevent revealing any information to P_1, P_2 chooses a random integer $\rho \xleftarrow{r} \mathbb{Z}_q$, and computes the ciphertex C_2 of $S'' = (r_1 r_2 + h + \rho \cdot q) \cdot K_{\mathrm{ID}}^{(2)}$. Then, P_1 computes $S' = \mathsf{Dec}_{sk}(C_2) \bmod q$, and finally computes the signature $S = S' K_{\mathrm{ID}}^{(1)}$.

To verify if the messages that P_1 communicated with P_2 are correct, we use the zero-knowledge proof. We describe the distributed signature generation phase formally in Phase 2, which is also presented in Fig. 5.

Phase 2 (Distributed Signature Generation):

1) *P_1's First Message:*
   a) P_1 chooses $r_1 \xleftarrow{r} \mathbb{Z}_q$, and computes $R_1 = g^{r_1}$.
   b) P_1 computes $C_1 = \mathsf{Enc}_{\mathrm{pk}}(r_1)$.
   c) P_1 sends $(\mathsf{prove}, 1, (R_1, C_1), (r_1, \mathrm{sk}))$ to $\mathcal{F}_{zk}^{R_{\mathrm{PDL}}}$.
2) *P_2's First Message:*
   a) P_2 receives $(\mathsf{proof}, 1, (R_1, C_1))$ from $\mathcal{F}_{zk}^{R_{\mathrm{PDL}}}$, if not or $R_1 = g^0$, it aborts.
   b) P_2 chooses $r_2 \xleftarrow{r} \mathbb{Z}_q$, and computes $R_2 = g^{r_2}$.
   c) P_2 sends $(\mathsf{prove}, 2, R_2, r_2)$ to $\mathcal{F}_{zk}^{R_{\mathrm{DL}}}$.
   d) P_2 computes $u = R_1^{r_2}$, $h = H_2(m, u)$.
   e) P_2 chooses $\rho \xleftarrow{r} \mathbb{Z}_q$, computes $C_2 = K_{\mathrm{ID}}^{(2)} \otimes (r_2 \otimes C_1 \oplus \mathsf{Enc}_{\mathrm{pk}}(\rho \cdot q + h))$.
   f) P_2 sends C_2 to P_1.
3) *P_1 Generates the Output:*
   a) P_1 receives $(\mathsf{proof}, 2, R_2)$ from $\mathcal{F}_{zk}^{R_{\mathrm{DL}}}$; if not; it aborts.
   b) P_1 computes $S' = \mathsf{Dec}_{\mathrm{sk}}(C_2) \bmod q$, then computes $S = S' K_{\mathrm{ID}}^{(1)}$.

c) P_1 computes $u = R_2^{r_1}$ and $h = H_2(m, u)$.
d) P_1 verifies (h, S) by the identity ID, if the signature is valid, it then outputs (h, S), otherwise, it aborts.

Correctness: Due to $C_1 = \mathsf{Enc}_{\mathrm{pk}}(r_1)$, $C_2 = K_{\mathrm{ID}}^{(2)} \otimes (r_2 \otimes C_1 \oplus \mathsf{Enc}_{\mathrm{pk}}(\rho \cdot q + h))$, $R_2 = g^{r_2}$, then P_1 can compute

$$\begin{aligned} u &= g^{r_1 r_2} \\ S &= \mathsf{Dec}_{\mathrm{sk}}(C_2) \mod q \cdot K_{\mathrm{ID}}^{(1)} \\ &= \mathsf{Dec}_{\mathrm{sk}}((C_1^{r_2} + \mathsf{Enc}_{\mathrm{pk}}(\rho \cdot q + h))^{K_{\mathrm{ID}}^{(2)}}) \mod q \cdot K_{\mathrm{ID}}^{(1)} \\ &= \mathsf{Dec}_{\mathrm{sk}}(\mathsf{Enc}_{\mathrm{pk}}(r_1)^{r_2} + \mathsf{Enc}_{\mathrm{pk}}(\rho \cdot q + h))^{K_{\mathrm{ID}}^{(2)}} \mod q \cdot K_{\mathrm{ID}}^{(1)} \\ &= ((r_1 r_2 + h + \rho \cdot q) \cdot K_{\mathrm{ID}}^{(2)}) \mod q \cdot K_{\mathrm{ID}}^{(1)} \\ &= (r_1 r_2 + h) \cdot K_{\mathrm{ID}}^{(2)} K_{\mathrm{ID}}^{(1)}. \end{aligned}$$

Therefore, the correctness of the proposed distributed signing protocol for the identity-based signature scheme in the IEEE P1363 Standard is proven.

IV. Security Analysis

A. Security Model

Definition 2 (IND-CPA Security): Let $\mathcal{A}$ be a PPT adversary, $\mathcal{C}$ be a challenger. The IND-CPA security is defined by the following game with a negligible advantage.

1) $\mathcal{C}$ generates the key-pair (pk, sk), $\mathcal{A}$ obtains pk.
2) $\mathcal{A}$ outputs two messages m_0, m_1 ($|m_0| = |m_1|$).
3) $\mathcal{C}$ selects $b \xleftarrow{r} \{0, 1\}$ and encrypts m_b such that $C^* = \mathsf{Enc}_{\mathrm{pk}}(m_b)$, then returns C^* to $\mathcal{A}$.
4) $\mathcal{A}$ outputs b', $\mathcal{A}$ wins the game when $b' = b$.

Definition 3: We define an experiment $\mathsf{Sign}_{\mathcal{A},\pi}(1^n)$, where π is a secure digital signature scheme such that $\pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$.

1) $(vk, \mathrm{sk}) \leftarrow \mathsf{Gen}(1^n)$.
2) $(m^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{Sign}_{\mathrm{sk}}(\cdot)}(1^n, vk)$.
3) Let $\mathcal{M}$ be the set of all m which can be queried. $\mathcal{A}$ can query oracle with m. Then, the experiment outputs 1 if $m^* \notin \mathcal{M}$ and $\mathsf{Verify}(m^*, \sigma^*) = 1$.

Definition 4: A signature scheme π is existentially unforgeable under chosen message attack (CMA) if for every probabilistic polynomial-time oracle machine $\mathcal{A}$, there exists a negligible function μ such that for every n

$$\Pr[\mathsf{Sign}_{\mathcal{A},\pi}(1^n) = 1] < \mu(n).$$

In the distributed signature generation phase, we define the experiment $\mathsf{DistSign}_{\mathcal{A},\Pi}^{b}(1^n)$, an adversary $\mathcal{A}$ who can control a party P_b ($b \in 1, 2$). In protocol Π, the honest party P_{3-b} instructs a stateful oracle $\Pi_b(.,.)$. $\mathcal{A}$ can choose which message needs to be signed, and can interact with party P_{3-b}. In this definition, the distributed signature generation phase should run after the distributed key generation phase. The oracle is queried by a session identifier and an input, and works as follows.

1) Upon receiving a query (sid, m), if the distributed key generation phase has not been executed, then the oracle output $\perp$.
2) Upon receiving a query (sid, m) after the distributed key generation phase has been executed, the oracle invokes a machine M_{sid} which is instructed by P_{3-b} in protocol Π. M_{sid} is initialized with key share and any stored information from KGC in the distributed key generation phase. If P_{3-b} sends the first message in the signing phase, then the oracle outputs this message.
3) Upon receiving a query (sid, m) after the distributed key generation phase has executed and sid has been queried, the oracle sends the message m to M_{sid}, and returns the next message output from M_{sid}. If M_{sid} finishes execution, then it returns M_{sid}'s output.

In this experiment, $\mathcal{A}$ can control a party P_b with oracle access to Π_b. $\mathcal{A}$ wins, if it can forge a signature on a message m^* that has not been queried in the oracle.

Definition 5: We define an experiment $\mathsf{DistSign}_{\mathcal{A},\Pi}^{b}(1^n)$. Let $\pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$ be a two-party signing phase.

1) $(m^*, \sigma^*) \leftarrow \mathcal{A}^{(\Pi_b(.,.))}(1^n)$.
2) Let $\mathcal{M}$ be the set of all m which can be queried. $\mathcal{A}$ can query oracle with (sid, m). Then, the experiment outputs 1 if $m^* \notin \mathcal{M}$ and $\mathsf{Verify}(m^*, \sigma^*) = 1$.

Definition 6: A protocol Π is a secure two-party protocol for distributed signature generation for π, if for every $\mathsf{P.P.T}$ algorithm $\mathcal{A}$ and every $b \in \{1, 2\}$, there exists a negligible function μ for every n, $\Pr[\mathsf{DistSign}_{\mathcal{A},\Pi}^{b}(1^n) = 1] \leq \mu(n)$.

Definition 7: The functionality $\mathcal{F}_{\mathrm{BLMQ}}$ is combined with two functions: 1) extraction and 2) signing. The extraction function can be queried only once and after the key extraction phase, the signing function can be queried an arbitrary number of times. $\mathcal{F}_{\mathrm{BLMQ}}$ works with parties P_1 and P_2, and is defined as follows.

1) After receiving $\mathsf{Extract}(\mathsf{params}, \mathrm{ID})$ from both P_1 and P_2.
   a) Generate a BLMQ key pair $(h_{\mathrm{ID}}, K_{\mathrm{ID}})$ by computing $h_{\mathrm{ID}} = H_1(\mathrm{ID})$, and choose a random number $s \xleftarrow{r} \mathbb{Z}_p$. Compute $K_{\mathrm{ID}} = (h_{\mathrm{ID}} + s)^{-1} Q_1$. Choose a hash function $H_q : \{0, 1\}^* \rightarrow \{0, 1\}^{\lfloor \log |q| \rfloor}$ and store params, ID, H_q, K_{ID}.
   b) Send h_{ID} and H_q to both P_1 and P_2.
   c) Ignore future queries to $\mathsf{Extract}$.
2) After receiving $\mathsf{Sign}(\mathrm{sid}, m)$ from both P_1 and P_2, if $\mathsf{Extract}$ was queried and sid has not been used, then compute a BLMQ signature (h, S) of the message m by follows.
   a) Choose a number $r \xleftarrow{r} \mathbb{Z}_q$, compute $u = g^r$.
   b) Compute $h = H_q(m, u)$, and $S = (r + h) K_{\mathrm{ID}}$.

   Finally, send the signature (h, S) to both P_1 and P_2.

Definition 8: The Paillier-EC assumption [26] is hard. For every $\mathsf{P.P.T}$ adversary $\mathcal{A}$ there exists a negligible function μ that $\Pr[\mathsf{Paillier\text{-}EC}_{\mathcal{A}}(1^n) = 1] \leq (1/2) + \mu(n)$. Let G be a generator of a group $\mathbb{G}$ of order q. The experiment $\mathsf{Paillier\text{-}EC}_{\mathcal{A}}(1^n)$ is defined as follows.

1) Generate a Paillier key piar (pk, sk).
2) Select $r_0, r_1 \xleftarrow{r} \mathbb{Z}_q$ and compute $R = r_0 \cdot G$.
3) Select $b \in \{0, 1\}$ and compute $C = \mathsf{Enc}_{\mathrm{pk}}(r_b)$.
4) Let $b' = \mathcal{A}^{\mathcal{O}_C(\cdot,\cdot,\cdot)}$, if $\mathsf{Dec}_{\mathrm{sk}}(C') = \alpha + \beta \cdot r_b \mod q$, $\mathcal{O}_C(C', \alpha, \beta) = 1$.
5) If and only if $b' = b$, the experiment outputs 1.

B. Proof of Security

In this section, we prove that the protocol Π is a secure two-party protocol for distributed signature generation as shown in Theorem 1.

Theorem 1: If Paillier encryption is indistinguishable under chosen plain attack (CPA), and BLMQ signature is existentially unforgeable under a CMA, then our two-party protocol for distributed signature generation of identity-based signature in IEEE P1363 is secure.

Proof: We now prove the security of our proposed protocol. In addition, if $\mathcal{A}$ can break the protocol of zero-knowledge with the probability ϵ, then it can break the protocol with probability $\epsilon + \mu(n)$, μ is a negligible function.

In our proof, for any adversary $\mathcal{A}$ that launches an attack on the protocol, we construct an adversary $\mathcal{S}$ who forges a BLMQ signature in Definition 3 with the probability that is negligibly close to the probability that $\mathcal{A}$ forges a signature in Definition 5.

If Paillier encryption is indistinguishable under CPA, then for every P.P.T algorithm $\mathcal{A}$ and every $b \in \{1, 2\}$, there exists a P.P.T algorithm $\mathcal{S}$ and a negligible function μ such that for every n

$$|\Pr\left[\mathsf{Sign}_{\mathcal{S},\pi}(1^n) = 1\right] - \Pr\left[\mathsf{DistSign}^{b}_{\mathcal{A},\Pi}(1^n)\right] = 1| \leq \mu(n) \tag{1}$$

where Π denotes the protocol of Phase 2, and π denotes the BLMQ signature scheme. If we assume that BLMQ signature is secure, there exists a negligible function μ' for every n that $\Pr[\mathsf{Sign}_{\mathcal{S},\pi}(1^n) = 1] \leq \mu'(n)$. With (1), we conclude that $\Pr[\mathsf{Sign}^{b}_{\mathcal{A},\pi}(1^n) = 1] \leq \mu(n) + \mu'(n)$. We now prove (1) for $b = 1$ and $b = 2$, respectively.

When $b = 1$ (i.e., P_1 is the corrupted one), let $\mathcal{A}$ be a P.P.T adversary in $\mathsf{DistSign}^{1}_{\mathcal{A},\Pi}(n)$, we construct a P.P.T adversary $\mathcal{S}$ for $\mathsf{Sign}_{\mathcal{S},\pi}(n)$. $\mathcal{S}$ simulates the execution for $\mathcal{A}$ as follows.

1) In Sign, $\mathcal{S}$ receives $(1^n, \mathrm{ID})$, where ID is the user's identity which could generate the user's public key $H_1(\mathrm{ID})$.
2) $\mathcal{S}$ invokes $\mathcal{A}$ on input 1^n and simulates oracle $\mathcal{A}$ in DistSign. Upon receiving a query (sid, m), where sid is a new session identifier, $\mathcal{S}$ queries its signing oracle in Sign with m and receives a signature (h, S) where $S = \overline{S} \cdot Q_1$. We slightly modify the oracle that lets the signing oracle return $\overline{S}$ to the simulator $\mathcal{S}$. We let the adversary $\mathcal{A}$ compute $t_1 \cdot \mathsf{Dec}_{\mathrm{sk}}(C_2)$. u can be computed by the BLMQ signature verification algorithm. Then, $\mathcal{A}$ queries $\mathcal{S}$ with identifier sid. The query is processed as follows.
   a) The first message (sid, m_1) is processed by parsing the message m_1 since $m_1 = (\mathsf{prove}, 1, (R_1, C_1), (r_1, \mathrm{sk}))$. If $R_1 = g^{r_1}$ and $C_1 = \mathsf{Enc}_{\mathrm{pk}}(r_1)$, then $\mathcal{S}$ sets $R_2 = u^{1/r_1}$, and sets the oracle's reply message is $(\mathsf{proof}, 2, R_2)$ and sends it to $\mathcal{A}$. Otherwise, $\mathcal{S}$ simulates P_2 abortion.
   b) $\mathcal{S}$ chooses $\rho \xleftarrow{r} \mathbb{Z}_q$, then computes $C_2 = \mathsf{Enc}_{\mathrm{pk}}(\overline{S} + \rho \cdot q)(t_1)^{-1}$, where $\overline{S}$ is the value from the signature S received from $\mathcal{F}_{\mathrm{BLMQ}}$, and sets the oracle's reply message is C_2 and sends it to $\mathcal{A}$.
3) Once $\mathcal{A}$ halts and outputs (m^*, σ^*), $\mathcal{S}$ outputs (m^*, σ^*) and halts.

We now prove that (1) holds. In Phase 2, the view of $\mathcal{A}$ in the simulation of the distributed signature generation phase is computationally indistinguishable from its view from a real process of Phase 2. The difference between $\mathcal{A}$'s view in real execution and in the simulation is C_2. In addition, because u is generated randomly by $\mathcal{F}_{\mathrm{BLMQ}}$, and the distribution between u^{1/r_1} and g^{r_2} is identical then R_2's distribution between the real execution and the simulation is identical. The zero-knowledge proof and verification are also identically distributed. So, the only difference is C_2. During the simulation, it is an encryption of $(\overline{S}+\rho\cdot q)(t_1)^{-1}$ whereas in a real execution, it is a ciphertext of $(r + \rho \cdot q + h) \cdot K^{(2)}_{\mathrm{ID}}$.

We observe that, in the definition of BLMQ signature, $S = (r + h)K_{\mathrm{ID}} = (r + h)K^{(1)}_{\mathrm{ID}}K^{(2)}_{\mathrm{ID}} \mod q$. Thus, $(r + h)K^{(2)}_{\mathrm{ID}} = (t_1)^{-1} \cdot \overline{S} \mod q$ means that there exists an integer $l \in \mathbb{Z}_q$ that $(r + h)K^{(2)}_{\mathrm{ID}} = (t_1)^{-1} \cdot \overline{S} + l \cdot q$. In the protocol, the operation without a modular reduction is $(r + h)K^{(2)}_{\mathrm{ID}}$. Therefore, the difference between the real execution and the simulation with $\mathcal{S}$ as follows.

1) *Real:* The ciphertext C_2 encrypts $(t_1)^{-1} \cdot \overline{S} \mod q + l \cdot q + \rho \cdot q$.
2) *Simulation:* The ciphertext C_2 encrypts $(t_1)^{-1} \cdot \overline{S} \mod q + \rho \cdot q$.

It is worth pointing out that the distribution between the real execution and the simulation is identical. This proves that (1) holds for $b = 1$.

When $b = 2$ (i.e., P_2 is the corrupted one). The message C_2 sent by P_2 may be corrupted by $\mathcal{A}$, and the simulator cannot detect whether C_2 is a correct ciphertext. As in [26], we let $\mathcal{S}$ simulates P_1 abortion at some random point, $\mathcal{S}$ chooses $i \xleftarrow{r} \{1, \ldots, p(n) + 1\}$ randomly, where $p(n)$ is the upper bound on the number of queries made by $\mathcal{A}$. $\mathcal{S}$ chooses i with the probability of $(1/[p(n) + 1])$, that is $\mathcal{S}$ simulates $\mathcal{A}$'s view with a probability of $(1/[p(n) + 1])$. The probability of $\mathcal{S}$ forging a signature in Sign is at least $(1/[p(n) + 1])$ times of the probability that $\mathcal{A}$ forges a signature in DistSign.

Let $\mathcal{A}$ be a P.P.T adversary in $\mathsf{DistSign}^{2}_{\mathcal{A},\Pi}(n)$, we construct a P.P.T adversary $\mathcal{S}$ for $\mathsf{Sign}_{\mathcal{S},\pi}(n)$. The adversary $\mathcal{S}$ simulates the execution for $\mathcal{A}$ as follows.

1) In Sign, $\mathcal{S}$ receives $(1^n, \mathrm{ID})$, where ID is the identity to generate the user's public key $H_1(\mathrm{ID})$.
2) $\mathcal{S}$ invokes $\mathcal{A}$ on input 1^n and simulates oracle $\mathcal{A}$ in DistSign. Upon receiving a query (sid, m), where sid is a new session identifier, $\mathcal{S}$ sets the oracle reply with $(\mathsf{proof}, 1, R_1, C_1)$, where $R_1 = u^{1/r_2}$, and sends it to $\mathcal{A}$. Then, $\mathcal{S}$ queries its signing oracle in Sign with m and receives a signature (h, S) and $\mathcal{S}$ can compute u in the BLMQ signature verification algorithm. Then, $\mathcal{A}$ queries $\mathcal{S}$ with identifier sid, which is processed as follows.
   a) The first message (sid, m_1) is processed by parsing m_1 as $(\mathsf{prove}, 2, R_2, r_2)$ which should be sent to $\mathcal{F}^{R_{\mathrm{DL}}}_{zk}$. $\mathcal{S}$ verifies the equation $R_2 = g^{r_2}$ and if the equation does not hold, it simulates P_1 causing it to abort the protocol.
   b) The second message (sid, m_2) is processed by parsing m_2 as C_2. If this is the ith query by

$\mathcal{A}$, then $\mathcal{S}$ simulates P_1's abortion. Otherwise, it continues.

3) Once $\mathcal{A}$ halts and outputs (m^*, σ^*), $\mathcal{S}$ outputs (m^*, σ^*) and halts.

Let j be the first query to oracle Π with (sid, m_2), and P_1 does not obtain the valid signature (h, S) which corresponds to the public key $H_1(\text{ID})$. If $j = i$, then the difference between the distribution of $\mathcal{A}$'s view in real execution and the simulated execution by $\mathcal{S}$ is the ciphertext C_1. Since $\mathcal{S}$ does not hold the Paillier private key in the simulation, the indistinguishability of the simulation follows from a reduction of indistinguishability of the encryption scheme under the CPA.

We can learn that

$$|\Pr\left[\text{Sign}_{\mathcal{S},\pi}(1^n) = 1 | i = j\right] - \Pr\left[\text{DistSign}^2_{\mathcal{A},\Pi}(1^n) = 1\right]| \le \mu(n)$$

so

$$\Pr\left[\text{DistSign}^2_{\mathcal{A},\Pi}(1^n) = 1\right] \le \frac{\Pr\left[\text{Sign}_{\mathcal{S},\pi}(1^n) = 1\right]}{1/(p(n)+1)} + \mu(n)$$

i.e.,

$$\Pr\left[\text{Sign}_{\mathcal{S},\pi}(1^n) = 1\right] \ge \frac{\text{DistSign}^2_{\mathcal{A},\Pi}(1^n) = 1}{1/(p(n)+1)} - \mu(n).$$

It means that if $\mathcal{A}$ can forge a signature in $\text{DistSign}^2_{\mathcal{A},\Pi}(1^n)$ with a non-negligible probability, then $\mathcal{S}$ can forge a signature in $\text{Sign}_{\mathcal{S},\pi}(1^n)$ with a non-negligible probability. Due to BLMQ signature being existentially unforgeable [29], therefore, our protocol is secure. This completes the proof of Theorem 1. ■

Theorem 2: If the Paillier-EC assumption is hard, then Phase 2 computes $\mathcal{F}_{\text{BLMQ}}$ securely in the $\mathcal{F}_{\text{zk}}$ model in the presence of a malicious static adversary.

Proof: We analyze the security for the case of a corrupted P_1 and a corrupted P_2. First, let P_1 be corrupted by an adversary $\mathcal{A}$, we construct a simulator $\mathcal{S}$.

In the signing phase, P_1 cannot do anything. All it does is to generate u and receive a ciphertex C_2 from P_2. Due to the simulator's ability to simulate the protocol in the signing phase, a simulator can make the result equal to u in a signature received from $\mathcal{F}_{\text{BLMQ}}$. Therefore, the main challenge is to prove that the simulator can generate P_1's view of the decryption of C_2, given only $\overline{S}$, (h, S) from $\mathcal{F}_{\text{BLMQ}}$, where $S = \overline{S} \cdot Q_1$.

1) On input $\text{Sign}(\text{sid}, m)$, $\mathcal{S}$ sends $\text{Sign}(\text{sid}, m)$ to $\mathcal{F}_{\text{BLMQ}}$ and receives a signature (h, S).
2) $\mathcal{S}$ computes u using the BLMQ verification procedure.
3) $\mathcal{S}$ invokes $\mathcal{A}$ with input $\text{Sign}(\text{sid}, m)$ and simulates the following messages to ensure that the result is u.
   a) $\mathcal{S}$ receives $(\text{prove}, 1, (R_1, C_1), (r_1, \text{sk}))$ from $\mathcal{A}$.
   b) If $R_1 = g^{r_1}$, then $\mathcal{S}$ sets $R_2 = u^{1/r_1}$, and sends $(\text{proof}, 2, R_2)$ to $\mathcal{A}$. Otherwise, $\mathcal{S}$ simulates P_2 abort, sends abortion to $\mathcal{F}_{\text{BLMQ}}$.
4) $\mathcal{S}$ chooses $\rho \xleftarrow{r} \mathbb{Z}_q$, then computes $C_2 = \text{Enc}_{\text{pk}}(\overline{S} + \rho \cdot q)(t_1)^{-1}$, where $\overline{S}$ is the value from the signature S received from $\mathcal{F}_{\text{BLMQ}}$, and sets the oracle reply $\mathcal{A}$ with C_2.

The only difference between the view of $\mathcal{A}$ in real and simulation is the way that C_2 is chosen. In the simulation, it is a ciphertext of $(\overline{S} + \rho \cdot q)(t_1)^{-1}$, in real execution, it is a ciphertext of $(r + \rho \cdot q + h) \cdot K_{\text{ID}}^{(2)}$. It is statistically close between these two scenarios.

Let P_2 be corrupted by $\mathcal{A}$, and we construct a simulator $\mathcal{S}$. In the signature generation phase, $\mathcal{S}$ works as follows.

1) On input $\text{Sign}(\text{sid}, m)$, $\mathcal{S}$ sends $\text{Sign}(\text{sid}, m)$ to $\mathcal{F}_{\text{BLMQ}}$ and receives a signature (h, S).
2) $\mathcal{S}$ computes u using the BLMQ verification procedure.
3) $\mathcal{S}$ invokes $\mathcal{A}$ with $\text{Sign}(\text{sid}, m)$, sets $R_1 = u^{1/r_2}$ and sends $\mathcal{A}$ the message $(\text{proof}, 1, (R_1, C_1))$ internally.
4) $\mathcal{S}$ receives $(\text{prove}, 2, R_2, r_2)$ which indicates that $\mathcal{A}$ intends to send to $\mathcal{F}_{zk}^{R_{\text{DL}}}$.
5) $\mathcal{S}$ verifies the equation $R_2 = g^{r_2}$. If the equation does not hold, then $\mathcal{S}$ simulates P_1 abort.
6) $\mathcal{S}$ receives C_2 from P_2, decrypts C_2 by using sk and reduces the result by modulo q. $\mathcal{S}$ checks if it is equal to $((\tilde{r_1} r_2 + h) K_{\text{ID}}^{(2)}) \mod q$, where $C_1 = \text{Enc}_{\text{pk}}(\tilde{r_1})$. If the equation holds, then $\mathcal{S}$ sends "continue" to the trusted party P_1, and lets P_1 provide the output. Otherwise, $\mathcal{S}$ sends "abort" to P_1 to instruct P_1 to abort.

We modify $\mathcal{S}$ to a simulator $\mathcal{S}'$ which has an oracle $\mathcal{O}_c(c', \alpha, \beta)$. The oracle $\mathcal{O}_c(c', \alpha, \beta)$ outputs 1 if $\text{Dec}_{\text{sk}}(c', \alpha, \beta) = \alpha + \beta \cdot \tilde{r_1} \mod q$. $\mathcal{S}'$ simulates $\mathcal{S}$ as follows.

1) Compute $\alpha = h \cdot K_{\text{ID}}^{(2)} \mod q$.
2) Compute $\beta = r_2 \cdot K_{\text{ID}}^{(2)} \mod q$.
3) Query $\mathcal{O}_c(c', \alpha, \beta)$ to get b.
4) If $b = 1$, $\mathcal{S}'$ continues to simulate $\mathcal{S}$.

$\mathcal{S}$ accepts if $\mathcal{S}'$ accepts because these checks by $\mathcal{S}$ and $\mathcal{S}'$ are equivalent. Due to the Paillier-EC assumption [26], we conclude that the output generated by $\mathcal{S}'$ in the ideal model is computationally indistinguishable from the real execution. Since the output distributions of $\mathcal{S}$ and $\mathcal{S}'$ are identical in the ideal model, therefore the output generated by $\mathcal{S}$ in the ideal model is computationally indistinguishable from the real execution. This completes the proof for Theorem 2. ■

C. Zero-Knowledge Proof

In our protocol, the main zero-knowledge proof for the relation is R_{PDL}, which is defined by [26]. We use this zero-knowledge proof directly. Thus, we omit the construction here.

1) Proof That r Is Discrete Log of R: In this section, we propose the constructions of zero-knowledge proof for the relation R_{DL} such that

$$R_{\text{DL}} = \{(\mathbb{G}_3, g, R, r) | R = g^r\}.$$

We use Schnorr zero knowledge proof [35] to satisfy this requirement. In the signing phase, if a malicious P_2 sends (prove, R_2, r_2) to $\mathcal{F}_{zk}^{R_{\text{DL}}}$, then it also receives a correct message (proof, R_1) from $\mathcal{F}_{zk}^{R_{\text{DL}}}$. However, P_1's message in the protocol is assumed to be zero knowledge and hence, does not reveal any information about the random integer r_1. The detailed zero-knowledge proof protocol is described as follows.

TABLE I
SECURITY LEVEL

Type	Symmetric cipher key length	Bit length of p in prime field $\mathbb{F}_p$
MNT $k = 6$	80	160
BN $k = 12$	128	256
KSS $k = 18$	192	512
BLS $k = 24$	256	640

TABLE II
EACH ALGORITHM OF BLMQ SIGNATURE (AVERAGES WERE COMPUTED OVER 1000 EXECUTIONS)

Curve \ Algorithm	KeyGen (milliseconds)	Sign (milliseconds)	Verify (milliseconds)
$k = 6$	5.83	22.16	139.94
$k = 12$	8.22	66.01	188.61
$k = 18$	25.99	733.96	2738.34
$k = 24$	61.76	18676.92	36478.61

The joint statement is $(\mathbb{G}_3, g, R)$, the prover has a witness r and wishes to prove that $R = g^r$. When the Schnorr zero-knowledge proof generation algorithm uses as input the public parameter $P = (R, g, Q_1, Q_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3)$, signer's identity ID, secret value r, and public value $V = g^r$. It outputs (z, e) as follows.

1) Select $k \xleftarrow{r} \mathbb{Z}_q$, compute $K = g^k$.
2) Compute $e = H(P, \mathrm{ID}, K, V)$.
3) Compute $z = K - re$.

The Schnorr zero-knowledge proof verification algorithm uses as input the public parameter $P = (R, g, Q_1, Q_2, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3)$, signer's identity ID, public value $V = g^r$ and Schnorr zero-knowledge proof values(z, e). Its outputs are valid or invalid as follows.

1) Perform public key validation for V.
2) Let $K_v = g^z V^e$.
3) Let $e_v = H(P, \mathrm{ID}, K_v, V)$.
4) If $e_v = e$, then the signature is verified.

V. PERFORMANCE AND EXPERIMENTAL RESULTS

We used MIRACL Cryptographic SDK [36] to implement our protocol. We implemented and deployed the proposed protocol on two Android devices (Google Nexus 6 with a Quad-core, 2.7GHz processor, 3G bytes memory and the Google Android 7.1.2 operating system; Samsung Galaxy Nexus with a dual-core 1.2GHz processor, 1G bytes memory and the Google Android 4.0 operating system) and a PC with an i7-6700 processor, 8G bytes memory and the Microsoft Windows 7 operating system. As shown in Fig. 6, in our implementation, the two Android phones denote two participants, and the PC represents the KGC.

Table I shows the different security levels of the curves.

In order to evaluate the different security levels, we evaluate the following curves from Type-3 pairings.

1) MNT $k = 6$ curve that achieves AES-80 security.
2) BN $k = 12$ curve that achieves AES-128 security.
3) KSS $k = 18$ curve that achieves AES-192 security.
4) BLS $k = 24$ curve that achieves AES-256 security.

First, we implemented the BLMQ signature scheme on a Samsung Galaxy Nexus, and show the run time of each algorithm for the BLMQ signature scheme in Table II and Fig. 7.

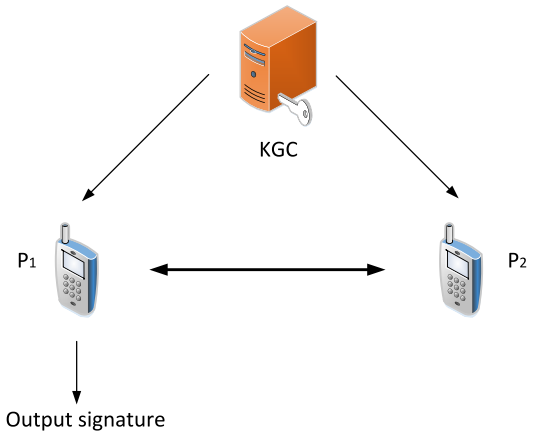


Fig. 6. The model of our proposed protocol.

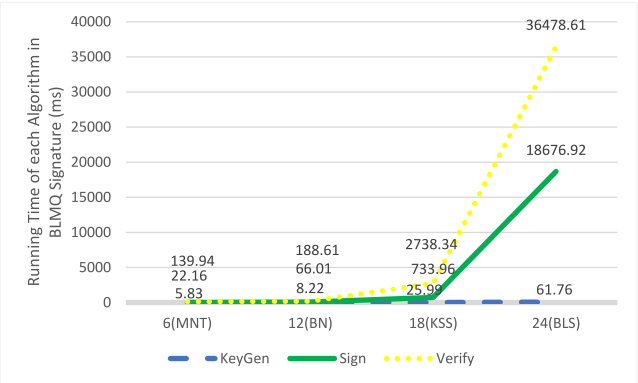


Fig. 7. Run times of each algorithm in BLMQ signature.

Then, we presented each algorithm in the distributed key generation phase. Table III and Fig. 8 show the run times of these algorithms and the verification algorithm.

Furthermore, we analyzed the computation cost of each progress in the distributed signature phase. Table IV and Fig. 9 show the results obtained. Step 1 denotes the progress made by P_1 before P_1 sends a message to P_2, Step 2 denotes the progress made by P_2, and Step 3 denotes the progress made by P_1 after receiving message from P_2. Table V presents the total run times.

TABLE III
DISTRIBUTED KEY GENERATION AND VERIFICATION (AVERAGES WERE COMPUTED OVER 1000 EXECUTIONS)

Curve \ Algorithm	Setup (milliseconds)	Distribute (milliseconds)	Verify (milliseconds)
$k = 6$	145.27	567.42	122.42
$k = 12$	145.92	1107.8	191.17
$k = 18$	2055.02	6211.14	2364.34
$k = 24$	18506.28	8268.26	32116.37

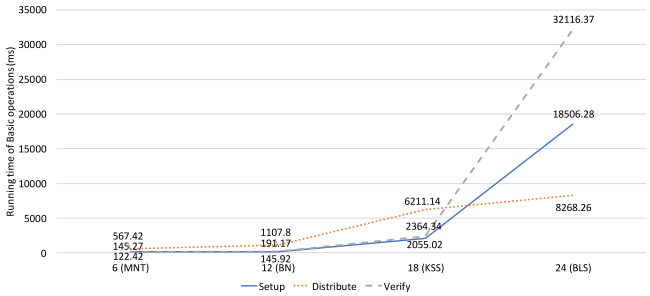


Fig. 8. Run times of basic operations.

TABLE IV
DISTRIBUTED SIGNATURE GENERATION (AVERAGES WERE COMPUTED OVER 1000 EXECUTIONS)

Curve \ Algorithm	Step 1	Step 2	Step 3
$k = 6$	146.03ms	222.43ms	134.49ms
$k = 12$	198.28ms	336.42ms	174.83ms
$k = 18$	1647.47ms	2782.22ms	1505.2ms
$k = 24$	18419.27ms	35972.69ms	17812.87ms

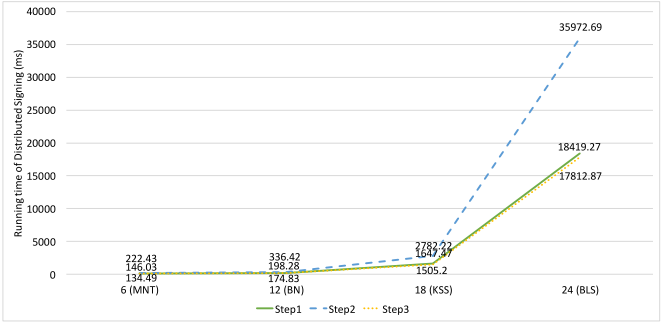


Fig. 9. Run time of distributed signing.

TABLE V
DISTRIBUTED SIGNATURE GENERATION RUN TIME (AVERAGE OVER 1000 EXECUTIONS)

Curve \ Device	P_1	P_2
$k = 6$	280.52ms	222.43ms
$k = 12$	373.11ms	336.42ms
$k = 18$	3152.67ms	2782.22ms
$k = 24$	36232.14ms	35972.69ms

VI. CONCLUSION

The ubiquity of IoT devices will continue in the future, ranging from autonomous vehicles to smart cities to smart military [37], and so on. Thus, ensuring the security of such wireless communications will become even more important and challenging, due to the increasingly complex emerging environment and requirements.

In this paper, we proposed an efficient and secure two-party distributed signing protocol for the identity-based signature scheme in the IEEE P1363 Standard, which is designed to generate a valid signature without recovering the private key. Both the security analysis and the performance evaluation of our proposed protocol have demonstrated its potential to be used for the distributed signature generation in the wireless environment.

In the future, we will implement a prototype of the proposed protocol for evaluating it in a real-world environment. This will allow us to identify areas that may need further refinement or optimization.

REFERENCES

[1] Statista. *Internet of Things (IoT) Connected Devices Installed Base Worldwide From 2015 to 2025 (in Billions)*. Accessed: Nov. 2016. [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

[2] S. Boddy and J. Shattuck. *The Hunt for IoT: The Rise of Thingbots*. Accessed: Aug. 9, 2017. [Online]. Available: https://www.f5.com/labs/articles/threat-intelligence/the-hunt-for-iot-the-rise-of-thingbots

[3] R. J. Robles, T.-H. Kim, D. Cook, and S. Das, "A review on security in smart home development," *Int. J. Adv. Sci. Technol.*, vol. 15, no. 1, pp. 456–461, 2010.

[4] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1933–1954, 4th Quart., 2014.

[5] A. Shamir, "Identity-based cryptosystems and signature schemes," in *Advances in Cryptology—CRYPTO'84* (Lecture Notes in Computer Science), G. R. Blakley and D. Chaum, Eds., vol. 196. Heidelberg, Germany: Springer, Aug. 1984, pp. 47–53.

[6] D. Boneh and M. Franklin, "Identity-based encryption from the weil pairing," in *Advances in Cryptology—CRYPTO 2001*. Heidelberg, Germany: Springer, 2001, pp. 213–229.

[7] K. G. Paterson, "ID-based signatures from pairings on elliptic curves," *Electron. Lett.*, vol. 38, no. 18, pp. 1025–1026, Aug. 2002.

[8] F. Hess, "Efficient identity based signature schemes based on pairings," in *Proc. 9th Annu. Int. Workshop Sel. Areas Cryptography (SAC)*, vol. 2595, Aug. 2003, pp. 310–324.

[9] J. C. Cha and J. H. Cheon, "An identity-based signature from gap Diffie–Hellman groups," in *Proc. 6th Int. Workshop Theory Pract. Public Key Cryptography (PKC)*, vol. 2567. Miami, FL, USA, Jan. 2003, pp. 18–30.

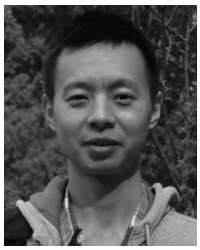
[10] J. Li, J. Li, X. Chen, C. Jia, and W. Lou, "Identity-based encryption with outsourced revocation in cloud computing," *IEEE Trans. Comput.*, vol. 64, no. 2, pp. 425–437, Feb. 2015.

[11] A. Boldyreva, C. Gentry, A. O'Neill, and D. H. Yum, "Ordered multisignatures and identity-based sequential aggregate signatures, with applications to secure routing," in *Proc. ACM 14th Conf. Comput. Commun. Security (CCS)*, Alexandria, VA, USA, Oct. 2007, pp. 276–285.

[12] C. Zhang, R. Lu, X. Lin, P.-H. Ho, and X. Shen, "An efficient identity-based batch verification scheme for vehicular sensor networks," in *Proc. IEEE 27th Conf. Comput. Commun. (INFOCOM)*, Phoenix, AZ, USA, 2008, pp. 246–250.

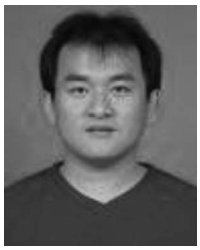
[13] Y. Zhang, Y. Jiang, B. Li, and M. Zhang, *An Efficient Identity-Based Homomorphic Signature Scheme for Network Coding*. Cham, Switzerland: Springer Int., 2018, pp. 524–531, doi: 10.1007/978-3-319-59463-7_52.

[14] 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. Depend. Secure Comput.*, to be published, doi: 10.1109/TDSC.2018.2857775.
[15] N. Sae-Bae and N. Memon, "Online signature verification on mobile devices," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 6, pp. 933–947, Jun. 2014.
[16] D. He, S. Zeadally, and L. Wu, "Certificateless public auditing scheme for cloud-assisted wireless body area networks," *IEEE Syst. J.*, vol. 12, no. 1, pp. 64–73, Mar. 2018.
[17] D. Wang, H. Cheng, D. He, and P. Wang, "On the challenges in designing identity-based privacy-preserving authentication schemes for mobile devices," *IEEE Syst. J.*, vol. 12, no. 1, pp. 916–925, Mar. 2018.
[18] L. Harn and M. Fuyou, "Multilevel threshold secret sharing based on the Chinese remainder theorem," *Inf. Process. Lett.*, vol. 114, no. 9, pp. 504–509, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0020019014000659
[19] J. Kurihara, S. Kiyomoto, K. Fukushima, and T. Tanaka, "A new (k, n)-threshold secret sharing scheme and its extension," in *Proc. 11th Int. Conf. Inf. Security (ISC)*, vol. 5222. Taipei, Taiwan, Sep. 2008, pp. 455–470.
[20] T. Tassa, "Hierarchical threshold secret sharing," *J. Cryptol.*, vol. 20, no. 2, pp. 237–264, Apr. 2007.
[21] A. Shamir, "How to share a secret," *Commun. Assoc. Comput. Mach.*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[22] P. D. MacKenzie and M. K. Reiter, "Two-party generation of DSA signatures," in *Advances in Cryptology—CRYPTO 2001* (LNCS 2139), J. Kilian, Ed. Heidelberg, Germany: Springer, Aug. 2001, pp. 137–154.
[23] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. ACM 13th Conf. Comput. Commun. Security (CCS)*, Alexandria, VA, USA, Oct./Nov. 2006, pp. 89–98.
[24] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Security Privacy*, Berkeley, CA, USA, May 2007, pp. 321–334.
[25] R. Gennaro, S. Goldfeder, and A. Narayanan, "Threshold-optimal DSA/ECDSA signatures and an application to bitcoin wallet security," in *Proc. ACNS 14th Int. Conf. Appl. Cryptography Netw. Security*, vol. 9696. London, U.K., Jun. 2016, pp. 156–174.
[26] Y. Lindell, *Fast Secure Two-Party ECDSA Signing*. Cham, Switzerland: Springer Int., 2017, pp. 613–644, doi: 10.1007/978-3-319-63715-0_21.
[27] TPW Group. (2017). *IEEE P1363 Standard Specifications for Public Key Cryptography*. [Online]. Available: http://grouper.ieee.org/groups/1363/
[28] *IEEE Standard Specifications for Public-Key Cryptography*, IEEE Standard 1363-2000, pp. 1–228, Aug. 2000.
[29] P. S. L. M. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *Advances in Cryptology—ASIACRYPT 2005* (LNCS 3788), B. K. Roy, Ed. Heidelberg, Germany: Springer, Dec. 2005, pp. 515–532.
[30] X. Lin, X. Sun, P. H. Ho, and X. Shen, "GSIS: A secure and privacy-preserving protocol for vehicular communications," *IEEE Trans. Veh. Technol.*, vol. 56, no. 6, pp. 3442–3456, Nov. 2007.
[31] L. Zhou, V. Varadharajan, and M. Hitchens, "Achieving secure role-based access control on encrypted data in cloud storage," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 12, pp. 1947–1960, Dec. 2013.
[32] H. Debiao, C. Jianhua, and H. Jin, "An ID-based proxy signature schemes without bilinear pairings," *Ann. Telecommun.*, vol. 66, nos. 11–12, pp. 657–662, Dec. 2011, doi: 10.1007/s12243-011-0244-0.
[33] C. Hazay and Y. Lindell, "Efficient secure two-party protocols—Techniques and constructions," in *Information Security and Cryptography*. Heidelberg, Germany: Springer, 2010.
[34] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology—EUROCRYPT'99* (LNCS 1592), J. Stern, Ed. Heidelberg, Germany: Springer, May 1999, pp. 223–238.
[35] C.-P. Schnorr, "Efficient signature generation by smart cards," *J. Cryptol.*, vol. 4, no. 3, pp. 161–174, 1991.
[36] *Miracl Library*, Miracl, London, U.K., 2017. [Online]. Available: https://www.miracl.com/
[37] A. Castiglione, K.-K. R. Choo, M. Nappi, and S. Ricciardi, "Context aware ubiquitous biometrics in edge of military things," *IEEE Cloud Comput.*, vol. 4, no. 6, pp. 16–20, Nov./Dec. 2017.



Yudi Zhang received the master's degree from the Hubei University of Technology, Wuhan, China, in 2017. He is currently pursuing the Ph.D. degree at the School of Cyber Science and Engineering, Wuhan University, Wuhan.

His current research interests include cryptography and information security, in particular, cryptographic protocols.



Debiao He received the Ph.D. degree in applied mathematics from the School of Mathematics and Statistics, Wuhan University, Wuhan, China, in 2009.

He is currently a Professor with the School of Cyber Science and Engineering, Wuhan University. His current research interests include cryptography and information security, in particular, cryptographic protocols.

Sherali Zeadally received the bachelor's degree in computer science from the University of Cambridge, Cambridge, U.K., and the Doctorate degree in computer science from the University of Buckingham, Buckingham, U.K.

He is an Associate Professor with the College of Communication and Information, University of Kentucky, Lexington, KY, USA.

Dr. Zeadally is a Fellow of the British Computer Society and the Institution of Engineering Technology, U.K.

Ding Wang received the Ph.D. degree in information security from Peking University, Beijing, China, in 2017.

He is currently supported under the Boya Post Doctoral Fellowship from Peking University. He has authored over 40 papers at venues such as ACM CCS and in publications such as the IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, and his papers get over 1200 citations. His current research interests include password-based authentication and provable security.

Dr. Wang was a recipient of the Top-10 Distinguished Graduate Academic Award from Harbin Engineering University in 2013 and Peking University in 2016.

Kim-Kwang Raymond Choo (SM'15) received the Ph.D. degree in information security from the Queensland University of Technology, Brisbane, QLD, Australia, in 2006.

He currently holds the Cloud Technology Endowed Professorship at The University of Texas at San Antonio (UTSA), San Antonio, TX, USA, and has a courtesy appointment at the University of South Australia. In 2016, he was named the Cybersecurity Educator of the Year - APAC (Cybersecurity Excellence Awards are produced in cooperation with the Information Security Community on LinkedIn), and in 2015 he and his team won the Digital Forensics Research Challenge organized by Germany's University of Erlangen–Nuremberg.

Dr. Choo was the recipient of the 2018 UTSA College of Business Col. Jean Piccione and Lt. Col. Philip Piccione Endowed Research Award for Tenured Faculty, IEEE TrustCom 2018 Best Paper Award, ESORICS 2015 Best Research Paper Award, 2014 Highly Commended Award by the Australia New Zealand Policing Advisory Agency, Fulbright Scholarship in 2009, 2008 Australia Day Achievement Medallion, and British Computer Society's Wilkes Award in 2008. He is also a Fellow of the Australian Computer Society.