Balancing Privacy and Accountability for Industrial Mortgage Management

Liang Xue, Dongxiao Liu, *Student Member, IEEE*, Jianbing Ni, *Member, IEEE*, Xiaodong Lin, *Fellow, IEEE*, and Xuemin (Sherman) Shen, *Fellow, IEEE*

***Abstract*—Industrial mortgage enables companies to acquire loan for business venture or investment purposes by pledging their industrial assets to financial institutions. To prevent double-mortgage fraud of borrowers, information exchange among different financial institutions is necessary. On the other hand, it results in the privacy leakage of borrowers. In this article, we construct a blockchain-based accountable and privacy-preserving industrial mortgage scheme (BAPIM). BAPIM enables financial institutions to share the mortgage data of borrowers in an efficient and secure manner, that achieves the borrower identity privacy and accountability at the same time. Specifically, borrower identity is concealed on the blockchain by anonymous identity credential, while financial institutions can still uncover the identity of a misbehaving borrower if he pledges the same asset for multiple mortgages. We demonstrate that BAPIM achieves the desirable security properties and has high computational efficiency, so as to be suitable for the industrial mortgage management.**

***Index Terms*—Blockchain, industrial Internet, industrial mortgage, mortgage fraud, privacy preservation.**

I. Introduction

INDUSTRIAL Internet is reshaping the global industrial landscape in a wide range of industrial processes, such as investment, manufacturing, transportation, and sales [1], [2]. With the seamless connectivity and advanced data analytical technologies, industrial internet significantly boosts process management efficiency and promotes the development of innovative industrial manufacturing and products [3], [4]. The prevalence of the industrial Internet also makes the industrial mortgage management more intelligent, where the status and transfers of the industrial assets can be timely regulated. Industrial mortgage, also called commercial mortgage, enables borrowers to pledge valuable industrial assets, such as manufacturing plants, factories, warehouses, storage units, and industrial products to the financial initiations to acquire a certain amount of loans. Compared with unsecured loan, borrowers obtain loans with a lower interest in a mortgage loan [5]. The borrowers can use the mortgage loan to make new investment and expand the scalability of their businesses. When a borrower defaults on the loan or otherwise fails to abide by the loan items, the lender takes possession of the mortgages for paying off the debt.

However, an industrial mortgage loan allows the borrower to retain the ownership of the valuable industrial asset. It is possible that a dishonest borrower makes a repeated mortgage to acquire more money for investment. That is, the borrower uses the same asset as collateral for multiple financial institutions to defraud the lender of funds, which is denoted as the double-mortgage fraud and leaves lenders exposed to huge financial risks. A straightforward solution for the fraud is to establish a centralized database that records every mortgage case and identifies the double-mortgage misbehavior [6]. The solution may not work for the industrial mortgage management because of the following two reasons: 1) There is lack of mutual trust among financial institutions, especially for small-loan companies, to agree on the correctness and reliability of a centralized party [7]; 2) the mortgages may contain sensitive personal or enterprise-level information and direct share among different parties is restricted by regulations, such as general data protection regulation (GDPR) in Europe [8]. As a result, how to design a mechanism that quickly detects and blacklists malicious double-mortgage borrowers while protecting honest borrowers privacy is a challenging issue.

Blockchain technology is envisioned to promote universal trust among industrial partners and increase operational efficiency for business process management [7]. As the enabling technologies that support Bitcoin, blockchain is actually a decentralized ledger that can be publicly validated. The ledger consists of an increasing number of blocks of transactions. The consensus mechanism of blockchain ensures the consistent view of the public ledger and prevents the adversaries from deleting or modifying the appended data on the chain. As a trusted distributed database, blockchain technology provides a way of recording currency transactions or any other digital information that is designed to be transparent, auditable, highly resistant to outages [9]. Currently, the blockchain applications span across diverse industrial fields far beyond cryptocurrencies, which include supply chain, insurance, economics, Internet of Things,

Manuscript received July 3, 2019; revised September 26, 2019; accepted October 17, 2019. Date of publication November 8, 2019; date of current version February 28, 2020. Paper no. TII-19-2922. *(Corresponding author: Jianbing Ni.)*

L. Xue, D. Liu, and X. (Sherman) Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: liang.xue@uwaterloo.ca; dongxiao.liu@uwaterloo.ca; sshen@uwaterloo.ca).

J. Ni is with the Department of Electrical and Computer Engineering, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: jianbing.ni@queensu.ca).

X. Lin is with the School of Computer Science, University of Guelph, Guelph, ON N1G 2W1, Canada (e-mail: xlin08@uoguelph.ca).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2019.2952372

etc. [10]–[12]. However, simply putting the records of mortgage on the blockchain and comparing them for double-mortgage detection does not solve the problem since loan companies may not be willing to publish their data in a public manner. Moreover, even if fraudulent behavior is discovered, the anonymity nature of the blockchain prevents the misbehaving borrower's true identity from being revealed.

To address the issue, we construct a Blockchain-based accountable and Privacy-preserving Industrial Mortgage scheme (BAPIM), which not only protects the privacy of honest borrowers, but also helps financial companies or financial institutions to detect the misbehavior of borrowers. When fraudulent behavior is detected, the true identity of the malicious borrower can be revealed publicly. BAPIM is built based on blockchain and double authentication preventing signature to achieve secure and reliable industrial mortgage loan. The main contributions of BAPIM are two folds.

1) A blockchain-based accountable industrial mortgage scheme is designed to prevent a borrower from using the same asset as collateral to obtain multiple loans from different financial institutions. By taking advantages of the transparency and irreversibility of blockchain technology, BAPIM can help financial institutions efficiently identify the double-mortgage fraudulent behavior of a borrower and reduce the financial risks.
2) The extractability of a malicious borrower's identity is achieved by integrating ElGamal encryption, zero-knowledge proof and verifiable secret sharing. Financial institutions can anonymously authenticate the honest borrowers during the process of mortgage. Only when the double-mortgage behavior is discovered, the greedy borrower's private key can be extracted and the identity can be revealed by the financial institution. The security model of BAPIM is defined, and BAPIM is proven to be able to fulfill all the desirable security objectives under the security model.

The remainder of this article is organized as follows. In Section II, we describe the system model, security goals, system components, and the security model. In Section III, we present the preliminaries. The detailed BAPIM is given in Section IV. We give the correctness analysis and security analysis in Section V and show the performance evaluation in Section VI. The related work is presented in Section VII. Finally, Section VIII concludes this article.

II. Problem Formulation

In this section, we first present the system model, and analyze the security threats and the goals that need to be achieved in the system. Then, we define the system components and security model of BAPIM.

A. System Model

In industrial mortgage, borrowers usually mortgage their valuable assets, such as manufacturing plants, factories, warehouses, and industrial products, to obtain loans from financial institutions. Borrowers can be individuals who are going to start an active business venture, or companies that are willing to make new investment or expand the business, but do not have sufficient money for investment. Thus, they turn to the financial institutions for financing and get loans. In an industrial mortgage loan, the loan is secured by the industrial asset in the borrower's name. If the borrower wants to get a loan from a financial institution, the institution first checks the ownership of the industrial asset and evaluates the value. If the institution believes the value is higher than the amount of money that the borrower would like to get and the risk is low, the financial institution would provide the loan to the borrower. If the borrower fails to repay the loan in full, the institution can seize and sell the pledged asset to recover any outstanding balance.

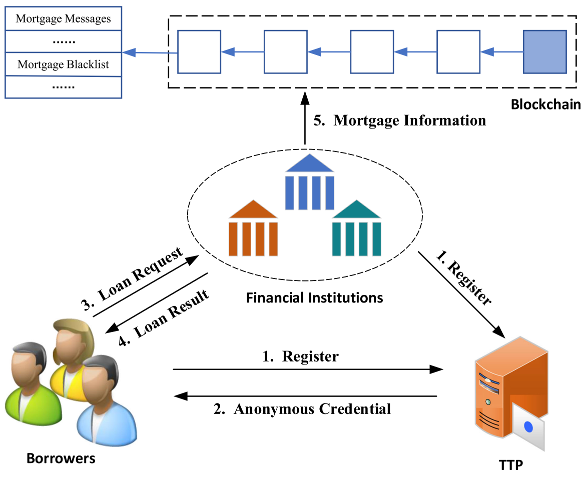


Fig. 1. System Model.

As shown in Fig. 1, there are three entities in our system: 1) borrowers, 2) financial institutions, and 3) a trust third party (TTP).

1) *Borrowers:* Borrowers can mortgage their valuable assets to acquire loans from financial institutions.
2) *Financial Institutions:* A financial institution can provide loans to borrowers if the loan request of a borrower is verified. Financial institutions also upload the records of mortgage information to the public blockchain for fraudulent behavior detection.
3) *TTP:* It is a trust entity that is responsible for issuing anonymous identity credentials to borrowers when borrowers register the industrial mortgage service. TTP is also in charge of certifying a borrower's legal ownership of an asset.

At a high level, BAPIM works as follows. Borrowers and financial institutions first register themselves to TTP. For a borrower, after sending its identity and public key to the TTP, it obtains an anonymous identity credential from TTP. When a borrower plans to apply for a mortgage loan from a financial institution, the borrower generates a loan request and sends it to the financial institution. After verifying the validity of the application, the financial institution offers the mortgage loan to the borrower and uploads the mortgage message to the blockchain. If the financial institution discovers the behavior of double mortgage, it can extract the identity of the malicious borrower and add the identity and the proof of the misbehavior to the mortgage blacklist maintained by the blockchain.

B. Security Threats and Goals

In mortgage loans, borrowers are unwilling to disclose their mortgaged assets to others. In addition, financial institutions need to verify the legality of loan requests, since malicious borrowers may apply for loans in the name of others. Also, a greedy borrower may pledge the same asset to different financial institutions to get repeated loans without being noticed. Hence, double mortgage behavior should be detected among the financial institutions while the identities and pledged assets of honest borrowers are not revealed. Therefore, for the purpose of achieving a privacy-preserving and accountable mortgage loan, three security goals should be achieved.

Anonymity: A borrower can apply for a mortgage loan without revealing its true identity, which means each financial institution does not know the identity of an honest borrower.

Unforgeability: An attacker cannot forge a legitimate loan request that can pass the verification, which means whether a borrower has the ownership of an asset and the signature of a loan request is valid can be verified by the financial institution.

Accountability: If the behavior of double mortgage is discovered, the financial institution is aware of the greedy borrower and can extract the borrower's identity without the assistance of TTP.

C. System Components

BAPIM consists of the following algorithms, namely, KGen, Register, ReqGen, ReqVerify, and Extract.

1) *KGen(λ):* Given a security parameter λ, the algorithm outputs a secret key sk and a public key pk. For simplicity, we use $(SK_b, PK_b), (SK_c, PK_c)$, and (SK_T, PK_T) to denote the secret-public key pair for a borrower, a financial institution, and TTP, respectively.
2) *Register(ID, PK_b, σ_b, SK_T):* Given the (ID, PK_b) of a borrower, the signature σ_b on (ID, PK_b) and the secret key SK_T of TTP, the algorithm outputs an anonymous identity certificate Crt_1 of the borrower.
3) *ReqGen(A, SK_b, PK_c):* Given a mortgage asset A, SK_b, and PK_c, the algorithm outputs an $m = (H(A), PK_c)$ and a loan request Req, where H is a cryptographic hash function.
4) *ReqVerify(PK_b, PK_T, Req):* Given the PK_b, PK_T and a loan request Req, the algorithm outputs a bit $b \in \{0, 1\}$.
5) *Extract($m_1, m_2, \sigma_1, \sigma_2$):* Given two mortgage messages (m_1, m_2) with the same asset but different financial institutions and their signatures (σ_1, σ_2), the algorithm outputs a secret key SK_b and an identity ID.

D. Security Model

In industrial mortgages, the security requirements include anonymity, unforgeability, and accountability. The game-based approach is employed to formally define the security model of BAPIM. The adversary can make queries to oracles that are defined in the games.

Definition 1 (GAME Anonymity): A challenger $\mathfrak{C}$ and an adversary $\mathfrak{A}$ are involved in the interactive game, which defines the property of anonymity. The details of the game are shown as below.

Initialization: Given a security parameter λ, $\mathfrak{C}$ runs *KGen* to generate two public–private key pair (pk_b, sk_b) and (pk_T, sk_T). $\mathfrak{C}$ keeps (sk_b, sk_T) itself and sends (pk_b, pk_T) to $\mathfrak{A}$.

Query: $\mathfrak{A}$ can make a polynomial number of queries to the Register oracle. When $\mathfrak{A}$ makes a registration query on an ID, the Register oracle encrypts the ID with pk_b and obtains the ciphertext $\overline{C}$, then it signs the ciphertext using sk_T to generate the signature σ. $\mathfrak{C}$ returns $(\overline{C}, \sigma)$ to $\mathfrak{A}$.

Challenge: $\mathfrak{A}$ chooses two IDs I_0, I_1 with the same length and sends them to $\mathfrak{C}$. $\mathfrak{C}$ chooses a random bit $b \in \{0, 1\}$ and encrypts the ID I_b with pk_b. Then, $\mathfrak{C}$ signs the ciphertext C^* and sends the signature σ^* and C^* to $\mathfrak{A}$.

Guess: For the I_b which is encrypted by $\mathfrak{C}$, $\mathfrak{A}$ outputs a guess $b' \in \{0, 1\}$.

$\mathfrak{A}$ wins the game if the guess of $\mathfrak{A}$ is correct. The advantage of $\mathfrak{A}$ in this game is defined as $\Pr[b' = b] - \frac{1}{2}$.

We say that BAPIM scheme achieves anonymity if the polynomial time adversary $\mathfrak{A}$ has at most a negligible advantage in the GAME Anonymity.

Definition 2 (GAME Unforgeability): The GAME unforgeability is an interactive game where $\mathfrak{C}$ and $\mathfrak{A}$ are involved. This game defines the property of unforgeability and details are shown as below.

Initialization: Given a security parameter λ, $\mathfrak{C}$ runs **KGen** to generate a private key sk_b and a public key pk_b. $\mathfrak{C}$ keeps sk_b private and sends pk_b to $\mathfrak{A}$. Moreover, $\mathfrak{C}$ initializes two empty sets $\mathcal{Q}$ and $\mathcal{R}$.

Query: $\mathfrak{A}$ can perform a polynomial number of signature queries to the Sign oracle. When $\mathfrak{A}$ makes a signature query on m, the Sign oracle first parses m as (a, c). If $a \in \mathcal{R}$, the oracle returns failure. Otherwise, it generates a signature σ on m, and returns it to $\mathfrak{A}$. Then, m is added to the set $\mathcal{Q}$ and a is added to the set $\mathcal{R}$.

Output: $\mathfrak{A}$ outputs a forged signature (m^*, σ^*).

$\mathfrak{A}$ wins the above game if σ^* is a valid signature of m^* and $m^* \notin \mathcal{Q}$.

We say that BAPIM scheme achieves unforgeability under chosen message attack if no polynomial adversary $\mathfrak{A}$ can win the GAME Unforgeability with a nonnegligible probability.

Definition 3 (GAME Accountability): GAME Accountability is an interactive game between $\mathfrak{C}$ and $\mathfrak{A}$. This game defines the property of accountability and the details are shown as below.

Initialization: Given a security parameter λ, $\mathfrak{C}$ runs **KGen** to generate a private key sk_b and a public key pk_b. Then, $\mathfrak{C}$ sends pk_b and sk_b to $\mathfrak{A}$.

Output: Based on pk_b and sk_b, $\mathfrak{A}$ outputs two messages and their respective signatures $(m_1, \sigma_1, m_2, \sigma_2)$, where $m_i = (a_i, c_i)$, $i \in [2]$ and for the two messages, $a_1 = a_2$ and $c_1 \neq c_2$.

$\mathfrak{A}$ wins the above game if: 1) For $i \in [2]$, σ_i is a valid signature of m_i; 2) $\mathbf{Extract}(pk_b, m_1, \sigma_1, m_2, \sigma_2)$ returns an $sk_b^{'}$, but $sk_b^{'} \neq sk_b$.

We say that BAPIM scheme achieves accountability if the polynomial time adversary $\mathfrak{A}$ can win the GAME Accountability with at most a negligible probability.

III. Preliminaries

This section reviews the preliminaries that are used to design BAPIM.

A. ECDSA

Elliptic curve digital signature algorithm (ECDSA) [13] belongs to the elliptic curve cryptosystems. The algorithms in ECDSA are described as below.

ECDSA.KGen(1^λ): Let G be an elliptic curve group chosen from a security parameter λ. G is equipped with the order q and a generator g. The algorithm chooses a collision-resistant hash function $H : \{0,1\}^* \rightarrow Z_q$. x is a random number in Z_q^*, which is set to be the secret key sk. The corresponding public key is $pk = g^x$.

ECDSA.Sign(sk, m): Given a message m that needs to be signed, a signer chooses a $k \in Z_q^*$ randomly, and computes $R = g^k$. Denote the x coordinate of R to be r. The signer calculates $s = k^{-1}(H(m) + rx) \bmod q$. This algorithm outputs $\sigma = (r, s)$ as the signature of m.

ECDSA.Verify(σ, m, pk): Given a signature σ on m, the verifier first parses the signature as $\sigma = (r, s)$, and computes $v = H(m)$ and $w = s^{-1} \bmod q$. Then, it calculates $u_1 = vw \bmod q$, $u_2 = rw \bmod q$ and $R = g^{u_1} \cdot pk^{u_2}$. If $R = r \bmod q$ holds, the signature is a valid signature and the verifier outputs 1. Otherwise, it outputs 0.

B. Verifiable Secret Sharing

The (k, n) threshold secret sharing scheme [14] can be utilized to distribute a secret s among n participants. The share each party owns is an evaluation of a polynomial $f(X) = \xi_{k-1}X^{k-1} + \cdots + \xi_1 X + s$.

For a party i, $i \in [n]$, the share can be calculated as $f(X_i)$. Given any k different shares, which provide k different points (x_i, y_i), $1 \leq i \leq k$, the secret s can be recovered as

$$s = \sum\nolimits_{i=1}^{k} \rho_i y_i, \quad \text{where} \quad \rho_i = \prod\nolimits_{j=1, j\neq i}^{k} \frac{-x_j}{x_i - x_j}.$$

Shamir's secret sharing can be publicly verifiable by utilizing the technique proposed by Feldman [15]. Verifiability means one can verify that a share is correctly generated. Let G be a multiplicative group which is equipped with the order q and a generator g. The basic idea of verifiability is to publish a sequence $(g^{\xi_{k-1}}, \ldots, g^{\xi_1}, g^{\xi_0})$, where $\xi_{k-1}, \ldots, \xi_1, \xi_0$ are coefficients of the polynomial $f(X)$ and $g^{\xi_0} = g^s$. Given a share y_i and its corresponding x_i, one can verify that the share is correct by checking whether the equation $g^{y_i} = \prod_{j=0}^{k-1} (g^{\xi_j})^{x_i^j}$ holds.

C. Σ-Protocols

A Σ-protocol [17], which belongs to a zero-knowledge proof of knowledge (ZKPoK) protocols [16], is an interactive three-move (commitment, challenge, response) protocol. In a Σ-protocols, given a statement x, a prover can prove that it knows a witness w which satisfies a relationship $R(x, w) = 1$.

TABLE I
ZKPoK for a DDH Tuple

Let $g_1, g_2, u_1, u_2 \in G$		
Prover $(u_1, u_2, k = log_{g_i} u_i)$		Verifier (u_1, u_2)
$r \leftarrow Z_q^*, r_i = g_i^r$	$\xrightarrow{r_1, r_2}$	
	$\xleftarrow{b}$	$b \overset{R}{\leftarrow} Z_q$
$v \leftarrow r + kb$	$\xrightarrow{v}$	accept iff $\forall i : g_i^v = r_i u_i^b$

In the Σ-protocol, the relationship R that a tuple of elements (g_1, g_2, u_1, u_2) forms a DDH tuple is presented as $(g_1, g_2, u_1, u_2, w) \in R \Leftrightarrow u_1 = g_1^w \wedge u_2 = g_2^w$, where (g_1, g_2, u_1, u_2) are in a cyclic group G. The proving process of a Σ-protocol for the DDH tuple is shown in Table I.

By utilizing the Fiat-Shamir transformation [18], ZKPoK can be converted to a noninteractive zero-knowledge proof (NIZKP) protocol [19]. The interaction between the two parties is removed by making use of a collision-free hash function to obtain the challenge. Compared with the interactive ZKPoK, the NIZKP protocol outputs a common reference string in the initial phase.

An NIZKP consists of three algorithms which are described as below.

NIZKP.Setup(1^k): Given a security parameter k, this algorithm generates a common reference string crs.

NIZKP.Proof(crs, x, w): This algorithm is run by the prover to generate a proof of the statement x. Given the crs and a witness w, the algorithm can output a proof π.

NIZKP.Verify(crs, x, π): This algorithm is run by the verifier to verify whether the statement x is true. Given the statement x, the proof π and crs, the algorithm can output a bit $b \in \{0, 1\}$.

An NIZKP protocol Π is zero-knowledge, if there exist an efficient simulator $S = (S_1, S_2)$ such that

$$|\Pr[crs \leftarrow Setup_{\Pi}(1^k) : A(crs) = 1] -$$
$$\Pr[(crs, \tau) \leftarrow S_1(1^k) : A(crs) = 1]| \leq \varepsilon_1,$$

where ε_1 is a negligible number, and the probability that A wins the experiment Zero-Knowledge$_{A,S}^{\Pi}(k)$, which is shown in Fig. 2, is also negligible, which means that

$$|\Pr[Zero-Knowledge_{A,S}^{\Pi}(k) = 1] - \frac{1}{2}| \leq \varepsilon_2,$$

where ε_2 is a negligible probability.

IV. Proposed BAPIM

In this section, we first give the overview of BAPIM and then provide the details of BAPIM.

A. Overview of BAPIM

KGen: It is the key generation algorithm that is run by borrowers, financial institutions and TTP. To be specific, given a security parameter λ, a borrower can generate its secret-public key pair (SK_b, PK_b), and a financial institution can generate (SK_c, PK_c) as its secret-public key pair. TTP can also create its secret-public key pair (Sk_T, Pk_T).

```
Experiment $Zero-Knowledge^{\Pi}_{A,S}(k)$
    $b \leftarrow \{0,1\}$
    $(crs, \tau) \leftarrow S_1(1^k)$
    $b^* \leftarrow A^{P_b(\cdot,\cdot)}(crs)$
        where oracle $P_0$ on input $(x, \omega)$
            Return $\pi \leftarrow Proof_{\Pi}(crs, x, \omega), if (x, w) \in R$
            Return $\perp$
        and oracle $P_1$ on input $(x, \omega)$
            Return $\pi \leftarrow S_2(crs, \tau, x), if (x, w) \in R$
            Return $\perp$
    return 1, if $b = b^*$
    return 0
```

Fig. 2. Zero-knowledge experiment.

Register: It is the register algorithm that is run by borrowers, financial institutions and TTP. A financial institution sends its name and public key to TTP such that borrowers can get the authenticated public key of a financial institution from TTP. A borrower registers with TTP by sending its identity ID, public key PK_b and the signature σ_b on (ID, PK_b) to TTP. If ID and PK_b are valid, TTP encrypts ID with the borrower's PK_b and the ciphertext is denoted by D. Then, TTP signs $T_1 = (PK_b, D)$, the resulting signature is denoted by σ_{T_1}. Finally, TTP sends the anonymous identity credential $Crt_1 = (T_1, \sigma_{T_1})$ to the borrower.

ReqGen: It is a loan request generation algorithm run by a borrower and TTP. When a borrower plan to apply for a mortgage loan from a financial institution, it first sends its PK_b, ID and the ownership certificate of an asset to TTP, who will generate a $T_2 = (PK_b, A)$, where A represents the asset of the borrower. TTP also generates a signature on T_2, which is denoted by σ_{T_2}. Then, TTP sends $Crt_2 = (T_2, \sigma_{T_2})$ to the borrower. The borrower creates a mortgage message $m = (H(A), PK_c)$, and generates its signature σ_m, where H is a hash function and PK_c is the public key of the financial institution. Finally, the borrower sends the loan request $Req = (m, \sigma_m, Crt_1, Crt_2)$ to the financial institution.

ReqVerify: It is a loan request verification algorithm run by a financial institution. After receiving a loan request, the financial institution first checks the validation of Crt_1 and Crt_2 by using the public key PK_T. Then, the financial institution checks whether Crt_1 is on the blacklist. If not, it verifies whether the signature σ_m of the message m is valid. If it is valid, the financial institution compares the mortgage message m with the previous mortgage messages on the blockchain that are uploaded by different financial institutions. If there is no active loan related to the asset A, the financial institution returns a success. Then, the financial institution generates a signature σ_M of $M = (m, \sigma_m, t_m)$, where t_m denotes the loan term, and uploads (M, σ_M, Crt_1) to the blockchain.

Extract: It is an identity extract algorithm that is run by a financial institution. Given the public key pk_b and two messages (m_1, m_2) with the same asset but different institutions and their corresponding signatures $(\sigma_{m_1}, \sigma_{m_2})$, the financial institution can extract the secret key of the borrower. Based on D in the Crt_1 and the secret key, the financial institution can obtain the true identity of the borrower.

B. Detailed BAPIM

The ECDSA signature, ZKPoK, and ElGamal encryption are employed to generate a loan request. To guarantee the extractability of the identity in case of misbehavior, when the borrower generates the signature of a mortgage message, the threshold secret sharing technique is utilized to generate a share of the secret key, and the share is embedded in the signature. To be specific, the borrower generates a polynomial of degree 1, where the constant term is the secret key of the borrower. The input for creating a share is the public key of a financial institution. Moreover, the borrower needs to prove that the share is correctly generated by using ZKPoK and verifiable secret sharing. When two mortgage messages contain the same asset but different institutions, one can recover the secret key and identity of the borrower by utilizing the shares in the two signatures.

The details of the proposed BAPIM are described as follows.

KGen: According to a security parameter λ, TTP chooses an elliptic curve group G, which is equipped with a prime order q and a generator g. TTP also chooses a hash function $H : \{0,1\}^* \rightarrow Z_q$ and a hash function $h : G \rightarrow Z_q^*$. A borrower first chooses a random $sk_s \in Z_q^*$ and $x_E \in Z_q^*$, and sets $pk_s = g^{sk_s}$ and $pk_E = g^{x_E}$. The key pair (sk_s, pk_s) is used to generate and verify ECDSA signatures, and (pk_E, x_E) is used for ElGamal encryption and decryption. Let n be the number of assets that a borrower can mortgage. The borrower chooses $2n$ random numbers $a_i \leftarrow Z_q^*, i \in [n]$ and $r_i \leftarrow Z_q^*, i \in [n]$. $\{a_i\}_{i\in[n]}$ are used as the coefficients of the polynomials of degree 1. The borrower computes $F_i = (g^{r_i}, pk_E^{r_i} g^{a_i})$, where $i \in [n]$ and gets crs by invoking **NIZPK.Setup** algorithm for the DDH tuple. The secret key of the borrower is $SK_b = (sk_s, (a_i, r_i)_{i\in[n]})$. The public key of the borrower is $PK_b = (pk_s, pk_E, (F_i)_{i\in[n]}, crs)$.

A financial institution chooses a random $SK_c \in Z_q^*$ and computes $PK_c = g^{SK_c}$. The signing key pair of the financial institution is (SK_c, PK_c).

TTP also generates its signing key pair by choosing a random $SK_T \in Z_q^*$ and computing $PK_T = g^{SK_T}$. The secret key of TTP is SK_T, and the public key of TTP is PK_T. The system parameters are shown in Table II.

Register: A financial institution sends its name and public key to TTP for borrowers' retrieval. When a borrower registers with TTP, the borrower sends its ID, public key PK_b and the signature σ_b on (ID, PK_b) to TTP, where σ_b is obtained by invoking **ECDSA.Sign** algorithm. After verifying ID and PK_b of the borrower, TTP chooses a random $\beta \in Z_q^*$. Then, it encrypts ID with the borrower's public key pk_s. The ciphertext is computed as $D_1 = g^{\beta}$, and $D_2 = ID \cdot pk_s^{\beta}$, where $ID \in G$. The ciphertext of ID is $D = (D_1, D_2)$. Then, TTP signs $T_1 = (PK_b, D)$ and gets the signature $\sigma_{T_1} = (r_{T_1}, s_{T_1})$ by invoking **ECDSA.Sign** algorithm. Finally, TTP returns $Crt_1 = (T_1, \sigma_{T_1})$ to the borrower.

TABLE II
SYSTEM PARAMETERS

Acronym	Definition
pk_s	Public key of a borrower for signature verification
sk_s	Private key of a borrower for signature generation
pk_E	Public key of a borrower for encryption
sk_E	Private key of a borrower for decryption
PK_b	Public key of a borrower
SK_b	Private key of a borrower
PK_T	Public key of TTP
SK_T	Private key of TTP
PK_c	Public key of a financial institution
SK_c	Private key of a financial institution
H	Cryptographic hash function
a_i	The coefficient of a polynomial of degree 1
F_i	Ciphertext of g^{a_i}

ReqGen: When a borrower plan to pledge an industrial asset A_i with $i \leq n$ to obtain a loan from a financial institution C, where $A_i \in \{0,1\}^*$, it first sends its PK_b, ID and the ownership certificate of A_i to TTP, who will generate a signature on $T_2 = (PK_b, A_i)$ by invoking **ECDSA.Sign** algorithm, and the resulting signature is denoted by $\sigma_{T_2} = (r_{T_2}, s_{T_2})$. Then, TTP sends $Crt_2 = (T_2, \sigma_{T_2})$ to the borrower.

After receiving Crt_2, the borrower generates the mortgage message $m = (H(A_i), PK_c)$, where PK_c is the public key of the financial institution. To sign m, the borrower first chooses a random $k \in Z_q^*$ and computes $R = g^k$. We denote the x coordinate of R by η. The borrower computes $s = k^{-1}(H(m) + \eta \cdot sk_s) \bmod q$ and $z = a_i \cdot h(PK_C) + sk_s$. After that, the borrower calculates $F'_{i,2} = F_{i,2} \cdot (pk_s \cdot g^{-z})^{\frac{1}{h(PK_C)}}$, and obtains π by invoking the NIZKP.Proof algorithm for the DDH tuple with the input $(crs, r_i, (g, pk_E, F_{i,1}, F'_{i,2})$. The signature of the mortgage message m is $\sigma_m = (\eta, s, z, \pi)$.

After that, the borrower sends the loan request $Req = (m, \sigma_m, Crt_1, Crt_2)$ to the financial institution.

ReqVerify: After receiving a loan request Req, a financial institution first verifies the validity of Crt_1 and Crt_2 by invoking the **ECDSA.Verify** algorithm with the input PK_T. Then, the financial institution checks whether Crt_1 is on the blacklist. If not, the financial institution verifies the correctness of the signature σ_m. The verification process is as follows.

The financial institution C first computes $v = H(m)$ and $w = s^{-1} \mod q$. Then, it calculates $u_1 = vw \mod q$, $u_2 = rw \mod q$, and $R = g^{u_1} pk_s^{u_2}$. After that, C checks whether $R_x = \eta$ holds. If this is not true, C aborts and returns failure. Otherwise, C computes $F'_{i,2} = F_{i,2} \cdot (pk_s \cdot g^{-z})^{\frac{1}{h(PK_C)}}$, and verifies that the share z in the signature is generated correctly by invoking **NIZKP.Verify** algorithm for DDH tuple with the input $(crs, (g, pk_E, F_{i,1}, F'_{i,2}), \pi)$. If the returned result is 1, the validity of (m, σ_m) is proved.

Then, C compares the mortgage message m with the previous messages which are uploaded to the blockchain by different financial institutions. If the first part of the mortgage message $H(A_i)$ is different from the other messages, C signs $M = (m, \sigma_m, t_m)$ by invoking **ECDSA.Sign** with the input SK_C. The obtained signature is denoted by σ_M. Finally, the financial institution uploads (M, σ_M, Crt_1) to the Blockchain and returns success to the borrower.

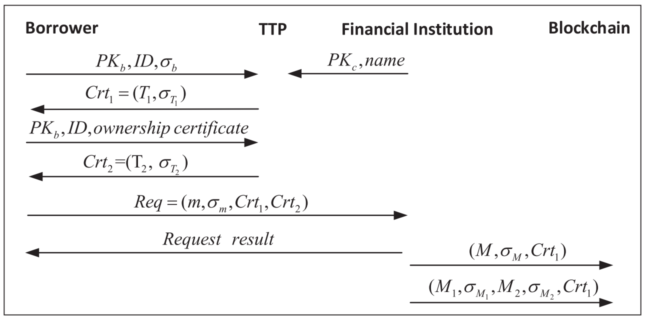

Borrower
TTP
Financial Institution
Blockchain
PK_b, ID, σ_b
$PK_c, name$
$Crt_1 = (T_1, \sigma_{T_1})$
$PK_b, ID, ownership\ certificate$
$Crt_2 = (T_2, \sigma_{T_2})$
$Req = (m, \sigma_m, Crt_1, Crt_2)$
Request result
(M, σ_M, Crt_1)
$(M_1, \sigma_{M_1}, M_2, \sigma_{M_2}, Crt_1)$

Fig. 3. Interaction process of BAPIM.

Extract: If a financial institution C discovers a mortgage message whose pledged asset is same with a previous mortgage message that is still within the term of the loan, indicating that the double-mortgage fraudulent behavior is detected, the financial institution can extract the secret key of the borrower based on the two signatures $(m_1, m_2, \sigma_{m_1}, \sigma_{m_2})$. The secret key can be calculated as

$$sk_s = z_1 \frac{h(PK_{C_2})}{h(PK_{c_2}) - h(PK_{c_1})} + z_2 \frac{h(PK_{c_1})}{h(PK_{c_1}) - h(PK_{c_2})}.$$

After recovering the secret key sk_s of the borrower, C can decrypt the ciphertext D with the sk_s, and obtains the identity of the borrower. Finally, C adds the $\{M_1, M_2, \sigma_{M_1}, \sigma_{M_2}, Crt_1\}$, which can be seen as the proof of the misbehavior, and the identity to the mortgage blacklist maintained by the blockchain. The interaction process of BAPIM is shown in Fig. 3.

If the repayment of a mortgage loan is finished and the borrower needs to pledge the same asset for the second time, the borrower can generate a new public–private key pair and register with TTP. Note that the new key pair is only used for the mortgage of that asset. In general, for a borrower, there is only one public–private key pair corresponding to its ID. For the simultaneous double-mortgage, financial institutions can periodically check mortgage messages on the blockchain to detect the misbehavior.

V. CORRECTNESS AND SECURITY ANALYSIS

In this section, we first show the correctness of BAPIM, then demonstrate that BAPIM achieves the properties of anonymity, unforgeability, and accountability under the security model.

A. Correctness Analysis

The correctness of the scheme can be elaborated as follows. First, a financial institution can verify the share z is correctly generated by calculating:

$$\begin{aligned} z &= a_i \cdot h(PK_c) + sk_s \\ g^z &= g^{a_i \cdot h(PK_c)} \cdot pk_s \\ (g^{a_i})^{-1} &= (g^{-z} \cdot pk_s)^{\frac{1}{h(PK_c)}} \\ F_{i,2} &= pk_E^{r_i} g^{a_i} \\ {pk_E}^{r_i} &= F_{i,2} \cdot (g^{a_i})^{-1} \end{aligned}$$

$$= F_{i,2} \cdot (pk_s \cdot g^{-z})^{\frac{1}{h(PK_C)}}$$
$$= F'_{i,2}$$

Hence, by verifying that $(g, pk_E, F_{i,1}, F'_{i,2})$ is a DDH tuple, the financial institution is convinced that z is a valid share of sk_s. Then, given

$$z_1 = a_i \cdot h(PK_{c_1}) + sk_s$$
$$z_2 = a_i \cdot h(PK_{c_2}) + sk_s$$

The financial institution can recover the sk_s by

$$sk_s = z_1 \frac{h(PK_{c_2})}{h(PK_{c_2}) - h(PK_{c_1})} + z_2 \frac{h(PK_{c_1})}{h(PK_{c_1}) - h(PK_{c_2})}$$

B. Security Analysis

Under the security model, we analyze the security of BAPIM in this part. We demonstrate that BAPIM achieves anonymity, unforgeability, and accountability by the following theorems.

Theorem 1: The proposed BAPIM achieves anonymity if the ECDSA signature scheme is existential unforgeable under the chosen message attacks (EUF-CMA) and the ElGamal encryption is indistinguishable under the chosen plaintext attacks (IND-CPA).

Proof: In the register phase, TTP encrypts the borrower's ID with its public key pk_s. Then, it signs the ciphertext using the **ECDSA.Sign** algorithm. The ciphertext D and its signature are included in the mortgage message that a borrower sends to a financial institution. Because the ECDSA is EUF-CMA secure, one can make sure that Crt_1 is generated by TTP. Moreover, since ElGamal encryption is IND-CPA secure, the adversaries cannot know any information about the ID. Thus, anonymity is achieved. ■

Theorem 2: The proposed BAPIM achieves unforgeability if the NIZKP protocol for the DDH tuple is adaptive zero-knowledge, the ElGamal encryption is IND-CPA secure and the ECDSA scheme is EUF-CMA secure.

Proof: This security property is proved by a series of games. In the following games, the successful event in a game G_i is denoted as S_i. ■

Game 0: Game 0 is the GAME Unforgeability.

Game 1: The process in Game 1 is almost identical as Game 0. The difference is that in the **KGen** algorithm of Game 1, we use the simulator $S_{1,\text{NIZKP}}$ to generate (crs, τ), which is used in the NIZKP protocol.

Analysis: If the NIZKP protocol for DDH tuple is adaptive zero-knowledge, these two games are indistinguishable with a negligible probability ϵ_1.

Game 2: The process in Game 2 is almost identical as Game 1. The difference is that in the **BAPIM.ReqGen** algorithm of Game 2, we use the simulator $S_{2,\text{NIZKP}}(crs, \pi, (g, pk_E, F_{i,1}, F'_{i,2}))$ to generate the proof π used in the NIZKP protocol.

Analysis: These two games are indistinguishable if the NIZKP protocol for DDH tuple is adaptive zero-knowledge, which means that $|Pr[S_2] - Pr[S_1]| \leq \epsilon_2$.

Game 3: The process in Game 3 is almost identical as Game 2. The difference is that in **BAPIM.KGen** algorithm of Game 3, the challenger replaces the results of Elgamal encryption $(F_i)_{i\in[n]}$ with random values in G.

Analysis: Assume the maximum number of assets that a borrower can pledge is n. To compute the difference between these two games, we add $n-1$ additional hybrids and each hybrid has the form of $(F_1, F_2, \ldots, F_n)$. For each j, $1 \leq j \leq n$, $F_j = (F_{j,1}, F_{j,2})$.

Let $\mathcal{H}_0 = (F_1, F_2, \ldots, F_n)$ where each F_j is generated by invoking the ElGamal encryption. In hybrid $\mathcal{H}_j$, we randomize all F_i for $i < j$. To be specific, for $i < j$, we set $F_i = (g^{r_i}, R \cdot g^{a_i})$, where R is a random value in group G. For $i \geq j$, we set $F_i = (g^{r_i}, (g^{sk_s})^{r_i} g^{a_i})$.

Because the ElGamal encryption is IND-CPA secure, the probability to distinguish the two consecutive hybrids is restricted by a negligible probability ϵ_3. Considering all the transactions together, the difference between these two games is $|Pr[S_3] - Pr[S_2]| \leq n \cdot \epsilon_3$. Considering that in practice, the number of assets that a borrower can mortgage would not be large, which means n is not a large number, thus $n \cdot \epsilon_3$ is a negligible probability.

Game 4: The process in Game 4 is almost identical as game 3. The difference is that in **BAPIM.ReqGen** algorithm of Game 4, z is a random number in group G rather than derived from the secret key sk_s.

Analysis: According to the security of threshold secret sharing scheme, the secret value is information-theoretically hidden for the adversary. Therefore, the probability that an adversary can differentiate these two games only with a negligible probability ϵ_4.

Game 5: The process in Game 5 is almost identical as Game 4. The difference is that we abort in Game 5 if the adversary forges a valid ECDSA signature.

Analysis: The distinguishable probability of these two games is that the adversary outputs a valid ECDSA signature. Since the ECDSA is EUF-CMA secure, the probability ϵ_5 of this event occurring is negligible.

In Game 5, we can see that the adversary can no longer win, which means $Pr[S_5] = 0$. Taking all games together, we have that $Pr[S_0] \leq \epsilon_1 + \epsilon_2 + n \cdot \epsilon_3 + \epsilon_4 + \epsilon_5$, where $Pr[S_0]$ represents the probability that the adversary can succeed in the original game, thus our BAPIM achieves unforgeability.

Theorem 3: The proposed BAPIM achieves accountability if the NIZKP protocol is sound.

Proof: Let $(m_1, m_2, \sigma_1, \sigma_2)$ be the output of the adversary, where $m_j = (H(A), PK_{c_j})$, $\sigma_j = (\cdot, z_j, \pi_j)$ for $j \in [2]$. Recall that $pk_b = (pk_s, pk_E, (F_i)_{i\in[n]}, crs)$ and $F_i = (F_{i,1}, F_{i,2})$. Moreover, for $j \in [2]$, $F'_{j,2} = F_{j,2} \cdot (pk_s \cdot g^{-z_j})^{\frac{1}{h(PK_{c_j})}}$. ■

Assume the adversary successfully generates two messages and their corresponding signatures with $F'_{1,2} \neq F'_{2,2}$, where the messages have the same first part but different second part. In **BAPIM.ReqGen** algorithm, the adversary needs to prove that both $(g, pk_E, F_{j,1}, F'_{j,2})$, $j \in [2]$ are valid DDH tuples. However, only one of the two tuples can be successfully verified because of the correctness of ElGamal encryption, which means the adversary cannot forge two tuples that satisfy the demands that the two messages have the same first part and different second part. Or else, these two tuples break the soundness of DDH.

Let E be the event that $F'_{1,2} \neq F'_{2,2}$, we have $Pr[E] \leq 2 \cdot \epsilon_s$, where ϵ_s is the soundness error of DDH.

Since $F'_{1,2} = F'_{2,2}$, $(h(PK_{c_1}), z_1)$ and $(h(PK_{c_2}), z_2)$ are two valid points on the same polynomial with degree 1. Thus, sk_s can be calculated as $sk_s = z_1 \frac{h(PK_{c_2})}{h(PK_{c_2})-h(PK_{c_1})} + z_2 \frac{h(PK_{c_1})}{h(PK_{c_1})-h(PK_{c_2})}$.

Thus, the borrower's identity is calculated by invoking the decryption algorithm of ElGamal encryption scheme with the input (sk_s, D).

VI. Performance Evaluation

In this section, we first numerically analyze the communication cost and computational cost of BAPIM and show the simulation results.

A. Numerical Analysis

Communication cost: In BAPIM, after a borrower generates its public key and private key, it needs to register with TTP. During the registration phase, the borrower sends TTP its ID, public key and the signature on them, which contains $2n + 4$ group elements. Here, n is denoted as the maximum number of the assets that the borrower can mortgage. In general, n is not a large number in practice. In BAPIM, to save the communication cost, the borrower can only send pk_s, ID and the signature to TTP. To respond to the borrower, TTP needs to return the ciphertext of the ID and the signature of the ciphertext, which contains $2n + 6$ group elements.

After the request generation process, the borrower transmits the loan request to the financial institution. The loan request includes m, σ_m, Crt_1 and Crt_2, where m contains 2 group elements and σ_m contains 3 group elements and a proof π. For the financial institution, if the loan request of the borrower is valid, the financial institution needs to sign the message M and upload M and its corresponding signature σ_M to Blockchain. If a double mortgage behavior is detected, the financial institution needs to upload $\{M_1, M_2, \sigma_{M_1}, \sigma_{M_2}, Crt_1\}$ to the blacklist.

Computation cost: BAPIM consists of five algorithms: KGen, Register, ReqGen, ReqVerify, and Extract. Among these algorithms, KGen is a prepossessing procedure, and in general, the key generation is only performed once for all entities. In the following, MUL and EXP denote the multiplication complexity and exponentiation complexity in group G. To calculate a public–private key pair, a borrower needs to perform $2n + 2$ EXP and n MUL operations, while TTP and a financial institution only need 1 EXP operation. In the registration phase, encryption and signing are required for TTP, which would cost 1 hash, 4 EXP and 2 MUL operations. During the ReqGen phase, the borrower needs 2 hash, 4 EXP, and 5 MUL operations.

To verify the correctness of the loan requeset, the financial institution needs to perform 3 hash, 11 EXP and 11 MUL operations. If the signature of the borrower is valid, the computational cost for the financial institution to sign the message m is 2 MUL, 2 EXP operations and 1 hash. If the fraudulent behavior of a borrower is discovered, to extract the identity of the borrower, the financial institution needs to perform 2 EXP and 4 MUL operations.

TABLE III
Time Cost of Cryptographic Algorithms

Algorithm	Time (Unit:ms)
Elgamal.Encrypt	0.6
Elgamal.Decrypt	0.68
ECDSA.Sign	0.7
ECDSA.Verify	0.95

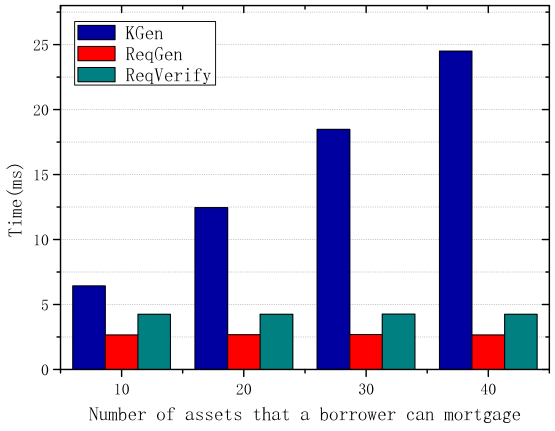


Fig. 4. Computation cost of algorithms in BAPIM.

B. Simulation Results

We conduct a simulation to collect the running time of each phase of BAPIM on a notebook with an Intel(R) Core(TM) i7-7500 U and CPU @ 2.9 GHz. The RAM is 8 GB. In the simulation, we employ the NIST p192 elliptic curve to generate system parameters, which is generated by Miracl library. SHA-1 is chosen as the hash function.

We first measure the performance of ElGamal encryption in terms of encryption and decryption, and evaluate the execution time of the ECDSA in terms of signing and verification. The simulation results are shown in Table III.

Then, we evaluate the performance of the algorithms in BAPIM. Let n denote the number of assets that a borrower can mortgage. As shown in Fig. 4, the key generation cost for the borrower increases linearly with n, since for each asset, the borrower needs to do an ElGamal encryption and generates an $F_i = (g^{r_i}, pk_E^{r_i} g^{a_i})$, where $i \in [n]$. According to the simulation results, if n is changed from 10 to 40 with an increment of 10, the time consumption of key generation for the borrower is 6.42, 12.45, 18.48, and 24.5 ms, respectively. Note that key generation can be performed offline once for each borrower, the overhead is acceptable for borrowers. For the algorithms **ReqGen** and **ReqVerify**, borrowers and financial institutions only need to do a fixed number of operations without changing with n, hence, the time cost for **ReqGen** algorithm is almost 2.68 ms, and the time consumption for **ReqVerify** is about 4.25 ms when n is changed from 10 to 40. For the identity extraction, a financial institution only needs 2 EXP operations and 2 MUL operations to extract the secret key of the malicious borrower and 1 ElGamal decryption to obtain the identity. Therefore, the time cost of **Extract** algorithm for the financial institution is small.

VII. Related Work

Mortgage fraud detection has already been investigated by social scientists and economists to reduce risks for financial institutions. The traditional approach for mortgage fraud prevention is to building a centralized computer system or the regulator who is responsible for collecting and analyzing the abnormal situations. Ngai *et al.* [20] employed data mining techniques, for example, logistic models and neural networks, to detect financial fraud. Gestel *et al.* [21] analyzed the facilitating circumstances of the mortgage fraud and proposed two approaches to reduce the mortgage fraud, one is to increase the integrity of professionals, and the other focuses on the information exchange. The methods above deal with the fraud detection based on mortgage information of borrowers, but the privacy of borrowers are not protected during mortgage data sharing. In this paper, we utilize the anonymous authentication to achieve a secure and reliable mortgage without leaking the identity information and the asset information of borrowers to others.

To protect users' privacy, many anonymous authentication schemes have been proposed [22]–[30]. In group signature schemes [22], it can be verified that a signature is generated from a member in this group, but do not know the real identity information of the signer. There is a group manager who can trace the signer of a signature, thus group signatures own the anonymity for others except for the group manager. Bellare [23] constructed a dynamic group signature scheme, which can support the addition or removal of the group members. Boneh *et al.* [24] designed a short group signature whose signature size is almost the same as the RSA signature. Zheng *et al.* [25] proposed a group signature scheme which can fulfill anonymous communication between the sender and the receiver. Moreover, it can achieve the traceability and auditing for the communication sender simultaneously. Ring signatures [26] are simplified group signature schemes without group managers. By making use of a ring signature scheme, one can create a signature endorsed by a ring member without leaking which member actually created the signature. Torres [27] proposed a ring signature scheme that can preserve the anonymous of the signer. Besides that, the scheme allows one to publicly check whether two or more signatures are derived from the same signer. Anonymous credentials [28] can also be used to achieve anonymous authentication. Anonymous credentials have a stronger privacy guarantee than traditional credentials. In an anonymous credential system, a user has multiple pseudonyms and for a same user, different pseudonyms cannot be linked together. Ni *et al.* [29] proposed a privacy-preserving valet parking scheme which extends anonymous authentication to support two-factor authentication for reducing the risks of vehicle theft while protecting the privacy of users. Yang *et al.* [30] proposed a decentralized anonymous credential system, where the service providers can authenticate users based on their historical behaviors. In our scheme, verifiable secret sharing, zero-knowledge proof and ElGamal encryption are integrated to achieve efficient and privacy-preserving double mortgage detection.

VIII. Conclusion

In this article, we proposed a blockchain-based accountable and privacy-preserving scheme (BAPIM) for industrial mortgages management. BAPIM enables a financial institution to share the mortgage records with others, such that the institution can detect fraudulent behavior of a borrower for double-mortgage prevention. To protect the borrowers' privacy, anonymous authentication of borrowers was realized and other financial institutions have no knowledge about the mortgage data of honest borrowers. If the double-mortgage behavior of a borrower is detected, the secret key and the identity of the borrower would be disclosed to the corresponding financial institution. BAPIM has high security guarantees and computational efficiency, and is suitable to be implemented to support industrial mortgage management. In the future, we aim to construct a privacy-preserving and verifiable repayment scheme for industrial mortgage based on the public blockchain.

References

[1] M. Aazam, S. Zeadally, and K. A. Harras, "Deploying fog computing in industrial Internet of Things and industry 4.0," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4674–4682, Oct. 2018.

[2] Y. Zhang, C. Xu, H. Li, K. Yang, J. Zhou, and X. Lin, "HealthDep: An efficient and secure deduplication scheme for cloud-assisted ehealth systems," *IEEE Trans. Ind. Informat.*, vol. 14, no. 9, pp. 4101–4112, Sep. 2018.

[3] S. Wu *et al.*, "Survey on prediction algorithms in smart homes," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 636–644, Jun. 2017.

[4] H. Yang, Q. Zhou, M. Yao, R. Lu, H. Li, and X. Zhang, "A practical and compatible cryptographic solution to ADS-B security," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3322–3334, Apr. 2018.

[5] M. B. Aalbers, "The financialization of home and the mortgage market crisis," *Financialization Housing*, pp. 40–63, 2016.

[6] Q. Yang and H. Wang, "Toward trustworthy vehicular social networks," *IEEE Commun. Mag.*, vol. 53, no. 8, pp. 42–47, Aug. 2015.

[7] J. Mendling *et al.*, "Blockchains for business process management-challenges and opportunities," *ACM Trans. Manag. Inf. Syst.*, vol. 9, no. 1, 2018, Art. no. 4.

[8] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. Shen, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Trans. Dependable Secure Comput.*, to be published, doi: 10.1109/TDSC.2018.2791432.

[9] A. Tapscott and D. Tapscott, "How blockchain is changing finance," *Harvard Business Rev.*, vol. 1, no. 9, pp. 2–5, 2017.

[10] K. Korpela, J. Hallikas, and T. Dahlberg, "Digital supply chain transformation toward blockchain integration," in *Proc. Hawaii Int. Conf. Syst. Sciences*, 2017, pp. 4182–4191.

[11] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing atop PoS blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, Jun. 2019.

[12] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Trans. Ind. Informat.*, vol. 13, no. 6, pp. 3154–3164, Dec. 2017.

[13] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, no. 1, pp. 36–63, 2001.

[14] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[15] P. Feldman, "A practical scheme for non-interactive verifiable secret sharing," in *Proc. Annu. Symp. Found. Comput. Sci.*, 1987, pp. 427–438.

[16] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM J. Comput.*, vol. 18, no. 1, pp. 186–208, 1989.

[17] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," in *Proc. Annu. Int. Cryptology Conf.*, 1997, pp. 410–424.

[18] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proc. Conf. Theory Appl. Cryptographic Techn.*, 1986, pp. 186–194.

[19] C. Rackoff and D. R. Simon, "Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack," in *Proc. Annu. Int. Cryptology Conf.*, 1991, pp. 433–444.

[20] E. W. T. Ngai, Y. Hu, Y. H. Wong, Y. Chen, and X. Sun, "The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature," *Decis. Support Syst.*, vol. 50, no. 3, pp. 559–569, 2011.

[21] B. Gestel, "Mortgage fraud and facilitating circumstances," in *Situational Prevention of Organised Crimes*. Wilan, 2013, pp. 129–147.

[22] D. Chaum and E. Heyst, "Group signatures," in *Proc. Workshop Theory Appl. Cryptographic Techn.*, 1991, pp. 257–265.

[23] M. Bellare, H. Shi, and C. Zhang, "Foundations of group signatures: The case of dynamic groups," in *Proc. Cryptographer's Track RSA Conf.*, 2005, pp. 136–153.

[24] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," in *Proc. CRYPTO*, 2004, pp. 41–55.

[25] H. Zheng, Q. Wu, B. Qin, L. Zhong, S. He, and J. Liu, "Linkable group signature for auditing anonymous communication," in *Proc. Australasian Conf. Inf. Secur. Privacy*, 2018, pp. 304–321.

[26] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Proc. Int. Conf. Theory Appl. Cryptology Inf. Security*, 2001, pp. 552–565.

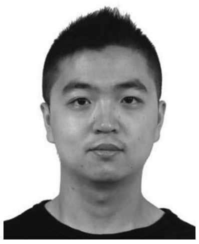
[27] W. A. A. Torres *et al.*, "Post-quantum one-time linkable ring signature and application to ring confidential transactions in blockchain," in *Proc. Australasian Conf. Inf. Secur. Privacy*, 2018, pp. 558–576.

[28] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Commun. ACM*, vol. 28, no. 10, pp. 1030–1044, 1985.

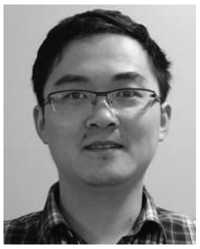
[29] J. Ni, X. Lin, and X. Shen, "Toward privacy-preserving valet parking in autonomous driving era," *IEEE Trans. Veh. Technol.*, vol. 68, no. 3, pp. 2893–2905, Mar. 2019.

[30] R. Yang, M. H. Au, Q. Xu, and Z. Yu, "Decentralized blacklistable anonymous credentials with reputation," *Comput. Secur.*, vol. 85, pp. 353–371, 2019.

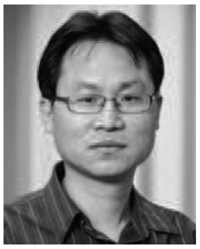
Liang Xue received the B.S. degree in information security and the M.S. degree in computer engineering from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2015 and 2018, respectively. Currently, She is working toward the Ph.D. degree in electrical and computer engineering with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada. Her research interests include applied cryptography, cloud computing, and blockchain.


Dongxiao Liu (S'13) received the B.S. and M.S. degrees in information security from the School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2013 and 2016, respectively. Currently, he is working toward the Ph.D. degree in electrical and computer engineering with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada.

His research interests include applied cryptography and privacy enhancing technologies for blockchain.


Jianbing Ni (M'18) received the B.E. and M.S. degrees in information security and computer engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2014, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2018.

He is currently a Postdoctoral Research Fellow with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. His research interests are applied cryptography and network security, with current focus on cloud computing, smart grid, mobile crowdsensing, and Internet of Things.


Xiaodong Lin (M'09–SM'12–F'17) received the Ph.D. degree in information engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 1998, and the second Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2008.

He is currently an Associate Professor of Network Security and Applied Cryptography with the School of Computer Science with the University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security.

Dr. Lin was the recipient of the Outstanding Achievement in Graduate Studies Award from the University of Waterloo.

Xuemin (Sherman) Shen (M'97–SM'02–F'09) received Ph.D. degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA , in 1990.

He is currently a University Professor with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada. His research interests include resource management in interconnected wireless/wired networks, wireless network security, social networks, smart grid, and vehicular ad hoc and sensor networks.

Dr. Shen is a registered Professional Engineer of Ontario, Canada, a Fellow of the Engineering Institute of Canada, the Canadian Academy of Engineering, the Royal Society of Canada, and a Distinguished Lecturer of IEEE Vehicular Technology Society and Communications Society. He is the Editor-in-Chief for IEEE INTERNET OF THING JOURNAL and the Vice President on publications of IEEE Communications Society. He served as the Technical Program Committee Chair/Co-Chair for IEEE Global Communications Conference' 16, IEEE IEEE International Conference on Computer Communications'14, IEEE Vehicular Technology Conference'10 Fall, the Symposia Chair for IEEE International Conference on Communications '10, Tutorial Chair for IEEE Vehicular Technology Conference'11 Spring, Chair for IEEE Communications Society Technical Committee on Wireless Communications, and P2P Communications and Networking. He is the recipient of the Joseph LoCicero Award, in 2015, Education Award, in 2017, Harold Sobol Award, in 2018, and James Evans Avant Garde Award from the IEEE Communications Society, in 2018. He was also the recipient of the Excellent Graduate Supervision Award, in 2006, and the Outstanding Performance Award in 2004, 2007, 2010, 2014, and 2018 from the University of Waterloo, the Premier's Research Excellence Award (PREA), in 2003, from the Province of Ontario, Canada.