An Efficient NIZK Scheme for Privacy-Preserving Transactions Over Account-Model Blockchain

Shunli Ma, Yi Deng, Debiao He, Jiang Zhang, and Xiang Xie

Abstract—We introduce the abstract framework of decentralized smart contracts system with balance and transaction amount hiding property over account-model blockchain. To build a concrete system with such properties, we utilize a homomorphic public-key encryption scheme and construct a highly efficient non-interactive zero knowledge (NIZK) argument based upon the encryption scheme to ensure the validity of the transactions. Our NIZK scheme is perfect zero knowledge in the common reference string model, while its soundness holds in the random oracle model. Compared to previous similar constructions, our proposed NIZK argument dramatically improves the time efficiency in generating a proof, at the cost of relatively longer proof size.

Index Terms—Non-interactive zero knowledge, decentralized smart contracts, Account model

1 Introduction

Bitcoin [1], as the first widely successful decentralized digital currency, has drawn a lot of attention to the conception of blockchain. A blockchain is a tamper-proof digital ledger of transactions with chronological order maintained by distributed consensus nodes (called miners). The miners reach consensus not only on the transactions (e.g., money transfer records or other data) but also on the involving computations (e.g., validate or update the transactions). This guarantees the blockchain to possess decentralization, verifiability and immutability. Due to these properties, blockchain has been used in the design of systems for data storage [2], provenance [3], [4], sharing economy [5], dynamic key management [6], supply chain finance and so forth.

Although the blockchain can provide a powerful abstraction for the design of distributed protocols, the security and privacy issues (e.g., the leakage of user real identity, transaction amount and balance) should not be ignored from the protection of users' interests. Among these security and privacy concerns, hiding the transaction amount and balance is especially important when designing a blochchain-based system involving economic dealings (e.g., sharing economy or supply chain finance system). Here, we take the blockchain-driven supply chain finance (BDSCF) system [7] as an example to specify the potential threats without a protection mechanism for money transfer records.

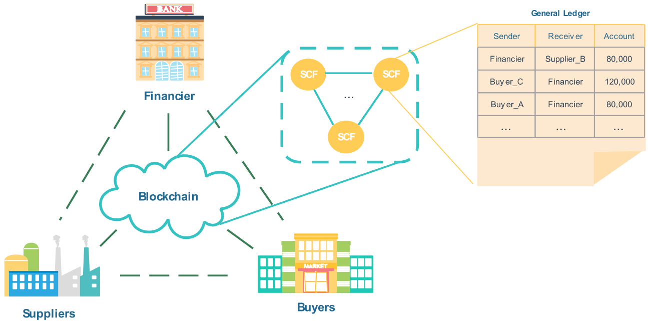
The BDSCF system was proposed to cut unnecessary costs during the deal appears between a supplier and a buyer who trust different supply chain finances (SCFs). Due to the integration of blockchain into supply chain finance system, SCFs (as the distributed miners) collectively maintain a general ledger (see Fig. 1) which avoids complicated data synchronism across the participating SCFs and eliminates the inefficiencies in financial flaws. Consequently, it helps the company financing make a higher profits and lower cost. Although BDSCF can enhance the efficiency of trading processes among supply chain partners and improve the buyer-supplier relation during the payment process, the disclosure of the transferred and balance in general ledger to SCFs which may leak key trade secrets of the suppliers. That is, the price of products from different suppliers involved in the general ledger can be estimated by analysing transaction records and balance in account. As a result, the suppliers' incentives to adopt this blockchain-based mechanism will be diminished for their dinterests are compromised, which seriously limits the application and scalability of BDSCF.

In order to protect suppliers' commercial interests, we consider a direct but efficient method, i.e., hiding the transferred and balance involved in the ledger. If we can conceal the amount in both the user's account and the transaction, the threats of amount-change analysed by compromised SCFs or other adversaries will be mitigated.

There has been progress in designing privacy-preserving schemes (e.g., Confidential Transaction [8], Zerocash [9], Monero [10]), details of which will be described in the Section 1.2. Most of them focus on hiding the transaction accounts via several cryptographic techniques (e.g., cryptographic commitment, zero-knowledge proof, ring signature,

- S. Ma and Y. Deng are with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, and the School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China. E-mail: {mashunli, deng}@iie.ac.cn.
- D. He is with the School of Cyber Science and Engineering, Wuhan University, Wuhan, Hubei 430072, China, and also with the State Key Laboratory of Cryptology, Beijing 100878, China. E-mail: hedebiao@163.com.
- J. Zhang is with the State Key Laboratory of Cryptology, Beijing 100878, China. E-mail: jiangzhang09@gmail.com.
- X. Xie is with the Juzix Technology Co. Ltd., Shenzhen, Guangdong 518000, China. E-mail: xiexiang@juzix.io.

Manuscript received 25 Jan. 2018; revised 14 Aug. 2019; accepted 21 Jan. 2020. Date of publication 24 Jan. 2020; date of current version 12 Mar. 2021.
(Corresponding author: Yi Deng.)
Digital Object Identifier no. 10.1109/TDSC.2020.2969418



Sender	Receiver	Account
Financier	Supplier_B	80,000
Buyer_C	Financier	120,000
Buyer_A	Financier	80,000
...	...	...

Fig. 1. The model of blockchain-based supply chain finance system.

etc). Notice that the coins of them are in Bitcoin's UTXO (Unspent Transaction Outputs) model and a user's balance is the sum of all outputs regulated by wallet. In the UTXO model, your wallet will simultaneously create a new address for the change you are owed when greater coins are sent to another user. Subsequently, the emergency of Ethereum [11] has introduced an innovation model (the Account model), which relies on global state storage of accounts, balances, code and storage (i.e., the user's balance now is kept as global state). Analogous to a bank account, there is a debit and corresponding credit to the states with a transaction.

When considering the privacy of user's balance, previous UTXO-based researches may not work for the following reasons. First, the cryptographic commitment scheme may bring about the difficulty for the concurrent balance-updating in the system. Second, high computational complexity greatly restricts their application in the lightweight but widespread used devices (e.g., mobile phone). Finally, none of them support the smart contract system of Ethereum, which offers more flexible and arbitrary trading operations running in the blockchain. Thus, we are motivated to propose a mechanism with the Account model for creating an expressive decentralized smart contract (DSC) system with the above hiding and updating.

In order to achieve hiding and timely updating operations to the balance, we employ the homomorphic encryption (HE) schemes. Both the amount of transferred records and balance are encrypted by the HE algorithms and stored in ciphertext. The homomorphism of HE allows the miners to directly update the balance in ciphertext without the need of decryption, that is, given encryptions $E(v_1), E(v_2), \ldots, E(v_t)$ of the balance $v_1, v_2, \ldots, v_t$, the miners can efficiently compute a ciphertext of $f(v_1, v_2, \ldots, v_t)$, where $f(\cdot)$ is an efficiently computable function (this function is mainly related to addition or substraction operation in our paper). In addition, we propose a zero knowledge (ZK) proof to prove two basis statements required by a transaction. One is "equivalence" (i.e., Alice's balance decreases v and Bob's should correctly add v when Alice transfers v coins to Bob), and the other is "enough" (i.e., Alice's balance should not be less than v if she want to transfer money v to others). Our goal in this paper is to construct an efficient ZK scheme suitable for lightweight devices, under the premise that the proof size is not excessively increased. An efficient Σ-protocol can be used to prove two ciphertexts correspond the same message for the "equivalence" condition. A range proof works well for "enough" condition. In a subsequent work, Bünz *et al.* propose a relatively short range proof-Bulletproofs [12]. However, the recursive execution of the protocol gains the overhead of prover and verifier; since Bulletproofs use Pedersen vector commitments to hide the secret, we have to introduce this primitive and prove the plaintext-equivalence relation between a hommorphic encryption ciphertext and a Pedersen vector commitment when adapting this method. Hence, we utilize the techniques from [13] to construct a range proof for "enough" condition. The main idea of the range proof is that for a secret $t \in [0, u^l)$, the prover writes it in u-ary notation (i.e., $t = \sum_{j=0}^{l-1} t_j \cdot u^j$) and shows that each element t_j in the range $[0, u)$. At a high level, their range proof can be converted into a special Σ-protocol with a setup process to produce signatures of each elements in $[0, u)$. The whole protocol we propose is public coin; we compile it into a NIZK scheme in the random oracle model, via Fiat-Shamir heuristic method.

1.1 Our Contributions

In this section, we summarize the contributions of this paper as follows:

1) We utilize a homomorphic public-key encryption scheme to hide the balance and transaction amount. The main contribution of this paper is to construct a non-interactive zero knowledge (NIZK) scheme to prove the validity of the transactions. The NIZK scheme has a highly efficient prover, at the cost of a longer proof. The in-depth security proof shows that our proposed scheme is provably secure under the random oracle model.
2) Based on the above NIZK scheme, we introduce a priori mechanism enabling programmability (i.e., decentralized smart contract) with balance hiding property under the Account model. This mechanism can be applied in various financial scenarios and can also work when a system involves economic dealings or even change in digital assets.
3) We analyze the performance of the proposed scheme both in asymptotic and practical terms, and also implement it on a personal computer. The encouraging results indicate that our scheme is practicable and maneuverable in the mentioned actual applications.

1.2 Related Work

In this section, we briefly review some existing cryptographic techniques around the privacy protection in the blockchain, however which are not suitable to the demand of balance confidentiality and timely updating in our system.

Bitcoin Core Developer Gregory Maxwell [8] first conceptualizes Confidential Transaction as a solution for keeping the transaction amounts unrevealed. Their solution is based on the Pedersen commitment scheme [14], where the transaction amounts are masked by random blinding factors before sent to the recipients and lately notarized by the recipients. The clear thing is that, these masked amounts still can be used for certain types of calculations, which means that all inputs and outputs of a transaction can be added up respectively and these two sums can be compared to ensure trade-off during the verifying process without revealing the real values.

Ring Confidential Transaction (RingCT) is another variant CT approach for hiding transaction amounts. Collaborated with the linkable ring signature scheme [15], Monero [16] (another proof-of-work cryptocurrency) achieves the requirements of decentralization, privacy and anonymity. Similar to [8], the RingCT scheme improves the privacy of the blockchain by allowing the amounts sent in a transaction to be concealed in an anonymous set. In addition, the linkable mechanism is equipped to ensure any double-spending behaviors can be detected timely.

However, the CT-based schemes uses blinding factors for inputs and outputs, which are picked in special so that they add up correctly. This may cause lower randomness and reduce the security of the whole scheme. In addition, the blinding factors may need to be somehow synchronized to both sides, which may lead to concurrency problems and have slightly difficulty when implementing into a financial system (e.g., BDSCF).

Another cryptographic method, upon which there has been a lot of research, is zero-knowledge proof. Zerocash [9] employs the zero-knowledge succinct non-interactive argument of knowledge (zkSNARKs) [17], [18] and cryptographic commitment schemes to achieve the highest level of privacy protection and anonymity of the cryptocurrency based on UTXO model. The transfer transaction consists of a cryptographic commitment to a new coin, which specifies the coin's value, owner address and unique serial number. When consuming the input coins, zero-knowledge proofs and serial numbers are needed to prove the ownership of the input coins and the trade-off between the inputs and outputs. zkSNARKs generate proof with constantant group elements, and have very fast verification time. However, on one hand the cryptographic commitments generated by the one-way hash functions do not support the Account model, since homomorphic operations are not considered while Zerocash was designed. on the other hand the proof generation process and the trusted setup process in zkSNARKs are rather expensive which leads to the worse efficiency and not suitable for the lightweight devices (e.g., mobile phones). Besides, they also rely on strong unfalsifiable assumptions such as the *knowledge-of-exponent* assumption. In subsequent work Ben-Sasson *et al.* present a proof system named zkSTARKs [19], which are zero-knowledge based on *collision resistant hash functions*. However the proof constructions is still very costly and the proof size becomes longer. Bünz *et al.* propose Bulletproofs [12]. They represent a secret as inner products of two vectors, and use Pedersen vector commitments to hide the vectors, then construct zero knowledge proofs via recursively invoking their protocol for inner products. Hyrax [20] gives a zero knowledge proof for layered arithmetic circuit via using the techniques that apply Cramer-Damgard transformation [21] to doubly-efficient Interactive Proofs [22]. When consider real-life applications, especially for lightweight devices, the proof generation process of Bulletproofs and Hyrax is too slow to be used. All of the constructions mentioned above aim at general relations, which is also reduce the performance since it does not exploit the characteristic of the underlying statement and needs to convert the statement to the arithmetic circuit or the RAM model.

Instead of UTXO model, Ethereum [11] introduce the Account model (mentioned in Section 1) and a decentralized arbitrary user-defined programming system running in the blockchain, named of smart contract system. Followed the idea of smart contract, Kosba *et al.* [23] implements a cryptographic suite that can blind transactions with programmable logic. It applies smart contract to store the committed coins generated by the users and determine the payout distribution. Once the users open the commitments and uncover the information to the manager (who is trusted not to disclosed the user's private data), the manager then interact with the smart contract to generate new coins and pay to the recipients. The new coins will lately be submitted to the blockchain with zero knowledge proofs for its legality. This scenario provides programmability without exposing explicit transaction information to the public. However, since the manager always knows users' quotes, this scheme is not suitable for the privacy protection in terms of transaction amount and balance in our scenario.

1.3 Organization

We organize the remainder of this paper as follows. Section 2 contains background materials such as bilinear pairings, homomorphic encryption, Σ-protocols, non-interactive zero knowledge proofs and some complexity assumptions. In section 3, we describe our NIZK scheme, including theconstruction with its corresponding proof. Section 4 discusses the concrete instantiation of our scheme and demonstrates a comparison with previous schemes. Section 5 concludes this paper.

2 Preliminaries

In this section we give basic definitions of cryptographic primitives including required tools and complexity assumptions, along with some properties if necessary.

Notations. If n is an integer, we denote $[n] = \{1, \ldots, n\}$. For any set S, $x \leftarrow_{\$} S$ means sampling uniformly at random some element x from the set S. Besides, for any distribution D, $x \leftarrow_{\$} D$ means sampling x from the probability distribution D, and $v \in_R D$ denotes that variable v is uniformly random in D. We write $y = A(x; r)$ to represent that an algorithm A takes input x and randomness r, output y. The formula $y \leftarrow A(x)$ means picking randomness r uniformly at random and setting $y = A(x; r)$.

In this paper, we denote by n the security parameter, and abbreviate probabilistic polynomial-time as PPT. A function $\epsilon(n)$ is negligible in n if $\epsilon(n) = o(1/n^c)$ for all $c \in \mathbb{N}$. $\epsilon(n) = \mathsf{negl}(n)$ denotes that $\epsilon(n)$ is a negilible function in n.

For any two distribution ensembles $\{X_n\}_{n\in\mathbb{N}}$ and $\{Y_n\}_{n\in\mathbb{N}}$ indexed by a security parameter n, we write $X_n \overset{c}{\approx} (\overset{s}{\approx} \equiv) Y_n$ to represent the two distribution ensembles are computational indistinguishable (statistical indistinguishable, identical).

2.1 Cryptographic Primitives

Bilinear Groups. We call $\mathcal{G}_{bp}(1^n)$ the bilinear group generator which takes a security parameter as input and outputs a description of a bilinear group $\mathsf{gk} = (p, G_1, G_2, G_T, e, g_1, g_2)$ such that p is a n-bit prime. We follow the notation of [24]:

- G_1, G_2, G_T are multiplicative cyclic groups of order p. The elements g_1, g_2 generates G_1, G_2 respectively.
- $e : G_1 \times G_2 \rightarrow G_T$ is a nondegenerate bilinear map, and $e(g_1, g_2)$ generates G_T.

- $\phi : G_2 \rightarrow G_1$ is a computable isomophism, and $g_1 = \phi(g_2)$.
- $\forall a, b \in \mathbb{Z}, e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.
- It is efficient to compute group operations, compute the bilinear map, and decide the membership in G_1, G_2 and G_T.

Definition 1 (DLIN Assumption). *With G_1 as above, the Decision Linear assumption (DLIN) [25] in G_1 states that for all non-uniform* PPT $\mathcal{A}$ *we have*

$$\left| \begin{array}{l} \Pr\left[\begin{array}{c} u, v, g \leftarrow_{\$} G_1, r, s \leftarrow_{\$} \mathbb{Z}_p : \\ \mathcal{A}(u, v, g, u^r, v^s, g^{r+s}) = 1 \end{array}\right] \\ - \Pr\left[\begin{array}{c} u, v, g, \rho \leftarrow_{\$} G_1, r, s \leftarrow_{\$} \mathbb{Z}_p : \\ \mathcal{A}(u, v, g, u^r, v^s, \rho) = 1 \end{array}\right] \end{array} \right| \leq \mathsf{negl}(n).$$

A public-key encryption (PKE) scheme consists of three PPT algorithms (PKE.Gen, PKE.Enc, PKE.Dec) which indicate key generation, encryption, and decryption. We require that $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Gen}(1^n)$ and for any valid plaintext m and randomness r, $\mathsf{PKE.Dec}_{\mathsf{sk}}(\mathsf{PKE.Enc}_{\mathsf{pk}}(m; r)) = m$. A PKE scheme is IND-CPA secure(a.k.a. semantically secure [26]) if

$$\Pr\left[\begin{array}{cc} b \leftarrow_{\$} \{0,1\} & \\ (\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{PKE.Gen}(1^n) & \\ m_0, m_1 \leftarrow \mathcal{A}(\mathsf{pk}) & : b = b' \\ c = \mathsf{PKE.Enc}_{\mathsf{pk}}(m_b; r) & \\ b' \leftarrow \mathcal{A}(\mathsf{pk}, c) & \end{array}\right] \leq \mathsf{negl}(n).$$

In this paper, we design a hommomorphic PKE scheme to hide the balance and transaction for higher privacy require. Next we describe the construction and give its security proof under the DLIN assumption.

Definition 2. *The homomorphic encryption (HE) scheme is a triple of algorithms* (PKE.Gen, PKE.Enc, PKE.Dec)*:*

- $\mathsf{PKE.Gen}(1^n)$*: Choose* $h \leftarrow_{\$} G_1, x_1, x_2 \leftarrow_{\$} \mathbb{Z}_p;$ and *set* $\mathsf{sk} = (x, y), \mathsf{pk} = (X_1 = g_1^{x_1}, X_2 = g_1^{x_2}, g_1, h)$, *finally output the pair of keys.*
- $\mathsf{PKE.Enc}_{\mathsf{pk}}(m; (r, s))$*: Compute* $C_1 = X_1^r, C_2 = X_2^s$, $C_3 = g_1^{r+s} \cdot h^m$, and output $C = (C_1, C_2, C_3)$, *where* $r, s \leftarrow_{\$} \mathbb{Z}_p$ *denotes the randomness used by* Enc.
- $\mathsf{PKE.Dec}_{\mathsf{sk}}(C)$*:* Parse C into a tuple (C_1, C_2, C_3), *compute* $h_m = C_3/(C_1^{1/x} \cdot C_2^{1/y})$. *One can efficiently get the message* $m = log_h^{h_m}$ *if the plaintext space is small.*

Lemma 1. *The HE scheme described above is semantically secure under the DLIN assumption.*

Proof. Suppose an efficient adversary $\mathcal{A}$ breaks the HE scheme in the IND-CPA sense with non-negligible probability $\mathsf{poly}(n)$, then an algorithm $\mathcal{B}$ could be constructed from it as follows to break the DLIN assumption (i.e., given a tuple $u, v, g, s_1 = u^r, s_2 = v^s, s_3$, decides whether $s_3 = g^{r+s}$):

```
Algorithm $\mathcal{B}^{\mathcal{A}}$ $(u, v, g, s_1, s_2, s_3)$:
  choose $h \leftarrow_{\$} G_1$, and set $\mathsf{pk} = (u, v, g, h)$;
    $(m_0, m_1) \leftarrow \mathcal{A}(\mathsf{pk})$;
    sample $b \leftarrow_{\$} \{0,1\}$, then set $c^* = (s_1, s_2, s_3 \cdot h^{m_b})$;
    $b' \leftarrow \mathcal{A}(\mathsf{pk}, c^*)$;
    if $b = b'$ then return 1;
    else return 0.
```

- If $s_3 = g^{r+s}$, then the probability of $\mathcal{B}$ outputs 1 is equal to $\mathcal{A}$'s probability of guessing the hidden bit correctly, which is $\mathsf{poly}(n) + \frac{1}{2}$.
- If s_3 is a random element in G_1, then $s_3 \cdot h^{m_b}$ is uniformly distributed in G_1, and independent of b, so the probability of $\mathcal{A}$ answers correctly is $\frac{1}{2}$.

Hence, the probability of $\mathcal{B}$ to distinguish distributions $\{u, v, g, u^r, v^s, g^{r+s}\}$ and $\{u, v, g, u^r, v^s, \rho\}$ equals $\mathsf{poly}(n)$, a non-negligible probability, which contradicts the DLIN assumption. □

Definition 3 (q-SDH Assumption). *With the bilinear group* $\mathsf{gk} = (p, G_1, G_2, G_T, e, g_1, g_2)$ *generated by* $\mathcal{G}_{bp}(1^n)$, *the q-Strong Diffie-Hellman (q-SDH) assumption [27] associated to* gk *holds if for all non-uiform* PPT $\mathcal{A}$, *we have*

$$\Pr\left[\begin{array}{c} \mathsf{gk} \leftarrow \mathcal{G}_{bp}(1^n), x \leftarrow_{\$} \mathbb{Z}_p : \\ (c, g_1^{1/(x+c)}) \leftarrow \mathcal{A}(g_1, g_1^x, \ldots, g_1^{x^q}, g_2, g_2^x) \end{array}\right] \leq \mathsf{negl}(n),$$

where $c \in \mathbb{Z}_p$.

In a signature scheme, there exist a triple of polynomailtime algorithms (Sig.Gen, Sig.Sign, Sig.Vrfy) for generating keys, signing, and verifing signatures, respectively. We require that under $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Sig.Gen}(1^n)$ and for any valid message m, $\mathsf{Sig.Vrfy}_{\mathsf{vk}}(m, \mathsf{Sig.Sign}_{\mathsf{sk}}(m)) = 1$.

Consider the following attack model, the advesary submits q queries $m_1, \ldots, m_q$ to the challenger for asking their signatures. The challenger runs $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Sig.Gen}(1^n)$ and sends vk to the adversary, together with signatures $\sigma_1, \ldots, \sigma_q$ on $m_1, \ldots, m_q$. We say the adversary wins if it outputs a signature σ' such that $\mathsf{Sig.Vrfy}_{\mathsf{vk}}(m', \sigma') = 1$ and $m' \notin \{m_1, \ldots, m_q\}$. A signature scheme is said to be existential unforgeability against weak chosen message attacks (EUF-WCMA) if no PPT adversary wins the game with a non-negligible probability.

Definition 4 (Boneh-Boyen Signature). *Boneh-Boyen signature consists of three polynomial-time algorithms:*

- $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{Sig.Gen}(1^n)$*: The randomized key generation algorithm takes the security parameter n as input, randomly choose* $\lambda \leftarrow_{\$} \mathbb{Z}_p$, set $(\mathsf{sk}, \mathsf{vk}) = (\lambda, g_2^{\lambda})$.
- $\sigma \leftarrow \mathsf{Sig.Sign}_{\mathsf{sk}}(m)$*: The deterministic signing algorithm uses the private signing key* sk *and input m. It* outputs $\sigma = g_1^{\frac{1}{\lambda+m}}$.
- $\{0,1\} \leftarrow \mathsf{Sig.Vrfy}_{\mathsf{vk}}(m, \sigma)$*: Given the public verification key* vk*, the deterministic verification algorithm outputs 1 if* $e(\sigma, \mathsf{vk} \cdot g_2^m) = e(g_1, g_2)$*, and 0 otherwise.*

Under the q-SDH assumption, the Boneh-Boyen signature scheme is EUF-WCMA, which is sufficient enough for our goal. For more detail information on this proof, see [27].

Σ-Protocol. Let $R = \{(x, w)\}$ be a binary relation which can be efficiently decidable, where x is a statement and $|w| = \mathsf{poly}(n)(|x|)$ is a witness. Let $L_R = \{x : \exists w \text{ s.t. } (x, w) \in R\}$ be an NP language.

Definition 5 (Σ-Protocol). *A 3-round public-coin protocol* $\Pi = (a, c, z)$ *between a prover* P *and a verifier* V *is a Σ-protocol for language L_R if the following conditions hold:*

- *Completeness: If* P *and* V *execute the protocol on input x and private input w to* P *in which $(x, w) \in R$, then* V *always accepts.*

- *Special soundness: For any statement x, given two accepting transcripts on input x: $(a, c, z), (a, c', z')$ where $c \neq c'$, there exists a* PPT *algorithm* Ext *which can compute the witness w s.t. $(x, w) \in R$.*
- *Special honest verifier zero knowledge (SHVZK): There exists a* PPT *algorithm* Sim*, on input x and a challenge c, can perfectly simulate the conversations between the honest* P, V *on input x. Formally speaking,*

$$\left\{\mathsf{Sim}(x,c)\right\}_{x\in L_R, c\in\{0,1\}^n} \equiv \left\{ < \mathsf{P}(w), \mathsf{V}(c) > (x)\right\}_{x\in L_R, c\in\{0,1\}^n};$$

where $\mathsf{Sim}(x,c)$ *represents the output of simulator on input x and c, and $< \mathsf{P}(w), \mathsf{V}(c) > (x)$ denotes the real output transcript of the protocol.*

Definition 6 (NIZK Arguments). *A Non-Interactive Zero Knowledge (*NIZK*) argument system for an NP language L_R consists of a triple of* PPT *algorithms (*K, P, V*):*

- *Completeness: For each $crs \leftarrow \mathsf{K}(1^n)$ and $(x, w) \in R$, we have:*

$$\Pr[\pi \leftarrow \mathsf{P}(x, w, crs) : \mathsf{V}(x, \pi, crs) = 1] \geq 1 - \mathsf{negl}(n).$$

- *(Adaptive) Soundness: For all non-uniform* PPT *prover* P^**, the probability*

$$\Pr\left[\begin{matrix} crs \leftarrow \mathsf{K}(1^n), (x,\pi) \leftarrow \mathsf{P}^*(crs) : \\ x \notin L_R \wedge \mathsf{V}(x,\pi,crs) = 1 \end{matrix}\right] \leq \mathsf{negl}(n).$$

- *(Adaptive) Zero-Knowledge: There exists a* PPT *simulator $S = (S_1, S_2)$, such that for all stateful non-uniform* PPT *adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$, we have*

$$\left[\begin{matrix} \Pr\left[\begin{matrix} crs \leftarrow \mathsf{K}(1^n) \\ (x,w) \leftarrow \mathcal{A}_1(crs) \\ \pi \leftarrow \mathsf{P}(crs,x,w) \end{matrix} : \begin{matrix} (x,w) \in R \wedge \\ \mathcal{A}_2(crs,\pi) = 1 \end{matrix}\right] \\ -\Pr\left[\begin{matrix} (crs,td) \leftarrow S_1(1^n) \\ (x,w) \leftarrow \mathcal{A}_1(crs) \\ \pi \leftarrow S_2(crs,x,td) \end{matrix} : \begin{matrix} (x,w) \in R \wedge \\ \mathcal{A}_2(crs,\pi) = 1 \end{matrix}\right] \end{matrix}\right] \leq \mathsf{negl}(n).$$

We call the NIZK argument perfect zero-knowledge if the above probability equals 0.

The above definition discribes the NIZK argument in the common reference string (CRS) model which is generated by a trusted third party. Using Fiat-Shamir heuristic [28], a Σ-protocol can be transformed into a NIZK argument under the random oracle model: P computes a, applies a random oracle H to a and obtains the challenge $c = \mathsf{H}(a)$, then computes z according to the Σ-protocol and send the proof (a, c, z) to V.

We will construct NIZK in the common reference string model by applying Fiat-Shamir heuristic to a Σ-protocol, which allows us to achieve perfect zero knowledge without relying on a random oracle, though the soundness of our construction is proved in the random oracle model.

2.2 Decentralized Smart Contracts Over Blockchains

A smart contract is a piece of code which is stored in the blockchain network on each participant node. It can be seen as a digital version of a traditional contract. The property of decentralization of blockchain has improved the development of smart contracts. Assume in a payment system which owns the ACOUNT model, user A want to transfer t coins to user B. Then we can deploy the transfer action and some necessary checks in the blockchain as a smart contract to automatically execute the operation in the following way. User A posts a transaction on the blockchain that basically says

> *Transfer t of my coins to B, and σ is a signature of t.*

Being triggered by this message, the smart contract first checks the validity of the signature, and that A has more than t coins, If so does the transfer action and publishes the transation on the blockchain, otherwise it ignores the transation.

In the simplified transaction above, anyone can learn the money t being transfered from A to B (i.e., there is no guarantee in the privacy of users' balance and transaction amount). But we can get around this problem by changing the verification procedure accordingly deployed in the smart contract. Suppose that every user's balance is encrypted with a homomorphic encryption scheme $E(\cdot)$ and saved on the ledger in the form of ciphertext. A could post the transaction as follows.

> *Transfer $E(t)$ of my coins to B, here is a non-interactive zero knowledge proof π to prove the correctness of $E(t)$ and that my balance is larger than t.*

In next section, we will introduce the abstract framework of a decentralized smart contracts system that allows the users to transfer money with privacy of balance and transaction amount and give a concrete construction of its main building block, a NIZK argumment system.

3 NIZK Argument and DSC Scheme

In this section, we introduce the framewrok of a decentralized smart contract (DSC) system with the property of hiding balance and transaction amount and present a new NIZK argument for the two basic statements introduced in section 1 to fulfill the DSC system. We also prove the correctness and security of the NIZK argument. With respect to the "equivalence" statement, the basic idea is that we first construct a Σ-protocol to prove the given two ciphertexts corresponding to some transaction amount own a same plaintext which is encrypted with an HE scheme. Then using Fiat-Shamir heuristic method, we build a NIZK protocol between the two parties. Now the key technique to use is a set membership proof protocol. We get the full NIZK scheme acting as a building block in our DSC system when put the two proofs together. Note that we also put forward a system public parameter generated once serving as common reference string in the NIZK argument which can be reused in other proofs.

3.1 Decentralized Smart Contract System

Suppose a NIZK argument with a prover P and a verifier V, we deploy the verification procedure in the blockchain to

Decentralized Smart Contract System

- **Setup** The algorithm Setup produces alist of system public parameters:
  - input: security parameter n
  - output: system public parameters PP, which also servers as the common reference string of NIZK scheme $crs \leftarrow \mathsf{NIZK.K}(1^n)$
- **PartyInitial** The algorithm PartyInitial generates every user's (say A) information using a homomorphic encryption scheme:
  - input: PP
  - output: $PK_A, SK_A, C_A = \mathsf{Enc}_{PK_A}(t_A)$

  The public key PK_A also links to the address for receiving coins. Only the ciphertext C_A of A's balance t_A is stored in the account book.
- **Transfer** The algorithm Transfer is invoked when some party A transfer t coins to B.
  - input: $PP, t, t_A, PK_A, C_A, PK_B$
  - A generates a transfer statement x, then run $\pi \leftarrow \mathsf{NIZK.P}(crs, x, w)$, and posts (x, π) to the blockchain.
- **Redeem** The algorithm Redeem deployed in the blockchain for automatically transfer will be triggered by the Transfer algorithm.
  - input: PP, x, π
  - if $\mathsf{NIZK.V}(crs, x, \pi) = 1$, then any accounter will find the sender A and the receiver B from the statement x, and publish the transaction, then update A's balance to C_A/C_t and B's balance to $C_B \cdot \hat{C}_t$;
  - otherwise, the accounter ignores it.

Fig. 2. DSC system.

obtain a smart contract which can automatically do the transfer operation. Fig. 2 is a formal description of a DSC system.

Next we present the concrete construction of the NIZK argument system.

3.2 The Construction of NIZK and its Security

Here we only consider two parties A and B playing the role of sender and receiver respectively. Suppose that the plaintext space is $[0, 2^{\mathcal{L}})$, where $\mathcal{L} = u \times l$. Before constructing a concrete NIZK argument, we specify the algorithm of *Setup* and *PartyInitial*:

Setup. Generate a bilinear group $(p, G_1, G_2, G_T, e, g_1, g_2) \leftarrow G_{bp}(1^n)$. Randomly choose $h \leftarrow_{\$} G_1$, then there must be $\omega \in \mathbb{Z}_p$ satisfying $h = g_1^{\omega}$. Let $g_T = e(g_1, g_2)$.

Run $(\mathsf{sk} = \lambda, \mathsf{vk} = g_2^{\lambda}) \leftarrow \mathsf{Sig.Gen}(1^n)$, then compute the signatures of the integers between 0 and $2^u - 1$:

$$\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{2^u-1}) = \left(g_1^{\frac{1}{\lambda}}, g_1^{\frac{1}{\lambda+1}}, \ldots, g_1^{\frac{1}{\lambda+2^u-1}}\right);$$

and the following bilinear maps:

$$\begin{aligned} T &= (T_0, T_1, \ldots, T_{2^u-1}) \\ &= (e(\sigma_0, g_2), e(\sigma_1, g_2), \ldots, e(\sigma_{2^u-1}, g_2)). \end{aligned}$$

Finally, output the public parameter $PP = (p, G_1, G_2, G_T, e, g_1, h, g_2, g_T, \mathsf{vk}, \sigma, T)$

PartyInitial. When some party A with balance t_A enter into this system. We use the homomorphic encryption described in Definition 2 to initialize his account information such as the public key, private key, and encrypted balance:

- Private key: $\mathsf{sk}_A = (x_{A1}, x_{A2}) \in \mathbb{Z}_p^2$;
- Public key: $\mathsf{pk}_A = (X_{A1}, X_{A2}) \in G_1^2$, where $X_{A1} = g_1^{x_{A1}}, X_{A2} = g_1^{x_{A2}}$;
- Encrypted balance: $C_A = \mathsf{PKE.Enc}_{\mathsf{pk}_A}(t_A; (y_1, y_2)) = (C_1 = X_{A1}^{y_1}, C_2 = X_{A2}^{y_2}, C_3 = g_1^{y_1+y_2} \cdot h^{t_A})$.

According the rules of DSC system, next we show how to construct the transfer statement x for party A to send t coins to B. First A gets the ciphertext of t_A from the public ledger, $\tilde{C} = (\tilde{C}_1, \tilde{C}_2, \tilde{C}_3) = (X_{A1}^{\tilde{y}_1}, X_{A2}^{\tilde{y}_2}, g_1^{\tilde{y}_1+\tilde{y}_2} \cdot h^{t_A})^1$, and decrypt it to get t_A. Using randomness $y_1, y_2 \leftarrow_{\$} \mathbb{Z}_p$, A encrypts the transfered coins t under the public key of A and B respectively: $C = (C_1, C_2, C_3) = (X_{A1}^{y_1}, X_{A2}^{y_2}, g_1^{y_1+y_2} \cdot h^t)$; $\hat{C} = (\hat{C}_1, \hat{C}_2, \hat{C}_3 = C_3) = (X_{B1}^{y_1}, X_{B2}^{y_2}, g_1^{y_1+y_2} \cdot h^t)$.

Define the language L proved by P as follows:

The transfer statement $x = (C, \hat{C}, \mathsf{pk}_A, \mathsf{pk}_B, \tilde{C}) \in L$ and the according witness $w = (\mathsf{sk}_A = (x_{A1}, x_{A2}), y_1, y_2, t_A, t)$, such that

(i) $C_i = X_{Ai}^{y_i}$, for $i = 1, 2$;

(ii) $\hat{C}_i = X_{Bi}^{y_i}$, for $i = 1, 2$;

(iii) $C_3 = g_1^{y_1+y_2} \cdot h^t$;

(iv) $\frac{\tilde{C}_3}{C_3} = \tilde{C}_1^{\frac{1}{x_{A1}}} \cdot \tilde{C}_2^{\frac{1}{x_{A2}}} \cdot g_1^{-y_1-y_2} \cdot h^{t_A-t}$;

(v) $t \in [0, 2^{\mathcal{L}}), t' = t_A - t \in [0, 2^{\mathcal{L}})$, where $t = \sum_{j=0}^{l-1} t_j \cdot (2^u)^j$, $t' = \sum_{j=0}^{l-1} t'_j \cdot (2^u)^j$, $0 \le t_j, t'_j < 2^u$;

OR there exists $\omega \in \mathbb{Z}_p$, such that

(vii) $h = g_1^{\omega}$.

Proof generation by P.

1. Taking PP as common input, prover A generates a NIZK proof for the above statement with private input $(\mathsf{sk}_A, y_1, y_2, t_A, t)$ in the following way:

   Σ-protocols can be used to prove Equation (i-iv). Equation (v) can be proved by utilizing the range proof in [13].

   Randomly sample $r_1, r_2, \ell, k \leftarrow_{\$} \mathbb{Z}_p$, compute for $i = 1, 2$:

   $$R_i = X_{Ai}^{r_i}; \hat{R}_i = X_{Bi}^{r_i};$$

   For $j \in [0, l)$, randomly sample $v_j, v'_j, s_j, w_j, q_j, m_j \leftarrow_{\$} \mathbb{Z}_p$, then compute:

   $$V_j = \sigma_{t_j}^{v_j}, V'_j = \sigma_{t'_j}^{v'_j};$$

   $$D_1 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot s_j}\right) \cdot g_1^{r_1+r_2};$$

   $$D_2 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot w_j}\right) \cdot \tilde{C}_1^{\ell} \cdot \tilde{C}_2^{k} \cdot g_1^{-r_1-r_2};$$

   $$a_j = T_{t_j}^{-s_j \cdot v_j} \cdot g_T^{q_j}, a'_j = T_{t'_j}^{-w_j \cdot v'_j} \cdot g_T^{m_j};$$

[1] Note that A probably does not know $\tilde{y}_1, \tilde{y}_2$ since the ciphertexts on the ledger may be timely updated many times.

Randomly choose $\hat{c} \leftarrow_{\$} \mathbb{Z}_p, \hat{z} \leftarrow_{\$} \mathbb{Z}_p$, and set $\alpha = g_1^{\hat{z}} / h^{\hat{c}}$.

Denote $(R_1, R_2, \hat{R}_1, \hat{R}_2, \{V_j, V_j'\}_{j=0}^{l-1}, D_1, D_2, \{a_j, a_j'\}_{j=0}^{l-1}, \alpha)$ by a. We obtain the challenge via computing:

$$\tilde{c} = \mathsf{H}(a); c = \tilde{c} + \hat{c};$$

where H represents a random oracle which can be instantiated by a secure hash function.

Compute (all modulo p):

$$\begin{aligned} z_1 &= r_1 - c \cdot y_1; & z_2 &= r_2 - c \cdot y_2; \\ z_{v_j} &= q_j - c \cdot v_j; & z_{v_j'} &= m_j - c \cdot v_j'; \\ z_{t_j} &= s_j - c \cdot t_j; & z_{t_j'} &= w_j - c \cdot t_j'; \\ z_\ell &= \ell - \frac{c}{x_{A1}}; & z_k &= k - \frac{c}{x_{A2}}; \end{aligned}$$

Finally, A sends to B the proof $\pi = (a, c, z)$, where $z = (z_1, z_2, \{z_{v_j}, z_{v_j'}\}_{j=0}^{l-1}, \{z_{t_j}, z_{t_j'}\}_{j=0}^{l-1}, z_\ell, z_k, \hat{z})$.

Proof verification by V. Upon receiving a proof π, the verifier V parses π into the form as above, then computes c. With the common input PP, $\forall i = 1, 2; j \in [0, l)$, V checks whether the following conditions hold:

$$R_i = {C_i}^c \cdot {X_{A_i}}^{z_i}; \tag{1}$$

$$\hat{R}_i = {\hat{C}_i}^c \cdot {X_{B_i}}^{z_i} \tag{2}$$

$$D_1 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot z_{t_j}} \right) \cdot C_3^c \cdot g_1^{z_1 + z_2}; \tag{3}$$

$$D_2 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot z_{t_j'}} \right) \cdot \left(\frac{\tilde{C}_3}{C_3} \right)^c \cdot \tilde{C}_1^{z_\ell} \cdot \tilde{C}_2^{z_k} \cdot g_1^{-z_1 - z_2}; \tag{4}$$

$$\begin{aligned} a_j &= e(V_j, \mathsf{vk})^c \cdot e(V_j, g_2)^{-z_{t_j}} \cdot g_T^{z_{v_j}}; \\ a_j' &= e(V_j', \mathsf{vk})^c \cdot e(V_j', g_2)^{-z_{t_j'}} \cdot g_T^{z_{v_j'}}; \end{aligned} \tag{5}$$

$$g_1^{\hat{z}} = \alpha \cdot h^{\hat{c}}; \tag{6}$$

Theorem 2. *Assuming the DLIN, q-SDH assumptions, the protocol described above is a* NIZK *argument with perfect completeness, perfect zero-knowledge and computational soundness in the RO model. Furthermore, perfect zero-knowledge holds in the standard CRS model.*

Proof. We prove each direction separately.

Perfect Completeness. Perfect completeness follows by direct verification.

Soundness. Assuming the unforgeability of the Boneh-Boyen signature which is based on q-SDH assumption, we prove the soundness under the random oracle model. If a PPT prover P^* generates an accepted argument $\pi = (a, c, z)$ for an invalid statement, where $a = \big(R_1, R_2, \hat{R}_1, \hat{R}_2, \{V_j, V_j'\}_{j=0}^{l-1}, D_1, D_2, \{a_j, a_j'\}_{j=0}^{l-1}, \alpha\big)$ and $z = (z_1, z_2, \{z_{v_j}, z_{v_j'}\}_{j=0}^{l-1}, \{z_{t_j}, z_{t_j'}\}_{j=0}^{l-1}, z_\ell, z_k, \hat{z})$

Then, we construct such an extractor Ext: Upon seeing the argument, Ext rewinds P^* to the oracle query $H(a)$ that returned c. It then reprogram the random oracle such that $c' = H(a)$ with $c \neq c'$ and continue the execution of P^* with the modified random oracle. In expected polynomial time, another valid argument appears:

$$\pi' = (a, c', z' = (z_1', z_2', \{z_{v_j}', z_{v_j'}'\}_{j=0}^{l-1}, \\ \{z_{t_j}', z_{t_j'}'\}_{j=0}^{2}, z_\ell', z_k', \hat{z}')).$$

The witness can be extracted by computing (for $i = 0, 1; j \in [0, l)$):

$$y_i = \frac{z_i - z_i'}{c' - c}, t_j = \frac{z_{t_j} - z_{t_j}'}{c' - c}, t_j' = \frac{z_{t_j'} - z_{t_j'}'}{c' - c}, \\ x_{A1} = \frac{c' - c}{z_\ell - z_\ell'}, x_{A2} = \frac{c' - c}{z_k - z_k'}.$$

Conditioned on the extracted witness, if $t \notin [0, 2^{\mathcal{L}})$ or $t' \notin [0, 2^{\mathcal{L}})$, then we can break the Boneh-Boyen signature in a weak chosen message attack model with non-negligible probability, taking P^* as a subroutine.

Perfect Zero-Knowledge. Instead of using the standard Fiat-Shamir heuristic method, we prove perfect zero-knowlege via constructing a simulatot Sim to prove statement $h = g_1^w$ without relying on a random oracle, see Fig. 3.

Parse the argument into 3 parts:

$$\pi = \big(a = (R_1, R_2, \hat{R}_1, \hat{R}_2, \{V_j, V_j'\}_{j=0}^{l-1}, D_1, D_2, \{a_j, a_j'\}_{j=0}^{l-1}, \alpha), \\ c, z = (z_1, z_2, \{z_{v_j}, z_{v_j'}\}_{j=0}^{l-1}, \{z_{t_j}, z_{t_j'}\}_{j=0}^{l-1}, z_\ell, z_k, \hat{z})\big).$$

For the sake of clarity and convenience, we denote the simulated argument by

$$\boldsymbol{\pi} = \big(\mathfrak{a} = (\mathcal{R}_1, \mathcal{R}_2, \hat{\mathcal{R}}_1, \hat{\mathcal{R}}_2, \{\mathcal{V}_j, \mathcal{V}_j'\}_{j=0}^{l-1}, \mathcal{D}_1, \mathcal{D}_2, \{\mathfrak{a}_j, \mathfrak{a}_j'\}_{j=0}^{l-1}, \boldsymbol{\alpha}), \\ \mathfrak{c}, \mathfrak{z} = (\mathfrak{z}_1, \mathfrak{z}_2, \{\mathfrak{z}_{v_j}, \mathfrak{z}_{v_j'}\}_{j=0}^{l-1}, \{\mathfrak{z}_{t_j}, \mathfrak{z}_{t_j'}\}_{j=0}^{l-1}, \mathfrak{z}_\ell, \mathfrak{z}_k, \hat{\mathfrak{z}})\big).$$

Observe that $\hat{c} \leftarrow_{\$} \mathbb{Z}_p$ is independent of a, $c = \mathsf{H}(a) + \hat{c}$ is uniformly distributed in $\mathbb{Z}_p$, and that $\mathfrak{c}$ is also chosen from $\mathbb{Z}_p$ at random in the simulation. Hence, the distribution $\{c\}$ is identical to $\{\mathfrak{c}\}$:

$$\{c\} \equiv \{\mathfrak{c}\}. \tag{7}$$

Set $\mathscr{C} = \{c\} = \{\mathfrak{c}\}$. Conditioned on (7), given $\bar{c} \in \mathscr{C}$, for every $\rho \in \mathbb{Z}_p$, since $\hat{z}$, r_1, r_2, $\ell, k, q_j, m_j, s_j, w_j, v_j, v_j' \leftarrow_{\$} \mathbb{Z}_p$ where $j \in [0, l)$, and they are all independent of c, then for any element $\tilde{z}$ in z, we have

$$\Pr[\tilde{z} = \rho | c = \bar{c}] = \frac{1}{p}$$

In the simulated argument, under the same condition, given the value $\mathfrak{z}_1, \mathfrak{z}_2, \mathfrak{z}_\ell$, $\mathfrak{z}_k, \mathfrak{z}_{v_j}, \mathfrak{z}_{v_j'}, \mathfrak{z}_{t_j}, \mathfrak{z}_{t_j'}, \mu \leftarrow_{\$} \mathbb{Z}_p$ which are independent of $\mathfrak{c}$, then for every element $\tilde{\mathfrak{z}}$ in $\mathfrak{z}$, we have

$$\Pr[\tilde{\mathfrak{z}} = \rho | \mathfrak{c} = \bar{c}] = \frac{1}{p}$$

Set $\mathscr{Z} = \{z^1, z^2, \{z_j^3, z_j^4\}_{j=0}^{l-1}, \{z_j^5, z_j^6\}_{j=0}^{l-1}, z^7, z^8, z^9 : z_i \leftarrow_{\$} \mathbb{Z}_p, i \in [10]\}$. Given $\bar{c} \leftarrow_{\$} \mathscr{C}$, for every $\bar{z} \in \mathscr{Z}$,

$$\Pr[z = \bar{z} | c = \bar{c}] = \Pr[\mathfrak{z} = \bar{z} | \mathfrak{c} = \bar{c}]. \tag{8}$$

Conditioned on (8), given $\bar{c} \in \mathscr{C}, \bar{z} \in \mathscr{Z}$, following from the verification strategy, the messages $R_1, R_2, D_1, D_2,$

1) Just do like the procedure of **Setup** and produce:

$$PP = (p, G_1, G_2, G_T, e, g_1, h, g_2, g_T, \mathsf{vk}, \sigma, T);$$
$$td = \omega;$$

where $h = g_1^{\omega}$.

2) Randomly choose $t, t' \xleftarrow{\$} [0, 2^{\mathcal{L}})$; $v_j, v'_j \xleftarrow{\$} \mathbb{Z}_p$, and write t, t' in base-2^u:

$$t = \sum_{j=0}^{l-1} (2^u)^j \cdot t_j, t' = \sum_{j=0}^{l-1} (2^u)^j \cdot t'_j;$$

then set $V_j = \sigma_{t_j}^{v_j}, V'_j = \sigma_{t'_j}^{v'_j}$, where $j \in \{0, 1, \cdots, l-1\}$.

3) Choose $c, z_1, z_2, z_{v_j}, z_{v'_j}, z_{t_j}, z_{t'_j}, z_\ell, z_k, \mu \xleftarrow{\$} \mathbb{Z}_p$, and compute ($i = 1, 2; j = 0, 1, \cdots, l-1$):

$$R_i = C_i^c \cdot X_{A_i}^{z_i};$$
$$\hat{R}_i = \hat{C}_i^{\,c} \cdot X_{B_i}^{z_i};$$
$$D_1 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot z_{t_j}} \right) \cdot C_3^c \cdot g_1^{z_1 + z_2};$$
$$D_2 = \prod_{j=0}^{l-1} \left(h^{(2^u)^j \cdot z_{t'_j}} \right) \cdot \left(\frac{\tilde{C}_3}{C_3} \right)^c \cdot \tilde{C}_1^{z_\ell} \cdot \tilde{C}_2^{z_k} \cdot g_1^{-z_1 - z_2};$$
$$a_j = e(V_j, \mathsf{vk})^c \cdot e(V_j, g_2)^{-z_{t_j}} \cdot g_T^{z_{v_j}};$$
$$a'_j = e(V'_j, \mathsf{vk})^c \cdot e(V'_j, g_2)^{-z_{t'_j}} \cdot g_T^{z_{v'_j}};$$
$$\alpha = g_1^{\mu}.$$

4) Denote the values obtained in 3) as a and compute $\tilde{c} = \mathsf{H}(a)$, then set $\hat{c} = c - \tilde{c}, \hat{z} = \mu + \hat{c} \cdot \omega$. Finally, output the simulated argument:

$$\pi = \Big(R_1, R_2, \hat{R}_1, \hat{R}_2, \{V_j, V'_j\}_{j=0}^{l-1}, D_1, D_2, \{a_j, a'_j\}_{j=0}^{l-1}, \alpha, c, z_1, z_2, \{z_{v_j}, z_{v'_j}\}_{j=0}^{l-1}, \{z_{t_j}, z_{t'_j}\}_{j=0}^{2}, z_\ell, z_k, \hat{z} \Big).$$

Fig. 3. Simulator for the New NIZK argument.

a_j, a'_j, α in π are determined where $j \in [0, l)$. For $\{V_j, V'_j\}$, we have

$$\Pr[V_j = \mathfrak{g} | c = \bar{c}, z = \bar{z}]$$
$$= \Pr[\sigma_{t_j}^{v_j} = \mathfrak{g} | c = \bar{c}, z = \bar{z}] = \frac{1}{p};$$
$$\Pr[V'_j = \mathfrak{g} | c = \bar{c}, z = \bar{z}]$$
$$= \Pr[\sigma_{t'_j}^{v'_j} = \mathfrak{g} | c = \bar{c}, z = \bar{z}] = \frac{1}{p};$$

where $\mathfrak{g} \leftarrow_{\$} G_1$, since $v_j, v'_j \leftarrow_{\$} \mathbb{Z}_p$.

Note that in the simulated argument, for fixed $\bar{c} \in \mathscr{C}$, $\bar{z} \in \mathscr{Z}$, the messages $\mathcal{R}_1, \mathcal{R}_2, \mathcal{D}_1, \mathcal{D}_2, \mathfrak{a}_j, \mathfrak{a}'_j, \boldsymbol{\alpha}$ are determined according to Sim. For arbitrary $\mathfrak{g} \in G_1, j \in [0, l)$,

$$\Pr[\mathcal{V}_j = \mathfrak{g} | \mathfrak{c} = \bar{c}, \mathfrak{z} = \bar{z}]$$
$$= \Pr[\sigma_{t_j}^{v_j} = \mathfrak{g} | \mathfrak{c} = \bar{c}, \mathfrak{z} = \bar{z}] = \frac{1}{p};$$
$$\Pr[\mathcal{V}'_j = \mathfrak{g} | \mathfrak{c} = \bar{c}, \mathfrak{z} = \bar{z}]$$
$$= \Pr[\sigma_{t'_j}^{v'_j} = \mathfrak{g} | \mathfrak{c} = \bar{c}, \mathfrak{z} = \bar{z}] = \frac{1}{p};$$

since $v_j, v'_j \leftarrow_{\$} \mathbb{Z}_p$.

Set $\mathscr{A} = \{a^1, a^2, \{a_j^3, a_j^4\}_{j=0}^{l-1}, a^5, a^6, \{a_j^7, a_j^8\}_{j=0}^{l-1}, a^9 : a^1, a^2, a^3, a^4, a^5, a_j^6, a_j^7, a^9 \leftarrow_{\$} G_1, a_j^7, a_j^8 \leftarrow_{\$} G_T\}$. Thus, given $\bar{c} \in \mathscr{C}, \bar{z} \in \mathscr{Z}$, for arbitrary $\bar{a} \in \mathscr{A}$,

$$\Pr[a = \bar{a} | c = \bar{c}, z = \bar{z}] = \Pr[\mathfrak{a} = \bar{a} | \mathfrak{c} = \bar{c}, \mathfrak{z} = \bar{z}]. \tag{9}$$

Combine (8) and (9), we conclude that for any non-uniform PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$,

$$\Pr\begin{bmatrix} (x, w) \leftarrow \mathcal{A}_1(1^n) \quad (x, w) \in R \\ : \\ (a, c, z) \leftarrow \mathsf{P}(x, w, PP) \quad \mathcal{A}_2(a, c, z) = 1 \end{bmatrix}$$
$$= \Pr\begin{bmatrix} (x, w) \leftarrow \mathcal{A}_1(1^n) \quad (x, w) \in R \\ : \\ (a, c, z) \leftarrow \mathsf{Sim}(x) \quad \mathcal{A}_2(a, c, z) = 1 \end{bmatrix}.$$

(Perfect) Zero-knowledge property is obtained. □

3.3 An Optimized Verifier

Instead of verifying Equation (5) with computing $4l$ pairing computations, V randomly selects $2l$ elements $\{d_j\}_{j=0}^{l-1}$, $\{d'_j\}_{j=0}^{l-1}$ from $\mathbb{Z}_p$, and checks whether the following equation holds:

$$\prod_{j=0}^{l-1} a_j^{d_j} \cdot \prod_{j=0}^{l-1} (a'_j)^{d'_j} = e\left(\prod_{j=0}^{l-1} V_j^{cd_j} \cdot \prod_{j=0}^{l-1} (V'_j)^{cd'_j}, \mathsf{vk} \right) \cdot e\left(\prod_{j=0}^{l-1} V_j^{-z_{t_j} d_j} \cdot \prod_{j=0}^{l-1} (V'_j)^{-z_{t'_j} d'_j}, g_2 \right) \cdot g_T^{\sum_{j=0}^{l-1} z_{v_j} d_j + \sum_{j=0}^{l-1} z_{v'_j} d'_j}. \tag{10}$$

Equation (10) only computes 2 pairing computations, which is more efficient than (5). But it also gains a probability to the soundness error. Next we claim this probability is negligible.

- (5) ⇒ (10): Upon substitution of all the values of $\{a_j\}_{j=0}^{l-1}, \{a'_j\}_{j=0}^{l-1}$ in (5), Equation (10) is obtained.
- (10) ⇒ (5): Consider equation (10):

$$Right\_Side = \prod_{j=0}^{l-1} e(V_j^{cd_j}, \mathsf{vk}) \cdot \prod_{j=0}^{l-1} e((V'_j)^{cd'_j}, \mathsf{vk}) \cdot \prod_{j=0}^{l-1} e(V_j^{-z_{t_j} d_j}, g_2) \cdot \prod_{j=0}^{l-1} e((V'_j)^{-z_{t'_j} d'_j}, g_2) \cdot \prod_{j=0}^{l-1} g_T^{z_{v_j} d_j} \cdot \prod_{j=0}^{l-1} g_T^{z_{v'_j} d'_j}$$
$$= \prod_{j=0}^{l-1} \Big(e(V_j, \mathsf{vk})^{cd_j} \cdot e(V'_j, \mathsf{vk})^{cd'_j} \cdot e(V_j, g_2)^{-z_{t_j} d_j} \cdot e(V'_j, g_2)^{-z_{t'_j} d'_j} \cdot g_T^{z_{v_j} d_j} \cdot g_T^{z_{v'_j} d'_j} \Big)$$
$$= \prod_{j=0}^{l-1} \Big(\big(e(V_j, \mathsf{vk})^c \cdot e(V_j, g_2)^{-z_{t_j}} \cdot g_T^{z_{v_j}} \big)^{d_j} \cdot \big(e(V'_j, \mathsf{vk})^c \cdot e(V'_j, g_2)^{-z_{t'_j}} \cdot g_T^{z_{v'_j}} \big)^{d'_j} \Big);$$

TABLE 1
Comparison of Our NIZK Scheme to Existing ZKP Systems, Where C is the Size of the Circuit With Depth d, and inp is the Size of its Input

		libsnark	Hyrax	Bulletproof	libSTARK	This Paper
Theoretical	Public Parameter Size	$O(C)$	-	-	-	$O(2^u)$
	Proof Size	$O(1)$	$O(d\log C+\sqrt{inp})$	$O(\log C)$	$O(\log^2 C)$	$O(l)$
Practical	Setup	311s	-	-	-	8.7s
	Proof	119s	103s	6144s	1031s	64.97ms
	Verify	5.4ms	26s	391s	68ms	48.96ms
	Public Parameter Size	337.25MB	-	-	-	0.44MB
	Proof Size	288B	3.2MB	2115B	735KB	3680B

$$Left\_Side = \prod_{j=0}^{l-1}\left((a_j)^{d_j}(a'_j)^{d'_j}\right);$$

if $Left\_Side = Right\_Side$, two cases occur:

1) $\forall j \in [0,l), a_j = e(V_j, \mathsf{vk})^c \cdot e(V_j, g_2)^{-z_{t_j}} \cdot g_T^{z_{v_j}}, a'_j = e(V'_j, \mathsf{vk})^c \cdot e(V'_j, g_2)^{-z_{t'_j}} \cdot g_T^{z_{v'_j}}$, which implies the correctness of (5).
2) There exist some d_j or $d'_j = 0$, which can lead to $a_j \neq e(V_j, \mathsf{vk})^c \cdot e(V_j, g_2)^{-z_{t_j}} \cdot g_T^{z_{v_j}}$ or $a'_j \neq e(V'_j, \mathsf{vk})^c \cdot e(V'_j, g_2)^{-z_{t'_j}} \cdot g_T^{z_{v'_j}}$ for some $j \in [0,l)$. This case happens with probability

$$\sum_{i=1}^{2l}\left(C_{2l}^{i}\frac{1}{p^i}\left(1-\frac{1}{p}\right)^{2l-i}\right)$$
$$=1-\left(1-\frac{1}{p}\right)^{2l} < \frac{2l}{p} < \frac{l}{2^{n-1}};$$

that is , a negilible probability, since p is a prime with n bits.

Overall, with an overwhelming probability $1-\frac{l}{2^{n-1}}$, Equation (5) ⇔ (10).

4 Evaluation

We evaluated our NIZK argument system on a personal computer. In order to show the superriorty of our scheme intuitively, we also took a comparison with several state-of-the-art zero knowledge proof systems.

4.1 Comparison

We compare our NIZK scheme with the folowing systems in both theoretical and practical aspects: zkSNARK, Hyrax, Bulletproofs and zkSTARK. In theoretical aspect, we focus on the computational complexity. The system parameter PP generated once for the proof is of the size $|G_2| + 2^u \cdot (|G_1| + |G_T|)$ (we omit the bilinear group parameters, and denote by $|G_1|$ the size of an element in G_1, likewise with $|G_2|, |G_T|$, and $|\mathbb{Z}_p|$), while the size of the whole proof is $(2l+5)\cdot|G_1| + 2l\cdot|G_T| + (4l+6)\cdot|\mathbb{Z}_p|$. Second, in the practical performance, we re-implemented the prior works for our personal statements. In this paper, our goal is to propose an efficient zero knowledge proof suitable for lightweight devices. Hence, we run experiments using coding language C++ on Linux (Deepin 15.11, 64 bits) with an Inter (R) Core(TM) i7-4770 CPU of 3.40 GHz and 16 GB RAM. However, the Bulletproofs, zkSNARK and zkSTARK provers are memory intensive, so we evaluate them on a server with 60 GB RAM and 16 cores at 2.80 GHz. More detailly, we use jsnark [29] to generate our circuit, which produces a zero knowledge proof using the libsnark [30] backend. For zkSTARK we base on their open-sourced implementation libSTARK [31]. We use the implementation at [32] to execute the experiments of Hyrax and Bulletproofs.

TABLE 2

Choice	Public Parameter	Proof
$l=1, u=30$	$\|G_2\| + 2^{30}(\|G_1\| + \|G_T\|)$	$7\|G_1\| + 2\|G_T\| + 10\|\mathbb{Z}_p\|$
$l=2, u=15$	$\|G_2\| + 2^{15}(\|G_1\| + \|G_T\|)$	$9\|G_1\| + 4\|G_T\| + 14\|\mathbb{Z}_p\|$
$l=3, u=10$	$\|G_2\| + 2^{10}(\|G_1\| + \|G_T\|)$	$11\|G_1\| + 6\|G_T\| + 18\|\mathbb{Z}_p\|$
$l=5, u=6$	$\|G_2\| + 2^{6}(\|G_1\| + \|G_T\|)$	$15\|G_1\| + 10\|G_T\| + 26\|\mathbb{Z}_p\|$
...	...	...

When implementing our NIZK scheme, we consider the message space as $[0, 2^{30})$, that is to say, $(2^u)^l = 2^{30}$. From the result shown in Table 1, the proof size grows linear with l, while the public parameter is linear with 2^u. Because the setup process is only executed once, we need find a tradeoff between the proof size and the public parameter size. From the result shown in Table 2, we choose $u=10$ and $l=3$, since $u=30$ and $u=15$ result in a more expensive setup process and a more lengthy public parameter, but $l=5$ indicates a larger proof size.

As shown in Table 1, comparing to other works, our NIZK scheme dramatically improves the prover running time by thousands of times. Our verifier is 9x slower than libsnark, which only runs in 5.4ms. Comparing to Hyrax, Bulletproof and libSTARK , the verification of our NIZK scheme is 531x, 7986x and 1.4x faster respectively. Our proof size is larger than libsnark, which is 288 bytes for all circuits, and Bulletproof, which is 2115 bytes for our statement. The proof size in our scheme is 3680 bytes, which is much better than Hyrax, libSTARK. Among all the systems, only libsnark and our NIZK scheme require trusted setup. Our scheme only needs to execute this setup once at the cost of 8.7s to generate 0.44 MB public parameter, while libsnark would take 311s to produce the public parameter with 337.25 MB size.

5 Conclusion

In this paper, we put forward a homomorphic encryption scheme and construct a contrete NIZK scheme to prove the validity of transactions. In our NIZK argument system, the public parameter serves as the common reference string which is only generated once for multi proofs. With respect to the security, we can achieve the zero-knowledge property in the standard CRS model, while the soundness can be obtained under the RO model. Based on the NIZK scheme, we describe a framework of a decentralized smart contract system with balance and transaction amount hiding under the Account model. We also demonstrate the practical performence of our NIZK scheme on a personal computer. The result gives our confidence in applying our scheme in practice.

Acknowledgments

We thank the anonymous reviewers for their invaluable suggestions and comments. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802500, Grant 2017YFB0802005, in part by the National Nature Science Foundation of China under Grant 61772521, Grant 61602046, Grant 61602045, Grant U1536205, Grant 61972294, Grant 61932016, in part by the Key Research Program of Frontier Sciences, CAS under Grant QYZDB-SSW-SYS035, in part by the the Open Project Program of the State Key Laboratory of Cryptology, in part by the Young Elite Scientists Sponsorship Program by CAST under Grant 2016QNRC001.

References

[1] S. Nakamoto, “Bitcoin: A peer-to-peer electronic cash system,” 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[2] H. Kopp, D. Mödinger, F. Hauck, F. Kargl, and C. Bösch, “Design of a privacy-preserving decentralized file storage with financial incentives,” in *Proc. IEEE Eur. Symp. Secur. Privacy Workshops*, 2017, pp. 14–22.

[3] X. Liang, S. Shetty, D. K. Tosh, C. A. Kamhoua, K. A. Kwiat, and L. Njilla, “Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability,” in *Proc. 17th IEEE/ACM Int. Symp. Cluster Cloud Grid Comput.*, 2017, pp. 468–477.

[4] Q. Xia, E. B. Sifah, K. O. Asamoah, J. Gao, X. Du, and M. Guizani, “Medshare: Trust-less medical data sharing among cloud service providers via blockchain,” *IEEE Access*, vol. 5, pp. 14757–14767, 2017.

[5] L. Xu, N. Shah, L. Chen, N. Diallo, Z. Gao, Y. Lu, and W. Shi, “Enabling the sharing economy: Privacy respecting contract based on public blockchain,” in *Proc. ACM Workshop Blockchain Cryptocurrencies Contracts*, 2017, pp. 15–21.

[6] A. Lei, H. Cruickshank, Y. Cao, P. Asuquo, C. P. A. Ogah, and Z. Sun, “Blockchain-based dynamic key management for heterogeneous intelligent transportation systems,” *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1832–1843, Dec. 2017.

[7] Y. Omran, M. Henke, R. Heines, and E. Hofmann, “Blockchain-driven supply chain finance: Towards a conceptual framework from a buyer perspective,” 2017. [Online]. Available: https://www.alexandria.unisg.ch/251095/

[8] G. Maxwell, “Confidential transactions,” 2015. Accessed: May 9, 2016, [Online]. Available: https://people.xiph.org/~greg/confidential_values.txt

[9] E. Ben-Sasson *et al.*, “Zerocash: Decentralized anonymous payments from bitcoin,” in *Proc. IEEE Symp. Security Privacy*, 2014, pp. 459–474.

[10] N. V. Saberhagen, “Cryptonote v 2.0,” 2013. [Online]. Available: https://bytecoin.org/downloads/whitepaper.pdf

[11] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum Project Yellow Paper*, vol. 151, pp. 1–32, 2014.

[12] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell, “Bulletproofs: Short proofs for confidential transactions and more,” in *Proc. IEEE Symp. Security Privacy*, 2018, pp. 315–334.

[13] J. Camenisch, R. Chaabouni, and A. Shelat, “Efficient protocols for set membership and range proofs,” in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2008, pp. 234–252.

[14] T. P. Pedersen *et al.*, “Non-interactive and information-theoretic secure verifiable secret sharing,” in *Proc. Annu. Int. Cryptol. Conf.*, 1991, vol. 91, pp. 129–140.

[15] J. K. Liu, V. K. Wei, and D. S. Wong, “Linkable spontaneous anonymous group signature for ad hoc groups,” in *Proc. Australas. Conf. Inf. Secur. Privacy*, 2004, vol. 4, pp. 325–335.

[16] S. Noether *et al.*, “Ring confidential transactions,” *Ledger*, vol. 1, pp. 1–18, 2016.

[17] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, “SNARKs for C: Verifying program executions succinctly and in zero knowledge,” in *Proc. Annu. Cryptol. Conf.*, 2013, pp. 90–108.

[18] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, “Succinct non-interactive zero knowledge for a von neumann architecture,” in *Proc. 23rd USENIX Secur. Symp.*, 2014, pp. 781–796.

[19] E. Ben-Sasson, I. Bentov, Y. Horesh, and M. Riabzev, “Scalable, transparent, and post-quantum secure computational integrity,” *IACR Cryptol. ePrint Archive*, vol. 2018, 2018, Art. no. 46.

[20] R. S. Wahby, I. Tzialla, A. Shelat, J. Thaler, and M. Walfish, “Doubly-efficient zksnarks without trusted setup,” in *Proc. IEEE Symp. Security Privacy*, May 2018, pp. 926–943.

[21] R. Cramer and I. Damgård, “Zero-knowledge proofs for finite field arithmetic, or: Can zero-knowledge be for free?,” in *Proc. Annu. Int. Cryptol. Conf.*, 1998, pp. 424–441.

[22] S. Goldwasser, Y. T. Kalai, and G. N. Rothblum, “Delegating computation: Interactive proofs for muggles,” *J. ACM*, vol. 62, pp. 27:1–27:64, Sep. 2015.

[23] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, “Hawk: The blockchain model of cryptography and privacy-preserving smart contracts,” in *Proc. IEEE Symp. Security Privacy*, 2016, pp. 839–858.

[24] D. Boneh, B. Lynn, and H. Shacham, “Short signatures from the weil pairing,” in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 514–532.

[25] D. Boneh, X. Boyen, and H. Shacham, “Short group signatures,” in *Proc. Annu. Int. Cryptol. Conf.*, 2004, vol. 3152, pp. 41–55.

[26] S. Goldwasser and S. Micali, “Probabilistic encryption & how to play mental poker keeping secret all partial information,” in *Proc. 14th Annu. ACM Symp. Theory Comput.*, 1982, pp. 365–377.

[27] D. Boneh and X. Boyen, “Short signatures without random oracles,” in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 2004, vol. 3027, pp. 56–73.

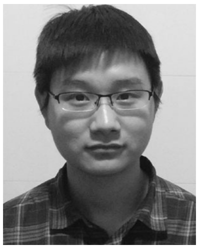
[28] A. Fiat and A. Shamir, “How to prove yourself: Practical solutions to identification and signature problems,” in *Proc. Conf. Theory Appl. Cryptographic Techn.*, 1986, pp. 186–194.

[29] jsnark, 2015. [Online]. Available: https://github.com/akosba/jsnark

[30] libsnark, 2014. [Online]. Available: https://github.com/scipr-lab/libsnark.git

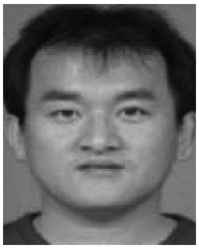
[31] libSTARK, 2018. [Online]. Available: https://github.com/elibensasson/libSTARK.git

[32] hyraxZK, 2018. [Online]. Available: https://github.com/hyraxZK/hyraxZK.git

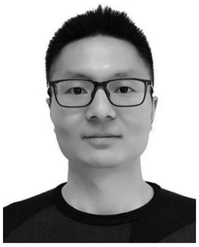


Shunli Ma received the BS degree in computer science and technology (network and information security) from Jilin University, Jilin, China, in 2014. He is currently working towards the PhD degree in the Institute of Information Engineering, Chinese Academy of Sciences and the School of Cyber Security, University of Chinese Academy of Sciences. His research interests include cryptographic protocols, especially non-interactive zero knowledge protocol, and verifiable computation protocol.

Yi Deng received the PhD degree from the Institute of Software, Chinese Academy of Sciences, in 2008. He is currently a professor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences. His research interets include theoretical cryptography, including zero knowledge proofs, security reduction methodology, and the round/communication/computational complexity of cryptographic protocols and their applications in cryptocurrency and blockchain.



Debiao He received the PhD degree in applied mathematics from the School of Mathematics and Statistics, Wuhan University, in 2009. He is currently a professor with the School of Cyber Science and Engineering, Wuhan University. His main research interests include cryptography and information security, in particular, cryptographic protocols.



Jiang Zhang received the PhD degree in information security from the Institute of Software, Chinese Academy of Sciences, in 2015. He is currently an associate researcher with the State Key Laboratory of Cryptology. His research interests include (post-quantum) cryptography and data privacy, particularly in provable security, public-key encryption, key establishment, fully homomorphic encryption, multiparty computation, and lattice-based cryptography.

Xiang Xie received the PhD degree from the Institute of Software, Chinese Academy of Sciences, in 2015. He is currently working at Juzix Technology Co. Ltd. His research interests include public-key cryptography, lattice-based cryptography, block-chain security, and multi-party computation.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.