Accepted Manuscript

Designated-verifier proof of assets for bitcoin exchange using elliptic curve cryptography

Huaqun Wang, Debiao He, Yimu Ji

PII: S0167-739X(17)31350-X
DOI: http://dx.doi.org/10.1016/j.future.2017.06.028
Reference: FUTURE 3526

To appear in: *Future Generation Computer Systems*

Received date : 14 December 2016
Revised date : 10 April 2017
Accepted date : 25 June 2017

Please cite this article as: H. Wang, D. He, Y. Ji, Designated-verifier proof of assets for bitcoin exchange using elliptic curve cryptography, *Future Generation Computer Systems* (2017), http://dx.doi.org/10.1016/j.future.2017.06.028

Designated-Verifier Proof of Assets for Bitcoin Exchange Using Elliptic Curve Cryptography

Huaqun Wang[1,2]

1. Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing, China
2. State Key Laboratory of Information Security,Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China

Debiao He

State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China

Yimu Ji[1,2,3]

1.School of Computer Science, Nanjing University of Posts and Telecommunications,Nanjing, China
2.Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nan-jing University of Posts and Telecommunications, Nanjing, China
3.College of Software, Nanjing College of Information Technology, Nanjing, China

Abstract

Based on the core technique of blockchain, bitcoin is designed for the first time. Bitcoin is a digital currency and a payment system. The blockchain is a digital ledger that records every bitcoin transaction that has ever occurred. The users' privacy is preserved in the bitcoin exchange by using the blockchain. In some application scenarios, it is important to show the buyer's assets strength in order to avoid the troublemakers. At the same time, it is also necessary to preserve the buyer's assets privacy. In this paper, we propose the novel concept of DV-PoA (designated-verifier proof of assets) for bitcoin exchange. Since bitcoin exchange's signature takes use of the elliptic curve cryptography, we design the first concrete DV-PoA scheme by using elliptic curve cryptography in order to be consistent with it. Then, we prove the security of the proposed DV-PoA scheme. After that, we analyze its efficiency from the two cases: theory and implementation. Our analysis shows that the designed DV-PoA scheme is

provably secure and efficient.

Keywords: Bitcoin, blockchain, designated-verifier signature, elliptic curve cryptography

1. Introduction

Cryptography currencies bring the users many advantages since they made the transactions be electronically authorized, cleared and settled. Although there exist many needs for cryptographic currency, many researchers launch the corresponding research in this field for decades of years [1][2][3][4]. In 2009, bitcoin was proposed and deployed [5]. In the past few years, bitcoin has achieved unprecedented success. Bitcoin transactions are executed with low (almost zero) fees. It is often called the first cryptocurrency, although prior systems existed, and it is more correctly described as the first decentralized digital currency. Bitcoin is both a cryptocurrency and an electronic payment system. Now, bitcoin is the largest of its kind in terms of total market value.

According to their habits and wills, the users can choose the block's addresses to keep their assets. These dispersedly kept assets have the following advantages: risk reduction of assets missing, maintaining social stability, privacy-preservation, *etc.* Thus, it is usual that users have many addresses where their assets are kept there. They manage the security of their assets by using the secret keys which correspond to the addresses on the blockchain. For the bitcoin, the corresponding addresses are the hash values of the public keys where the corresponding secret keys are kept secret by the users. The working process of blockchain is given in Figure 1. Supposed that *Alice* wants to send bitcoin to *Bob*. The working process is listed below: (1)*Alice* cooperates with *Bob* to create the transaction; (2) Based on the current time, the transaction is inserted into the current block; (3) When the time slot ends, the block is broadcasted to every party in the network; (4) The parties in the network approve all the transactions in the block is valid; (5) The block is added into the blockchain; (6) The bitcoin moves from *Alice* to *Bob*. In the real working environment,

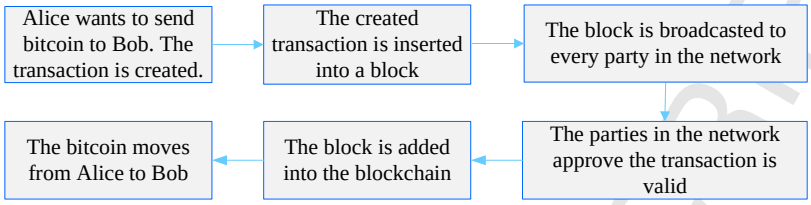

Figure 1: The working process of blockchain

when *Alice* buys something from *Bob*, *Alice* will send money (bitcoin) to *Bob*. In order to finish the transaction, the working process will be performed as the Figure 1. Of course, on the blockchain, any transaction has the similar working process as it.

Since most users have more than one address on the blockchain, their assets are dispersedly stored on many different addresses. In order to make the transaction succeed and get rid of the potential troublemakers, it is necessary to show the trading party (payer) has enough assets for the trading. Since all the transactions will be recorded on the blockchain, privacy-preservation is fundamental to blockchain development. In order to protect the trading party's assets privacy, the verifier must be designated. The other parties have no right to verify the payer's assets. At the same time, the proving process of assets must be non-transitive.

1.1. Motivation

When one entity (vendor, *e.g.*, company or individual) plans to sell an expensive goods, the feasible way is to advertise this marketing information to all the potential buyers. Based on the different purposes, many entities (*i.e.*, buyer) will contact the vendor and further negotiate the transaction details, such as unit price, total price, trading day, *etc.* When the entity has no enough bitcoin to buy the expensive goods, it is meaningless to negotiate the transaction details with them for the vendor. Although they also contact with the vendor and negotiate the transaction details, they only maybe interested in the transaction

or only want to confuse the vendor. Of course, the potential buyers are only the entities who have enough bitcoin to finish the transation. In order to avoid and remove the troublemakers, the vendor will set some basic conditions. For example, the vendor will require the potential buyer to prove it satisfies the basic condition, *e.g.*, it possesses enough bitcoins which are not less than the predefined constant. Usually, some entities possess a large number of bitcoins which far exceed the predefined constant. In order to protect its own bitcoin quantity privacy, the buyer only needs to prepare part of its assets which satisfy the vendor's basic condition. The buyer has the ability to choose them and proves that he owns these assets.

On the other hand, in order to protect the potential buyers' privacy, the potential buyers will designate the verifier. Except for the designated verifier, other entities have no ability to verify the potential buyers' proof of asset. Another security requirement is that the vendor cannot transmit the buyer's proof of assets. Otherwise, when the vendor is spiteful, the buyer's privacy can be leaked by the vendor. The core technology of bitcoin is blockchain. For the blockchain, the basic public key cryptography technique is elliptic cure public key cryptography. Thus, based on the above application requirements, it is necessary to study the designated-verifier proof of assets for bitcoin exchange using elliptic curve cryptography.

1.2. Related work

For the bitcoin, the informations of transaction and assets are recorded on the blockchain which can be regarded as the ledger [5]. The trusted third party is not needed by using the blockchain. Comparing with blockchain, CA (certificate authority) is the trusted third party in PKI (public key infrastructure). KGC (key generation center) is the trusted third party in identity-based public key cryptography. CA and KGC are the trusted third parties in the certificateless public key cryptography. Blockchain is a new form of information technology that has many important future applications, such as artificial intelligence, human enhancement, smart contract, *etc* [6]. A decentralized smart contrac-

t system does not store financial transactions in the clear on the blockchain. Thus, it is important to retain transaction privacy from the public's view [7]. CA and KGC do not exist in the information technology of blockchain.

Bitcoin works in practice, but not in theory. Bitcoin's stability relies on an unknown combination of socioeconomic factors which is hopelessly intractable to model with sufficient precision, failing to yield a convincing argument for the system's soundness. At the same time, there exist some security problems: privacy-preservation, key management, anonymity, *etc.* [8] Bitcoin exchange provides conversion services between bitcoin and other goods without the trusted third party. Buyers can 'withdraw' by performing the exchange to send the their bitcoin to a bitcoin address of the vendor when the exchange is agreed. When the customer stores his assets with an exchange, he may dispersedly store his assets on many addresses where the corresponding secret keys are known only to the customer on the blockchain. On the blockchain, the address is the hash value of the corresponding public key. For the depositor, his assets are the total value from his different bitcoin addresses. For the bitcoin exchange, the proof of liability is also fundamental. In 2015, Dagher *et al.* proposed a cryptographic proof of solvency scheme [9]. In their paper, the proposed scheme employed somewhat heavier cryptography, *e.g.*, additively homomorphic Pedersen commitments, zero-knowledge proofs, *etc.* In order to finish the transaction which is proposed in our motivation, the buyer must prove it owns sufficient bitcoin to match (or exceed) the agreed minimal quantity of the assets. Apart from the most widely used digital currency-bitcoin, there exist many different digital currencies, such as Zerocoin, Dogecoin, Litecoin, Peercoin, *etc.* Because the bitcoin transaction log is completely public, it is easy to break the users' privacy through analyzing the use's pseudonyms. In order to solve the problem, Miers *et al.* designed Zerocoin which is a cryptographic extension to Bitcoin that augments the protocol to allow for fully anonymous currency transactions [10]. Dogecoin is a decentralized, peer-to-peer digital currency that enables the users to easily send money online [11]. Litecoin is a peer-to-peer Internet currency that enables instant, near-zero cost payments to anyone in the world.

It is also an open source, global payment network that is fully decentralized. Bitcoin uses the mechanism of Proof-of-Work (PoW) where the node generating a block has to provide a proof that it can generate the pre-image of the hash value. Proof-of-stake (PoS) aims to replace the way of achieving consensus in a distributed system. Generating a block involves sending coins to oneself, which proves the ownership. The required amount of coins (also called target) is specified by the network through a difficulty adjustment process similar to PoW that ensures an approximate, constant block time. The first PoS based currency was PeerCoin [12] which is still in a period of PoW mining. Further development of the PeerCoin PoS protocol leads to NovaCoin [13] which uses a hybrid PoS/PoW system. There also exist many other digital coins.

In the public key cryptography, digital signature is concerned with the authenticity of data. For an unforgeable signatures scheme, it is infeasible to forge the signatures of other users to documents that they do not sign [14]. Along with the development of elliptic curve public key cryptography, ECDSA (elliptic curve digital signature algorithm) was proposed [15]. Based on the better properties (*e.g.*, shorter key, faster than RSA), ECDAS develops very rapidly. All kinds of special signature primitives are proposed. In 1996, the concept of designated-verifier signature (DVS) was introduced [16]. Until now, in the elliptic cure public key cryptography, most designated-verifier digital signature schemes were designed by using the bilinear pairings [17, 18, 19]. On the blockchain, the transaction takes use of ECDSA without bilinear-pairings. Based on the hardness assumption of elliptic curve computational Diffie-Hellman problem, an efficient DVS scheme using elliptic curve public key cryptography with pairing-free operation was proposed [20].

Zero-knowledge proofs are proofs that yield nothing (*i.e.*, "no knowledge") beyond the validity of the assertion. In order to preserve the buyer' secret keys privacy in the digital signature, it is inescapable to take use of the zero-knowledge proof technique. From the basic Σ-protocols such as the Schnorr proof of knowledge of a discrete logarithm [21] or the Chaum-Pedersen proof of representation of a Diffie-Hellman tuple [22], a non-interactive zero-knowledge

protocol (NIZKP) can be gained via the Fiat-Shamir transform [23]. In the digital signature and NIZKP, an indispensable component is cryptographic hash function. Assuming the existence of very strong collision-free hashing functions, one can construct a computationally sound (zero-knowledge) proof for any language in NP-hard problem, using only poly-logarithmic amounts of communication and randomness [14]. A hash function is any function that can be used to map data of arbitrary size to data of fixed size. Hash function is collision-resistant, which means that it is very hard to find data that will generate the same hash value. These functions are categorized into cryptographic hash functions and provably secure hash functions [24]. In the security proof, when the hash function is looked as the random oracle, the model is called the random oracle model; otherwise, the model is called the standard model.

1.3. Our contribution

In bitcoin exchange, it is important to prove the buyer's capacity to pay. In order to realize the function, the buyer needs to prove that his assets are not less than some agreed line of balance. In order to protect the buyer's privacy, only the designated verifier (*i.e.*, vendor) has the ability to verify the buyer's proof. In this paper, we proposed the novel concept of designated-verifier proof of assets for bitcoin transaction using elliptic curve public key cryptography. This security primitive comes from the application requirements. For the novel security primitive, we give its formal definition. Then, we formalize its system model and security model. By taking use of elliptic curve discrete logarithm problem, elliptic curve computational Diffie-Hellman problem and collision-resistance of cryptographic hash function, we design an efficient and secure DV-PoA scheme. Through analyzing its security and efficiency, our DV-PoA scheme is provably secure and efficient.

1.4. Organization

We organize the rest of our paper below. Section 2 gives some preliminaries that include the definition, system model, security model of DV-PoA scheme. It

Table 1: Notations and descriptions

Notations	Descriptions
DV-PoA	designated-verifier proof of assets
ECC	Elliptic curve public key cryptography
$\mathcal{F}_q$	The base field of the elliptic curve $E(\mathcal{F}_q)$
$\mathcal{G}$	A finite group which consists of all the points of $E(\mathcal{F}_q)$ with the point addition and scalar multiplication algorithms
P	A generator of $\mathcal{G}$
p	The order of the generator P
Alice	The buyer
Bob	The vendor
H_1, H_2	Two cryptographic hash functions
Cont	The contract which is agreed by *Alice* and *Bob*

also includes the basic knowledge of elliptic curve public key cryptography, some difficult problems and two formal symbolisms. Section 3 presents our concrete DV-PoA scheme and analyzes its correctness. In order to show the construction intuition, the construction architecture is also given in the section. Section 4 analyzes our concrete DV-PoA scheme's security and efficiency. Finally, our paper is concluded in Section 5. The future research directions are also given in this section.

Table 1 lists the notations and their descriptions which are used throughout this paper.

2. Preliminaries

First, we formalize its system model and security model for the proposed DV-PoA concept. Then, we give the formal definition of DV-PoA. At last, we give some preliminaries which will be used in the construction and security analysis of our concrete DV-PoA scheme.

2.1. System model, definition and security model

Our DV-PoA system consists of three different network entities: *Buyer, Vendor, Blockchain*. The three different network entities are described below.

1. *Buyer*: an network entity which plans to trade with the vendor. In order to ensure the vendor, the buyer will prove it possesses enough assets which satisfy the vendor's minimum asset requirement. In the paper, we denote *Alice* as the buyer.

2. *Vendor*: an entity which will sell its goods to the potential buyers. In order to get rid of the troublemakers, it will require the potential buyers to provide the proof of assets which satisfy the minimum asset requirement. In the paper, we denote *Bob* as the vendor.

3. *Blockchain*: an entity which is a digital ledger that records every bitcoin transaction that has ever occurred. A blockchain implementation consists of two kinds of records: transactions and blocks [25].

In order to design the concrete DV-PoA scheme and give the formal security proof, the formal definition and security model is necessary. First, we formalize the definition of our DV-PoA scheme. Then, we formalize DV-PoA's security model. They are given below:

Definition 1 (DV-PoA)**.** *A formal DV-PoA scheme comprises five phases: SetUp, Agreement, Sign, Verify-Decision, Sim. For the above five phases, we give their detailed characterizations below:*

1. *SetUp: On inputting the security parameter k, the system parameters are generated. Besides of the system parameters, Alice's multi secret key/public key pairs, Alice's storage addresses on the blockchain are also generated. At the same time, Bob's secret key/public key pair, and the corresponding storage address are also generated.*

2. *Agreement: Alice and Bob interact each other to decide on all the transactions details which include the buyer, vendor, unit price, total price,*

minimal asset requirement, object, etc. Let the agreed transaction details be contained in the file Cont.

3. *Sign: Alice picks part of its own addresses on the blockchain where the corresponding secret keys are known to her. By taking use of these secret keys, Alice signs the agreement Cont. The constraint is that the total assets satisfy the agreed file Cont.*

4. *Verify-Decision: On inputting the signature from Alice, Bob verifies its validity. If it fails, Bob refuses Alice; otherwise, Bob accepts Alice. Based on the condition that Alice's signature is valid, Bob decides whether Alice's total assets satisfy Cont. If the total assets satisfy the agreed file Cont, Bob will perform the transaction with Alice; otherwise, the transaction terminates.*

5. *Sim: Bob can also generate the signature which is indistinguishable from Alice's signature.*

For a practical DV-PoA scheme, efficiency and security are two important cases. A secure DV-PoA scheme must satisfy the security requirements below:

1. If some addresses' secret keys are unknown to *Alice*, it is infeasible to pass herself off as the addresses' owners to sign the agreement *Cont*.

2. The other verifiers have no ability to perform the phase *Verify-Decision* except for the designated-verifier which is determined by *Alice*.

3. The designated-verifier *Bob* cannot transfer *Alice*'s proof to any other party. Non-transferability must be satisfied.

Based on the above three security requirements, we give the corresponding formal security models below:

Definition 2 (Unforgeability)**.** *A DV-PoA scheme satisfies the unforgeability if for any PPT (probabilistic polynomial time) adversary $\mathcal{A}$ (i.e., malicious buyer Alice), the probability that $\mathcal{A}$ wins the following game is negligible. The following*

game is interacted between the challenger $\mathcal{C}$ and the adversary $\mathcal{A}$. The detailed interaction is shown below:

1. *SetUp: $\mathcal{C}$ creates the system parameters. Let Alice's assets address set be $\mathcal{S}$. Then, it also creates the corresponding secret keys s_i and public keys Pub_i where the corresponding address $Addr_i$ belongs to $\mathcal{S}$, i.e., $Addr_i \in \mathcal{S}$. For the bitcoin, the address is the hash value of the corresponding public key on the blockchain. $\mathcal{C}$ sends the secret keys $\{s_i, Addr_i \in \mathcal{S}\}$ to $\mathcal{A}$. $\mathcal{C}$ also creates the secret key s_B and public key Pub_B pair for Bob. Bob's receiving address is the hash value of Pub_B. The system parameters and public keys are made public.*

2. *Query: In the simulated interaction, $\mathcal{A}$ queries different oracles to $\mathcal{C}$ adaptively. $\mathcal{C}$ answers these queries on behalf of the different oracles. Although the interaction is not the real-world scene, $\mathcal{A}$ has not the computational ability to identify it. Thus, from $\mathcal{A}$'s view, the simulated interaction is computationally indistinguishable from the real-world interaction.*

   - *Hash-oracle. $\mathcal{A}$ submits the hash queries to $\mathcal{C}$. Based on the hash queries, $\mathcal{C}$ answers $\mathcal{A}$ with the corresponding hash values.*
   - *Sign-oracle. $\mathcal{A}$ submits the message m to $\mathcal{C}$ and queries the message's signature by using the secret keys $s_i, Addr_i \in \mathcal{S}_m$ where $\mathcal{S}_m \not\subseteq \mathcal{S}$. Based on the query, $\mathcal{C}$ computes the message m's signature and sends the signature to $\mathcal{A}$. In the phase Sign-oracle, denote the queried address-message pair set as $(\bar{\mathcal{S}}, \mathcal{M}) = \{(\mathcal{S}_m, m)\}$.*

3. *Forge: $\mathcal{A}$ forges a signature σ for the address-message pair $(\mathcal{S}_{\bar{m}}, \bar{m})$ where $(\mathcal{S}_{\bar{m}}, \bar{m}) \notin (\bar{\mathcal{S}}, \mathcal{M})$ and $\mathcal{S}_{\bar{m}} \not\subseteq \mathcal{S}$.*

$\mathcal{C}$ can undertake the task of the designated-verifier Bob since Bob's secret key is created by $\mathcal{C}$. Based on the above game, we say that $\mathcal{A}$ wins if the forged signature σ can pass $\mathcal{C}$'s verification with non-negligible probability. It can be

expressed by the following formula:

$$\Pr\left[V_{\mathcal{C}}(\mathcal{S}_{\bar{m}}, \bar{m}, \sigma) = 1 \;\middle|\; \begin{array}{c} (\mathcal{S}_{\bar{m}}, \bar{m}) \leftarrow \mathcal{A}^{O_{Hash}, O_{Sign}}(s_i, Addr_i \in \mathcal{S}) \\ (\mathcal{S}_{\bar{m}}, \bar{m}) \notin (\bar{\mathcal{S}}, \mathcal{M}),\ \mathcal{S}_{\bar{m}} \not\subseteq \mathcal{S},\ \mathcal{C} \leftarrow s_B \end{array} \right] < \frac{1}{p(k)}$$

where k is the security parameter and $p(k)$ is the polynomial of k.

The definition 2 gives the unforgeability model for the adversary *Alice*. The following definition 3 will give the security property of designated-verifier.

Definition 3 (Designated-verifier)**.** *The DV-PoA scheme is a designated-verifier signature scheme if the other entities cannot perform the phase Verify-Decision except for the designated-verifier.*

In order to preserve *Alice*'s asset quantity privacy, it is important to ensure that the designated-verifier cannot transfer *Alice*'s proof. In order to realize the security target, we define the non-transferability in the definition 4.

Definition 4 (Non-transferability)**.** *A DV-PoA scheme satisfies the non- transferability if for any PPT distinguisher $\mathcal{D}$, the probability that $\mathcal{D}$ wins the following game is negligible. The following game is the interaction between the challenger $\mathcal{C}$ and the distinguisher $\mathcal{D}$. The detailed interaction is shown below:*

1. *Setup: On inputting the security parameter k, $\mathcal{C}$ creates the system parameters. At the same time, it also creates Alice's secret key/public key pairs and Bob's secret key/public key pair. Alice and Bob's public keys are made public and the secret keys are made secret to the distinguisher $\mathcal{D}$.*

2. *$\mathcal{D}$ submits the hash queries and signature queries to $\mathcal{C}$. According to $\mathcal{D}$'s queries, $\mathcal{C}$ responds $\mathcal{D}$ with the corresponding hash values and signatures.*

3. *$\mathcal{D}$ submits a new message m^* to $\mathcal{C}$. $\mathcal{C}$ flips a fair coin $b \leftarrow \{0, 1\}$. If $b = 0$, $\mathcal{C}$ runs the algorithm Sign and returns $\mathcal{D}$ with the signature σ^* by using Alice's secret keys; otherwise, $\mathcal{C}$ runs the algorithm Sign and returns $\mathcal{D}$ with the signature σ^* by using Bob's secret key.*

4. *$\mathcal{D}$ submits new hash and signature queries to $\mathcal{C}$. $\mathcal{C}$ responds $\mathcal{D}$ with the corresponding hash values and signatures.*

5. $\mathcal{D}$ outputs a bit b' and wins if $b' = b$.

Based on the above game, we define $\mathcal{D}$'s advantage as

$$Adv_{\mathcal{D}}^{O_{Hash},O_{Sign}} = |\Pr[b' = b] - \frac{1}{2}|$$

If $Adv_{\mathcal{D}}$ is negligible, we say that the DV-PoA scheme satisfies the security property of non-transferability.

2.2. Some preliminaries

In order to construct the concrete DV-PoA scheme, we will take use of ECC (Elliptic curve public key cryptography) and cryptographic hash function. At the same time, we give some difficult problems on ECC. On the other hand, we also give two formal symbolisms, *i.e.*, (ϵ, τ, q)-forger for digital signature, (ϵ, τ)-collision-finder for hash function. They are given below.

2.2.1. ECC

In the elliptic curve public key cryptography, the cryptographic operations are performed on the elliptic curve group on the finite field. When the elliptic curve E is defined on the finite field $\mathcal{F}_q$ where q is a prime number, it is denoted as $E(\mathcal{F}_q)$. For the two elements $a, b \in \mathcal{F}_q^*$, $E(\mathcal{F}_q)$ can be given by the following equation: $E : y^2 = x^3 + ax + b \bmod q$. On the elliptic curve $E(\mathcal{F}_q)$, all the elements construct a finite group which is denoted as the symbol $\mathcal{G}$. We pick a generator P of $\mathcal{G}$. Let the order of P be the prime p. On $E(\mathcal{F}_q)$, we give the DLP (Discrete logarithm problem) and CDHP (Computational Diffie-Hellman problem) below:

Definition 5 (DLP)**.** *Given the pair $(P, P_1) \in \mathcal{G}^2$, DLP is to calculate $x \in \mathcal{F}_p^*$ such that $P_1 = xP$.*

Definition 6 (CDHP)**.** *Given the triple $(P, P_1, P_2) \in \mathcal{G}^3$, CDHP is to calculate xyP where $P_1 = xP$, $P_2 = yP$ and x, y are unknown.*

In this paper, we chooses the appropriate elliptic curve group $\mathcal{G}$ where *DLP, CDHP* are assumed to be computationally difficult [26][27].

2.2.2. Cryptographic hash function

A cryptographic hash function can map data of arbitrary size to a bit string of a fixed size. We denote it as $H : X \to Y$. A valid cryptographic hash function H satisfies the following four security properties:

- It is efficient to calculate the function value of H.
- It is computationally infeasible to get the pre-image from its hash value.
- A small change to a message should change the hash value so extensively that the new hash value appears uncorrelated with the old hash value.
- It is collision-resistant, *i.e.*, it is computationally difficult to find a collision pair (x_1, x_2) which satisfy $H(x_1) = H(x_2)$.

Cryptographic hash functions can be used in many information-security environments, such as digital signatures, message authentication codes (MACs), and other forms of authentication. We can also use them as ordinary hash functions, to index data in hash tables, for fingerprinting, to detect duplicate data or uniquely identify files, and as checksums to detect accidental data corruption.

2.2.3. Two formal symbolisms

For ECDSA and hash function, we give the corresponding formal symbolisms, *i.e.*, (ϵ, τ, q)-forger for digital signature, (ϵ, τ)-collision-finder for hash function.

Definition 7 ((ϵ, τ, q)-security)**.** *For a digital signature scheme, if an adversary runs in time at most τ, succeeds with probability at least ϵ, and makes at most q queries to the signing oracle, then we shall call the adversary a (ϵ, τ, q)-forger. If $q = 0$, we call the adversary a passive forger, and call the attack a non-message attack. A signature scheme satisfies the (ϵ, τ, q)-security if there does not exist an (ϵ, τ, q)-forger of the signature scheme.*

For a cryptographic hash function, it must satisfy the security property of (ϵ, τ)-collision-resistance.

Definition 8 ((ϵ, τ)-collision-resistance)**.** *An (ϵ, τ)-collision-finder of H is an algorithm C_H with running time at most τ and which with probability at least ϵ generates two distinct messages $x_1, x_2 \in \{0,1\}^*$ such that $H(x_1) = H(x_2)$. A cryptographic hash function satisfies the (ϵ, τ)-collision-resistance if such an (ϵ, τ)-collision-finder does not exist.*

3. Our proposed DV-PoA Scheme

First, we construct the concrete DV-PoA scheme based on the elliptic curve public key cryptography. Then, we analyze the constructed scheme's correctness.

3.1. Construction of our concrete DV-PoA scheme

Our concrete DV-PoA scheme consists of five phases: *SetUp, Agreement, Sign, Verify-Decision*, and *Sim*. In order to show the proposed DV-PoA scheme's construction intuition, we depict it in the Figure 2. In the concrete construction, there exist three different entities: the ledge entity *BlockChain*, the prover *Alice* and the verifier *Bob*. We assume that the system parameters have been generated. Based on the Fig. 2, our concrete DV-PoA scheme is outlined below:

1. Through the interaction between *Alice* and *Bob*, the agreement *Cont* is created. *Cont* contains some information, such as the prover, the verifier, object, unit price, total price, *etc.*

2. Through accessing and looking up the blockchain, *Alice* gets its own addresses on the blockchain and the corresponding assets where *Alice* owns these addresses' secret keys. From its whole addresses and assets, *Alice* determines the address set $\mathcal{S}$ where the corresponding assets will be shown to *Bob*.

3. *Alice* proves the proprietary rights to the addresses which belong to $\mathcal{S}$. Then, *Alice* sends the proof to *Bob*.

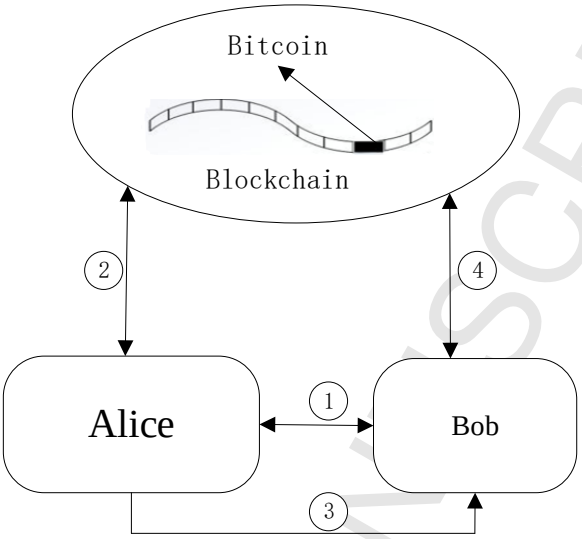

Figure 2: Construction architecture of our concrete scheme

4. *Bob* verifies the proof which comes from *Alice*. When it holds, *Bob* further confirms whether the total proved assets satisfy the agreement *Cont*.

In the following, we give the five different phases in detail:

SetUp: Pick the secure elliptic curve E on the finite field $\mathcal{F}_q$ where q is a prime number. Thus, all the points on E construct a finite group which is denoted as $\mathcal{G}$ which has the prime order p. Let P be a generator of $\mathcal{G}$. Two different hash functions are given: $H_1 : \{0,1\}^* \to \mathcal{F}_p$, $H_2 : \mathcal{G} \times \{0,1\}^* \to \mathcal{F}_p^*$. *Alice* picks the address set $\mathcal{S}$ whose corresponding secret keys are known to *Alice*. Let *Bob*'s public key be Pub_B and the corresponding secret key be s_B. The secret key/public key pair (s_B, Pub_B) satisfies $Pub_B = s_B P$. For the address $Addr_i \in \mathcal{S}$, we denote its secret key/public key pair as (s_i, Pub_i) which also satisfies $Pub_i = s_i P$.

Agreement: Suppose that *Alice* plans to trade with *Bob*. They interact each other to decide on some transaction details, *e.g.*, prover, verifier, object, unit price, total price, trading day, constraint condition, penalty, *etc.* These transaction details will be contained in a file which is denoted as *Cont*.

Sign: In order to generate the proof of assets, *Alice* performs the following procedures:

1. *Alice* picks a random number $k \in \mathcal{F}_p^*$ and calculates the corresponding point $R = kP = (x_R, y_R)$. Taking use of the x-coordinate, *Alice* calculates the number $r = x_R \bmod p$. If $r = 0$, *Alice* picks another number k from $\mathcal{F}_p^*$ and calculates R, r again until $r \neq 0$; otherwise, *Alice* performs the next procedure.

2. *Alice* calculates the hash value $e = H_1(Cont)$.

3. Taking use of *Bob*'s public key Pub_B and its own secret keys s_i where $Addr_i \in \mathcal{S}$, *Alice* calculates the point $K_{AB} = (\sum_{Addr_i \in \mathcal{S}} s_i) Pub_B$ and the hash value $s_{AB} = H_2(K_{AB}, Cont)$.

4. Taking use of s_{AB}, *Alice* calculates $w = k^{-1}(e + s_{AB} r) \bmod p$. If $w = 0$, goto 1 until $w \neq 0$ in this phase; otherwise, *Alice* sends $(Cont, r, w)$ to *Bob*.

Verify-Decision: Upon receiving the proof $(Cont, r, w)$ from *Alice*, *Bob* performs the following procedures to verify it and decide whether *Alice*'s total assets satisfy the agreement *Cont*.

1. *Bob* calculates the hash value $e = H_1(Cont)$ and the point $K_{AB} = s_B(\sum_{Addr_i \in \mathcal{S}} Pub_i)$. Then, *Bob* gets the secret value $s_{AB} = H_2(K_{AB}, Cont)$.

2. *Bob* calculates the three different numbers:

$$\bar{w} = w^{-1} \bmod p,\ u_1 = \bar{w} e \bmod p,\ u_2 = \bar{w} r \bmod p$$

3. *Bob* calculates the point $R' = (u_1 + u_2 s_{AB})P = (x_{R'}, y_{R'})$. By using the x-coordinate $x_{R'}$, *Bob* calculates $r' = x_{R'} \bmod p$.

4. If $r \neq r' \bmod p$, *Bob* rejects *Alice*'s proof $(Cont, r, w)$ and informs *Alice*; otherwise, *Bob* performs the procedures blow:

(a) From the agreement *Cont*, *Bob* gets the addresses $Addr_i, i \in \mathcal{S}$ of the blockchain. *Bob* interacts with the blockchain and gets the balance bal_i for the address $Addr_i \in \mathcal{S}$.

(b) *Bob* calculates *Alice*'s total asset $Asset = \sum_{Addr_i \in \mathcal{S}} bal_i$ and decides whether $Asset$ satisfies the agreement $Cont$. If it satisfies $Cont$, the trade between *Alice* and *Bob* continues; otherwise, *Bob* refuses to trade with *Alice*.

Sim:*Bob* has also the ability to produce the simulated proof of assets below:

1. *Bob* picks a random number $k \in \mathcal{F}_p^*$ and calculates the corresponding point $R = kP = (x_R, y_R)$. Taking use of the x-coordinate, *Bob* calculates the number $r = x_R \bmod p$. If $r = 0$, *Bob* picks another number k from $\mathcal{F}_p^*$ and calculates R, r again until $r \neq 0$; otherwise, *Bob* performs the next procedure.

2. *Bob* calculates $e = H_1(Cont)$, $K_{AB} = s_B(\sum_{Addr_i \in \mathcal{S}} Pub_i)$, and $s_{AB} = H_2(K_{AB}, Cont)$.

3. Taking use of s_{AB}, *Bob* calculates $w = k^{-1}(e + s_{AB}r) \bmod p$. If $w = 0$, goto 1 until $w \neq 0$ in this phase; otherwise, *Bob* gets the simulated proof $(Cont, r, w)$.

3.2. Correctness

Our proposed concrete DV-PoA scheme satisfies the correctness. It means that, if *Alice* honestly performs our proposed scheme, $(Cont, r, w)$ can pass *Bob*'s

verification. We deduce our DV-PoA scheme's correctness below:

$$\begin{aligned}
R' =&(u_1 + u_2 s_{AB})P \\
=&(\bar{w}e + \bar{w}rs_{AB})P \\
=&w^{-1}(e + rH_2(s_B \sum_{Addr_i \in \mathcal{S}} Pub_i, Cont))P \\
=&w^{-1}(e + rH_2(\sum_{Addr_i \in \mathcal{S}} s_i Pub_B, Cont))P \\
=&w^{-1}(e + rH_2(K_{AB}, Cont))P \\
=&w^{-1}(e + rs_{AB})P \\
=&kP \\
=&R
\end{aligned}$$

Thus, the formula $r' = x_{R'} = x_R = r \bmod p$ holds.

4. Performance analysis

We analyze the proposed DV-PoA scheme's performance which includes the security analysis and efficiency analysis. The security is based on the computational difficulty of some mathematical problems, such as DLP, DHP on the elliptic curve E and collision-resistance of cryptography hash function. In the generic group model, our proposed PV-PoA scheme is provably secure. On the other hand, we analyze our DV-PoA's efficiency from the storage cost and computation cost. At last, we implement our scheme to show our scheme's computation cost.

4.1. Security analysis

For a secure DV-PoA scheme, it must satisfy unforgeability. In other words, *Alice* cannot forge the signature on behalf of the addresses whose corresponding secret keys are unknown to her. By changing our point of view, if *Alice* cannot personate other entities to prove their assets, other entities cannot also personate *Alice* to prove her assets. Thus, unforgeability is the fundamental security

requirement for the DV-PoA scheme. In order to prove this security property, we review the security result of ECDSA.

Theorem 1. *[28] If there exists an (ϵ, τ, q_s) forger $\mathcal{A}$ of ECDSA with hash function H_1 in the generic group model for $\mathcal{F}_p$, then there exists an (ϵ', τ')-collision-finder C_{H_1} where*

$$\epsilon' \geq \epsilon - 3\binom{\tau'}{2}/p,\ \tau' \leq 2(\log p)(\tau + q_s)$$

provided that $\epsilon' \geq (p - \binom{\tau}{2})^{-1}$ and q_s denotes the query times to the signature oracle.

The above theorem 1 can be used to prove the following theorem:

Theorem 2 (Unforgeability)**.** *Based on the collision-resistance assumption of the hash function H_1, our proposed DV-PoA scheme is unforgeable. Speak specifically, if an adversary $\mathcal{A}$ can break our DV-PoA scheme with the non-negligible probability within some time period, then we can find the collision of H_1 with another non-negligible probability within some time period.*

Proof. In our DV-PoA scheme, we assume that *Alice* knows the secret keys whose corresponding public keys are mapped into the address set $\mathcal{S}$ by using the hash function, such as SHA-256. We denote $\bar{s} = (\sum_{Addr_i \in \mathcal{S}} s_i)s_B$ where $\bar{s}$ is unknown to *Alice*. We also denote $s_{AB} = H_2(\bar{s}P, Cont)$ as the secret key and $s_{AB}P$ as the public key. Since $\bar{s}P$ can be calculated by *Alice* and *Bob*, the secret value s_{AB} can also be shared between *Alice* and *Bob*. Based on the difficulty assumption of CDHP on the elliptic curve E, the secret value s_{AB} is unknown to the other parties except for *Alice* and *Bob*.

When we consider s_{AB} as the secret key for the signature, our proposed DV-PoA scheme is the standard ECDSA where the signature secret key/verification public key pair is $(s_{AB}, s_{AB}P)$. Based on the theorem 1, our proposed DV-PoA scheme satisfies the unforgeability. Further more, the unforgeability of the DV-PoA scheme comes from the collision-resistance property of H_1. □

Theorem 3 (Designated-verifier)**.** *Based on the difficulty assumption of DLP and CDHP on the elliptic curve E, only the designated-verifier has the ability to verify our DV-PoA scheme.*

Proof. In order to verify our scheme, the verifier must perform the verification formula

$$R' = (u_1 + u_2 H_2(k_{AB}, Cont))P = (x_{R'}, y_{R'})$$

where $k_{AB} = (\sum_{Addr_i \in \mathcal{S}} s_i) Pub_B = (s_B \sum_{Addr_i \in \mathcal{S}} s_i)P$. By using x-coordinate $x_{R'}$, the verifier verifies whether $r = x_{R'} \bmod p$ holds. Then, by using the agreement *Cont* and accessing the blockchain, the verifier decides whether or not to accept the signature and perform the transaction.

Based the difficulty assumption of DLP and CDHP on the elliptic curve E, it is difficult to get k_{AB} from $(P, Pub_B, Pub_i, i \in \mathcal{S})$. Thus, except for *Alice* and *Bob*, other entities cannot calculate k_{AB} and s_{AB}. So, the verification formula cannot be performed by the other verifiers except for the designated-verifier *Bob*. □

Theorem 4 (Non-transferability)**.** *Our proposed DV-PoA scheme satisfies the security property of non-transferability.*

Proof. We give the signature simulation from *Bob* below. *Bob* picks a random $k \in \mathcal{F}_p^*$ and calculates $R = kP = (x_R, y_R)$, $r = x_R \bmod p$. If $r = 0$, *Bob* picks another k and calculates R, r again until $r \neq 0$; otherwise, *Bob* calculates $e = H_1(Cont)$, $K_{AB} = s_B(\sum_{Addr_i \in \mathcal{S}} Pub_i)$, $s_{AB} = H_2(K_{AB}, Cont)$. Then, *Bob* calculates $w = k^{-1}(e + s_{AB} r) \bmod p$. The simulated signature is $(Cont, r, w)$. It is also the same as the phase *Sim* of the DV-PoA scheme.

It is easy to see that the simulated signature is computationally undistinguishable from the signature from the true signer *Alice*. Thus, our DV-PoA scheme satisfies the security property of non-transferability. □

Discussion: In this paper, based on the special digital currency-bitcoin, we proposed the concrete DV-PoA scheme. Our scheme takes use of the secret key of the buyer *Alice* and the public key of the vendor *Bob*. We can consider

the other digital currencies, such as Zerocoin, Dogecoin, Litecoin, Peercoin. For these digital currencies, the transaction mechanism is similar. Thus, our scheme can also be used by the other digital currencies.

4.2. Storage cost and computation cost

4.2.1. Storage cost

The bit transaction will be recorded on the blockchain. In order to reduce the storage size, it is important to analyze the storage cost. The storage cost mainly consists of the agreement *Cont* and the signature on *Cont*. Based on the breakthrough time, 160 bits secret key length in elliptic curve cryptology has the same security level as the 1024 bits secret key length in RSA cryptology. Usually, the elliptic curve is operated on the finite field $\mathcal{F}_q$ where the length of q is 512 bits. In the elliptic curve cryptology, the point size is $2 * |q| = 1024$ bits, *i.e.*, $|q| = 512$ bits. For the agreement *Cont*, the signature (r, w) length is $|r| + |w| = 2 * |p| = 2 * 160 = 320$ bits. Thus, *Alice* will sends $|Cont| + |r| + |w| = 320 + |Cont|$ bits to *Bob*. *Bob* stores $(Cont, r, w)$ on the blockchain.

4.2.2. Computation cost

In the designed DV-PoA scheme, the computation cost mainly comes from two phases: *Sign* and *Verify-Decision*. In the phase *Sign*, *Alice* performs 2 elliptic curve scalar multiplication on the group $\mathcal{G}$ which is operated on the finite field $\mathcal{F}_q$. In addition, *Alice* will perform 3 operations modulo p, 1 operation on the hash function H_1 and 1 operation on the hash function H_2. In the phase *Verify-Decision*, *Bob* performs 3 elliptic curve scalar multiplication and $|\mathcal{S}|$ point addition on the group $\mathcal{G}$. On the other hand, *Bob* performs 4 operations modulo p, 1 operation on the hash function H_1 and 1 operation on the hash function H_2.

In order to show our scheme's practical computation cost and the corresponding cost curve, we implement the proposed scheme. In our implementation, we take use of C programming language and Miracl library [29]. Both *Alice* and *Bob* work on DELL PowerEdge R420 Server with the following settings:

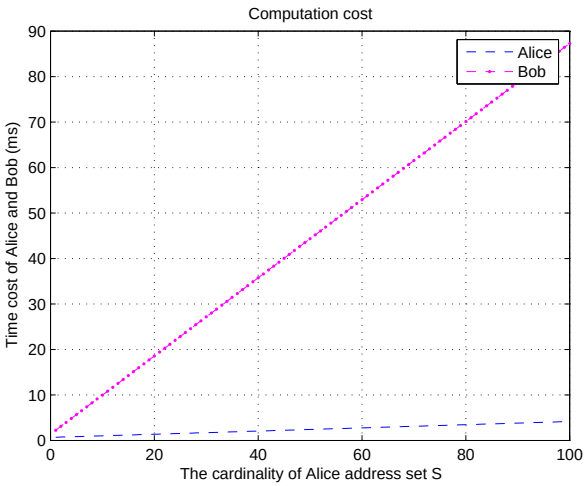

Figure 3: Time cost of *Alice* and *Bob*

- CPU: Intel R Xeon R processor E5-2400 and E5-2400 v2 product families
- Physical Memory: 8GB DDR3 1600MHz
- OS: Ubuntu 13.04 Linux 3.8.0-19-generic SMP i686

For a message, when $|S| = 1$, *Alice* will cost 0.679 milliseconds to finish one time signature by using the phase *Sign*. When $|S| = 1$, *Bob* will cost 2.327 milliseconds to finish the phase *Verify-Decision*. Figure 3 shows the computation time cost (millisecond) for *Alice* and *Bob*. The x-coordinate denotes the cardinality of *Alice*'s address set $\mathcal{S}$. The y-coordinate denotes the time cost (millisecond) of *Alice* and *Bob* in performing the phase *Sign* and *Verify-Decision*. Through the theoretical analysis and the practical implementation, our DV-PoA scheme is efficient and practical.

5. Future research directions and conclusion

In the research field of DV-PoA, there exist some difficult problems which have not been solved. In the future, we will study the following problems. When

the buyer consists of multi entities or the vendor consists of multi entities, the secure multi-party computation is necessary on the blockchain. How to design the corresponding multi-party computation which can be used to construct the multi-party DV-PoA scheme? In the public blockchain, the address is the hash value of the public key. Ring signature can be used to preserve the public key privacy. When the address is public and the public key is concealed in the hash function, how to use ring signature to preserve the public key privacy in DV-PoA?

In this paper, we propose the novel concept of designated-verifier proof of assets for bitcoin exchanges using elliptic curve cryptography. For the novel security primitive, we give its formal definition, system model and security model. Then, we construct a concrete DV-PoA scheme which comes from elliptic curve cryptography. Through analyzing its security and performance, our constructed DV-PoA scheme is provably secure and efficient.

Acknowledgments

The work of H. Wang was sponsored by Qing Lan Project of Jiangsu province and the Open Foundation of State Key Laboratory of Information Security of China (No. 2017-MS-15). The work of D. He was sponsored by the National Natural Science Foundation of China (No. 61501333, 61572379), the Natural Science Foundation of Hubei Province of China (No. 2015CFB257) and Fundamental Research Funds for the Central Universities. The work of Y. Ji was sponsored by the Open-End Fund of Key Laboratory of Intelligent Perception and Systems for High-Dimensional Information of Ministry of Education at Nanjing University of Science and Technology (TK216010).

6. *Reference

[1] M. Belenkiy. “E-Cash”, Handbook of Financial Cryptography and Security, CRC, 2011.

[2] J. Camenisch, S. Hohenberger, and A. Lysyanskaya. "Compact e-cash". EUROCRYPT 2005, 2005: 302-321.

[3] D. Chaum. "Blind signatures for untraceable payments", CRYPTO 1982, 1982: 199-203.

[4] R. Parhonyi. "Micropayment Systems", Handbook of Financial Cryptography and Security, CRC, 2011.

[5] Nakamoto S. "Bitcoin: A peer-to-peer electronic cash system", 2008. http://www.cryptovest.co.uk/resources/Bitcoin

[6] Swan M. "Blockchain thinking: The brain as a DAC (decentralized autonomous organization)", Texas Bitcoin Conference. 2015: 27-29.

[7] Kosba A, Miller A, Shi E, Wen Z, Papamanthou C. "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts", 2016 IEEE Symposium on Security and Privacy, 2016: 839-858.

[8] J. Bonneau, A. Miller, J. Clark, A. Narayanan, J. A. Kroll, and E. W. Felten. "Research Perspectives and Challenges for Bitcoin and Cryptocurrencies", IEEE Symposium on Security and Privacy, 2015.

[9] Dagher G G, Bünz B, Bonneau J, Clark J, Boneh D. "Provisions: Privacy-preserving proofs of solvency for Bitcoin exchanges", Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, 2015: 720-731.

[10] Miers I., Garman C., Green M., Rubin A. D., Zerocoin: Anonymous distributed e-cash from bitcoin. 2013 IEEE Symposium on Security and Privacy, 2013, pp. 397-411.

[11] W. Markus, Dogecoin, http://dogecoin.com/, 2013.

[12] Sunny King and Scott Nadal. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. peercoin.net, 2013.

[13] NovaCoin. http://coinwiki.info/en/novacoin.

[14] Golderich, Oded. "Foundations of Cryptography. Basic Applications", Cambridge University Press New York, NY, USA, 2004.

[15] Johnson D, Menezes A, Vanstone S. "The elliptic curve digital signature algorithm (ECDSA)", International Journal of Information Security, 1(1), 2001: 36-63.

[16] Markus Jakobsson, Kazue Sako, and Russell Impagliazzo. "Designated Verifier Proofs and Their Applications", EUROCRYPT'96, LNCS 1070, 1996: 143C154. Springer-Verlag.

[17] Islam, SK Hafizul, and G. P. Biswas. "A provably secure identity-based strong designated verifier proxy signature scheme from bilinear pairings." Journal of King Saud University-Computer and Information Sciences, 26(1), 2014: 55-67.

[18] Hou S, Huang X, Liu JK, Li J, Xu L. "Universal designated verifier transitive signatures for graph-based big data", Information Sciences, 318, 2015: 144-56.

[19] Huang Q, Wong DS, Susilo W. "P2ofe: privacy-preserving optimistic fair exchange of digital signatures", CT-RSA 2014, 2014: 367-384.

[20] Islam, Sk Hafizul, and G. P. Biswas. "An efficient and secure strong designated verifier signature scheme without bilinear pairings", Journal of applied mathematics and informatics 31(3_4), 2013: 425-441.

[21] C. P. Schnorr. "Efficient signature generation by smart cards", Journal of Cryptography, 4(3), 1991: 161-174.

[22] D. Chaum and T. P. Pedersen. "Wallet databases with observers", CRYPTO'92, 1992: 89-105.

[23] A. Fiat and A. Shamir. "Witness indistinguishable and witness hiding protocols", ACM STOC 1990, 1990: 416-426.

[24] Hash function, *https* : *//en.wikipedia.org/wiki/Hash_function*

[25] Economist Staff, "Blockchains: The great chain of being sure about things", The Economist.

[26] L. C. Washington, "Elliptic Curves: Number Theory and Cryptography", Chapman & Hall/CRC, 2008.

[27] Hung-Zih Liao, Yuan-Yuan Shen, "On the Elliptic Curve Digital Signature Algorithm", Tunghai Science, 8, 2016: 109-126.

[28] Brown, Daniel RL. "The exact security of ECDSA", In Advances in Elliptic Curve Cryptography. 2000.

[29] S. S. Ltd., "Miracl library." http://www.shamus.ie/index.php?page=home, 2011.

Huaqun Wang,

HuaqunWang received the BS degree in mathematics education from the Shandong Normal University and the MS degree in applied mathematics from the East China Normal University, both in China, in 1997 and 2000, respectively. He received the Ph.D. degree in Cryptography from Nanjing University of Posts and Telecommunications in 2006. Now, he is a professor of Nanjing University of Posts and Telecommunications. His research interests include applied cryptography, network security, and cloud computing security.

Debiao He,

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University in 2009. He is currently an Associate Professor of the State Key Lab of Software Engineering, Computer School, Wuhan University, Wuhan, China. His main research interests include cryptography and information security, in particular, cryptographic protocols.

Huaqun Wang

He Debiao

HIGHLIGHTS

1. We propose the concept and model of designated-verifier proof of assets for bitcoin exchange.
2. We construct the first concrete scheme to realize the designated-verifier proof of assets for bitcoin exchange by using elliptic curve cryptography.
3. We show that our proposed concrete scheme is efficient and provably secure.
4. Detailed performance analysis and experimental result are given.