Information Sciences 478 (2019) 449–460

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Machine learning based privacy-preserving fair data trading in big data market

Yanqi Zhao[a,b], Yong Yu[a,b,*], Yannan Li[a], Gang Han[c,**], Xiaojiang Du[d]

[a] School of Computer Science, Shaanxi Normal University, Xi'an 710062, China
[b] State Key Laboratory of Cryptology, P.O. Box 5159, Beijing 100878, China
[c] School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710129, China
[d] Dept. of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA

ARTICLE INFO

Article history:
Received 15 July 2018
Revised 10 October 2018
Accepted 17 November 2018
Available online 22 November 2018

Keywords:
Data trading
Privacy-preserving
Machine learning
Fairness

ABSTRACT

In the era of big data, the produced and collected data explode due to the emerging technologies and applications that pervade everywhere in our daily lives, including internet of things applications such as smart home, smart city, smart grid, e-commerce applications and social network. Big data market can carry out efficient data trading, which provides a way to share data and further enhances the utility of data. However, to realize effective data trading in big data market, several challenges need to be resolved. The first one is to verify the data availability for a data consumer. The second is privacy of a data provider who is unwilling to reveal his real identity to the data consumer. The third is the payment fairness between a data provider and a data consumer with atomic exchange. In this paper, we address these challenges by proposing a new blockchain-based fair data trading protocol in big data market. The proposed protocol integrates ring signature, double-authentication-preventing signature and similarity learning to guarantee the availability of trading data, privacy of data providers and fairness between data providers and data consumers. We show the proposed protocol achieves the desirable security properties that a secure data trading protocol should have. The implementation results with Solidity smart contract demonstrate the validity of the proposed blockchain-based fair data trading protocol.

1. Introduction

Currently, the emerging technologies and applications such as internet of things pervade everywhere in real life. As a result, massive data have been generated, produced and collected [8]. For instance, Facebook, Alibaba, Amazon and Tencent collect massive data from their user platforms [31]. These companies can take advantage of some advanced techniques such as machine learning and data mining to conduct big data analytics, making market decisions and improving their services. Big data have become invaluable asset and raised an emerging business pattern of data trading in data market such as DataExchange and Datacoup [15,20]. Data owners can obtain financial reward by sharing their data. Fig. 1 illustrates the process of data trading in a big data market. A data provider encapsulates the collected data and submits the dataset to

* Corresponding author at: SNNU, Xi'an 710119, China.
** Corresponding author.
E-mail addresses: yuyong@snnu.edu.cn, 18300794@qq.com (Y. Yu), hangang2016@mail.nwpu.edu.cn (G. Han).

https://doi.org/10.1016/j.ins.2018.11.028
0020-0255/

Fig. 1. The process of data trading in a big data market.

the big data market. Then, the data consumer submits his demands and the market matches the demand with the dataset. Next, the data consumer interacts with a data provider to conduct the transaction. To realize effective data trading in big data market, several challenges need to be resolved, including how to ensure the availability of trading data, how to protect the privacy of identity, how to ensure the fairness. In data trading, it is hard for data consumer to verify the availability of the encrypted data. In consideration of the privacy issue, the provider is usually reluctant to reveal his real identity to the data consumer. Furthermore, the atomic exchange and the payment fairness between the data provider and the data consumer are hard to guarantee.

A blockchain [21] is a hash-based data structure, which is a publicly verifiable ledger in bitcoin. Each block has a hash pointer to the previous block and a Merkle hash tree (MHT) is employed to integrate transactions into one block in an efficient way. In blockchain network, a large number of nodes known as miners mine a new block by running a consensus algorithm such as proof of work (POW), proof of stake (PoS) etc. The pros of blockchain benefit a wide domain such as supply chains [1], financial transaction [7,16,17] and internet of things (IoT) [19]. The decentralization of blockchain paved for data trading in big data market.

Related work. Zhou et al. [35] proposed distributed data vending framework based on blockchain by combining the data embedding and similarity learning. They got the trade-off between data retrieval and leakage risk by indexing the data. Delgado-Segura et al. [10] proposed a fair data trading protocol based on Bitcoin transaction. They used Bitcoin script language[1] to construct Bitcoin transaction, but the operation code OP_AND they used is disabled in real Bitcoin transaction. Chen et al. [6] presented a blockchain-based ecosystem for big data exchange. The blockchain technology was used to record transaction logs and other significant documents. Niu et al. [22] proposed TPDM, which integrates trust and privacy preserving in data markets by using homomorphic encryption and identity-based signature. Goldfeder et al. [13] raised the problem of buying physical goods which present a suit of schemes by using bitcoin to improve security and privacy properties. Campanelli et al. [5] defined the notion of zero-knowledge contingent service payment (ZKCSP) to purchase digital services. Kur et al. [18] proposed trustless machine learning contracts, which exchange machine learning models on Ethereum blockchain. However, these mechanisms are inadequate for ensuring privacy and availability for fair data trading in blockchian-based big data markets.

Our contributions. In this paper, we propose a new blockchain-based fair data trading protocol in big data market to conduct privacy-preserving, availability and fair data trading. The advantage of blockchain infrastructures is removing single-point-failure of big data market. The protocol makes use of ring signatures to enhance privacy of data provider identity, extends the double authentication preventing signature (DAPS) for fair data trading and utilizes similarity learning to guarantee the availability of trading data. We implement the protocol with smart contract in Solidity to demonstrate the validity of the blockchain-based fair data trading protocol.

1.1. Organization

The reminder of this paper is organized as follows. We describe blockchain-based data trading model in big data market in Section 2. Some building blocks of our protocol are given in Section 3. The details of blockchain-based fair data trading are described in Section 4, and the security analysis and evaluation are given in Sections 5 and 6. Finally, we conclude the paper in Section 7.

2. Blockchain-based data trading model in big data market

In the section, we introduce the blockchain-based data trading model and relevant security requirements.

2.1. Blockchain-based data trading model in big data market

As shown in Fig. 2. there are four participants including data provider, blockchain network, market manager and data consumer in the system. In big data market, data trading [2] is the exchange of data between a data provider and a data consumer. The market manager establishes a platform for data providers to vend data and a data consumer to purchase data.

[1] https://bitcoin.org/en/developer-reference#raw-transaction-format.

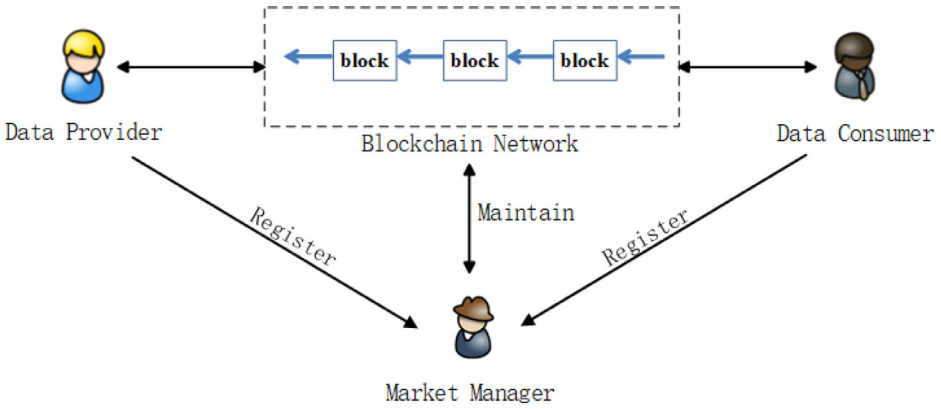

Fig. 2. Blockchain-based data trading model in big data market.

Data provider: The data provider is usually the owner of the data. A data provider has a list of topics to advertise the selling data, registers to the market manager and generates a topic transaction for vending data.

Market manager: The market manager maintains the blockchain network and provides a platform for data trading between the data providers and the data consumers. The market manager generates the system public parameters but does not participate the data trading.

Blockchain network: The blockchain network is a peer-to-peer network with a great many nodes. The blockchain network is a secure environment which has enough honest miners to defeat 51% attack. The data provider and the data consumer can conduct transactions on the blockchain for data trading.

Data consumer: The data consumer is usually the buyer of the data. He registers to the market manager and interacts with data providers to query data. Then, the data consumer generates the payment transaction to purchase data.

Definition 1 (Blockchain-based Data Trading protocol in Big Data Market)**.** The data trading protocol is a tuple of polynomial time algorithms (*Setup, Register, Publish, Query,* $Sign_1$, *Payment,* $Sign_2$, *Acc*) such that:

- $(pp) \leftarrow Setup(\lambda)$: This is a probabilistic algorithm that takes a security parameter λ as input and outputs the public parameters *pp*.
- $(PK, SK) \leftarrow Register(pp, \mathcal{ID})$: This algorithm takes *pp* and the identity $\mathcal{ID}$ of the data provider or the data consumer as input and outputs the corresponding public key *PK*, and secret key *SK* of $\mathcal{ID}$.
- $(T_P) \leftarrow Publish(pp, Topic, SK)$: The algorithm takes *Topic, pp* and *SK* of the data provider as input and outputs the transaction T_P.
- $(1/0) \leftarrow Query(pp, Topic)$: The algorithm takes *Topic, pp* as input and outputs a bit $b \in \{0, 1\}$.
- $(\sigma_1) \leftarrow Sign_1(pp, PK, SK)$: The algorithm takes the parameter *pp, PK, SK* as input and outputs a signatures σ_1.
- $(T_{Pay}) \leftarrow Payment(pp, \sigma_1, PK)$: The algorithm takes *pp,* σ_1 and *PK* as input, outputs the payment transaction T_{Pay}.
- $(T_{Sign}) \leftarrow Sign_2(pp, SK)$: The algorithm takes *pp, SK* as input, outputs the signature σ_2 and the transaction T_{Sign}.
- $(\mathcal{ID}) \leftarrow Acc(pp, ak, T_P)$: The algorithm takes *pp, ak,* T_P as input, outputs the identity $\mathcal{ID}$ of the data provider.

2.2. Security requirements

The following security requirements should be achieved in a blockchain-based data trading protocol.

Completeness: The completeness says that if the setup is properly and the data provider behaves honestly at all epochs, then an honest data consumer can obtain the valid data and the honest data provider can obtain the payment.

Confidentiality: A malicious participant such as a data consumer without the payment is unable to obtain the data.

Anonymity: To preserve privacy, a data trading protocol should guarantee the anonymity of the data provider. A data consumer executes data trading does not know the real identity $\mathcal{ID}$ of the data provider.

Availability: If the data provider behaves honestly, then the data consumer can obtain the data. If a malicious data provider gives invalid data, he will be punished in the sense that he cannot take his deposit back.

Fairness: The fairness says that, at the end of the protocol, either the data consumer gets valid data and the data provider gets payment or neither the data consumer and the data provider get nothing.

Accountability: The accountability means that a malicious data provider who provides invalid data, the market manager is able to reveal the identity of the data provider.

3. Building blocks

In this section, we review some building blocks used in our protocol.

3.1. Bilinear map

Let n be a composite number with factorization $n = pq$. G and G_T are two multiplicative cyclic groups of order n. g is the generation in group G. Then $e : G \times G \longrightarrow G_T$ is a bilinear map if it satisfies the following properties.

- Bilinear: for all $u, v \in G$ and $a, b \in Z$, $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degenerate: $e(g, g)$ is a generator of group G_T.
- The group operations on group G and G_T can be performed efficiently.

3.2. Complexity assumption

We introduce subgroup decision problem [4] as follows. The parameters n, p, q, G, G_T, g have the same settings as those in Section 3.1. Besides, G_p and G_q are two subgroups of G with order p and q respectively.

Definition 2 (Subgroup decision problem [4])**.** Given h selected randomly either from G (with probability 1/2) or from G_q (with probability 1/2), decide whether h is in G_q. For this problem one is given the description of G, but not given the factorization of n.

The assumption of subgroup decision problem is hard is called the subgroup hiding (SGH) assumption in [29].

3.3. Ring signatures

The concept of ring signature was first introduced by Rivest et al. in 2001 [26,27]. In a ring signature, a user chooses a group of users called a ring to generate a signature. A verifier is convinced that the signature is generated by a member of the ring, but cannot reveal which menber actually generated the signature. We adopt of the practical instantiation due to Shacham and Waters [29] is accountability by suitable variation [12]. We review the notations in the ring signature where R denotes a ring of the size k.

RS.Setup: The algorithm inputs a security parameter 1^λ to generate two cyclic groups G and G_T of order $n = pq$ and the subgroup G_q of G with order q. e denotes a bilinear map $e : G \times G \longrightarrow G_T$. Then, it chooses generators $g \in G$, $h \in G_q$, randomly chooses $b_0 \in Z_n$, and sets $B_0 = g^{b_0}$. Let $H : \{0,1\}^* \to \{0,1\}^k$ be a collision-resistent hash function and it chooses u', u_1, u_2, $\cdots$, $u_k \in G$. The public parameters $pp = (G, G_T, e, B_0, g, h, u', u_1, u_2, \cdots, u_k, H)$, and $ak = q$ as the accountable key.

RS.KGen: To generate the private key of a signer with index j, this algorithm chooses a random $a_j \in \mathbb{Z}_n$ and computes $pk_j = g^{a_j}$, $sk_j = a_j$.

RS.Sign: It inputs a message $M \in \{0,1\}^*$, a ring R of public keys, and the signer's key pair (pk_s, sk_s). This algorithm computes $H(M, R) \longrightarrow (m_1, m_2, \cdots, m_k)$. Let $k = |R|$, parse the elements of R as $pk_j \in G$, $1 < j < k$. Define $\{f_i\}_{j=1}^k$ as

$$f_j = \begin{cases} 1 & if\ \ j = s, \\ 0 & otherwise. \end{cases}$$

For each j, $1 < j < k$, this algorithm randomly chooses $t_j \in \mathbb{Z}_n$ and computes

$$c_j = (pk_j / B_0)^{f_j} h^{t_j}$$

$$\pi_j = ((pk_j / B_0)^{2f_j - 1} h^{t_j})^{t_j}$$

Then, it sets $t = \sum_{j=1}^k t_j$ and randomly chooses $r_0, r_1 \in \mathbb{Z}_n$ to compute

$$\sigma_1 = g^{r_0 a_s} \cdot \left(u' \prod_{j=1}^{k} u_j^{m_j} \right)^{r_1} \cdot h^{r_0 t},$$

$$\sigma_2 = g^{r_0},$$

$$\sigma_3 = h^{r_0},$$

$$\sigma_4 = g^{-r_1},$$

and outputs the signature $\sigma = (\sigma_1, \sigma_2, \sigma_3, \sigma_4, c_1, \cdots, c_k, \pi_1, \cdots, \pi_k)$.

RS.Ver: For each j, $1 < j < k$, this algorithm verifies $e(c_j, c_j/(pk_j/B_0)) \stackrel{?}{=} e(h, \pi_j)$. Then, it computes $C = \prod_{j=1}^k c_j$ and verifies

$$e(\sigma_2, B_0 C) \stackrel{?}{=} e(\sigma_1, g) \cdot e\left(\sigma_4, u' \prod_{j=1}^{k} u_j^{m_j} \right), \qquad e(\sigma_2, h) \stackrel{?}{=} e(g, \sigma_3)$$

RS.Acc: It inputs accountable key ak, a ring signature σ. For $1 < j < k$, if $(c_j)^{ak} \cdot B_0 \stackrel{?}{=} pk_j$, outputs the signer's public key pk_j.

3.4. Double-authentication-preventing signatures

The notation of double authentication preventing signature (DAPS) was introduced by Poettering and Stebila in 2014 [24,25]. A major application of DAPS is to punish the signer who generated two signatures of the messages with same heads and different payloads, such as penalizing double spending of transactions in bitcoin or defeating compelled certificate creation attack. We apply the DAPS carry out fair data trading and adopt the practical instantiation of DAPS scheme [11]. Here, $\prod = (KGen_{\prod}, Sign_{\prod}, Ver_{\prod})$ denotes a generic signature algorithm. The generic DAPS in the discrete logarithm (DL) setting as follows.

DAPS.Setup: The algorithm inputs a security parameter 1^{λ} and generates cyclic groups G_1 with order q. Let $H_1 : \{0,1\}^* \to \{0,1\}^l$ be collision-resistent hash function. Then, it chooses generator $g_1 \in G_1$ and common reference string crs.

DAPS.KGen: It randomly chooses $x_1 \in \mathbb{Z}_q^*$ as the secret key and computes $pk_1 = g_1^{x_1}$. Then, calls the generic algorithm $KGen_{\prod}$ to generate $(sk_{\prod}, pk_{\prod})$. Next, randomly chooses $a, b \in \mathbb{Z}_q^*$ and computes $Z = (Z_1 = g_1^a, Z_1' = pk_1^a g_1^b)$ and $Y = (a,b) \bigoplus H_1(sk_{\prod})$. Then, it sets $sk = (sk_{\prod}, (a,b))$ and $pk = (pk_{\prod}, pk_1, crs, Y, Z)$.

DAPS.Sign: Take a message $m \in \{0,1\}^*$ and sk as input, this algorithm parses m as (i, p) and computes the signature σ_1 as follows.

$$
\begin{array}{|l|}
\hline
\sigma' = Sign_{\prod}(sk_{\prod}, m) \\
z = bp + sk_{\prod} \\
Z'' = Z_1' \cdot (pk_{\prod} \cdot g_1^{-z})^{\frac{1}{p}} \\
\pi \leftarrow \mathrm{proof}(crs, (g_1, pk_1, Z_1, Z''), a) \\
\mathrm{sign}\ \sigma_1 = (\sigma', z, \pi) \\
\hline
\end{array}
$$

DAPS.Ver: Take the message $m \in \{0,1\}^*$, $\sigma_1 = (\sigma', z, \pi)$ and the public key $pk = (pk_{\prod}, pk_1, crs, Y, Z)$ as input, this algorithm verifies, if $Ver_{\prod}(pk_{\prod}, m, \sigma') \stackrel{?}{=} 1$ holds. If it holds, this algorithm computes $Z'' = Z_1' \cdot (pk_{\prod} \cdot g_1^{-z})^{\frac{1}{p}}$, and validates the proof $(crs, (g_1, pk_1, Z_1, Z''), \pi)$.

DAPS.Ext: Take (m_1, σ_1), (m_2, σ_2) and the public key pk as input, this algorithm verifies

$$\mathrm{DAPS.Ver}(pk, m_i, \sigma_i) \stackrel{?}{=} 1 \quad \text{for} \quad i = (1,2)$$

If it holds, this algorithm extracts the secret key $sk = (sk_{\prod}, (a,b))$ as follows.

$$sk_{\prod} = z_1 \frac{p_2}{p_2 - p_1} + z_2 \frac{p_1}{p_1 - p_2}$$

$$(a,b) = Y \bigoplus H_1(sk_{\prod})$$

The DAPS is provably secure in the random oracle model. It can be instantiated by elliptic curve digital signature algorithm [14] or Schnorr signatures [28] and interested readers can refer to [11].

3.5. Similarity learning

The distance metric learning has been extensively studied for decades and widely applied to computer vision, information retrieval system and bioinformatics [3]. It can immensely enhance the performance in classification, clustering and retrieval tasks [33]. Distance metric learning is to learn distance relation of specified data from pairwise similar/dissimilar points. In information retrieval systems, we can define similarity on specified data and learn a distance function $d(X, Y)$ to conduct efficient retrieval. Let $X_i \in \mathbb{R}^m$ denote a data vector where m is the number of features. Let $\mathcal{C} = \{X_1, \cdots, X_n\}$ denote samples collection of data points, where n is its size. The Mahalanobis distance [23] between any two vector X, Y can be expressed by

$$d_M(X,Y) = \sqrt{\|X - Y\|_M^2} = \sqrt{(X-Y)^T M (X-Y)}$$

where M is the parameter matrix of distance metric. We use distance metric learning to compute the similarity of plaintext data and decision making. The metric learning algorithm is extended to the multi-task setting when there are many tasks in [23,34]. In the multi-task setting [23], the distance for task t is defined as

$$d_t(X,Y) = \sqrt{(X-Y)^T (M_0 - M_T)(X-Y)}$$

The specific regularization term is defined as

$$\min_{M_0, \cdots, M_T} = \gamma_0 \|M_0 - I\|_{\mathrm{F}}^2 + \sum_{t=1}^{T} \gamma_t \|M_t\|_{\mathrm{F}}^2$$

The parameter γ_t controls the regularization of M_t where $t = 0, \ldots, T$. The multi-task metric learning is able to effectively improve the performance of distance metrics that learned for retrieval tasks and interested readers can refer to [23].

3.6. Smart contract

Smart contract, a self-executing program code, was firstly proposed by Nick Szabo [30]. Smart contract is derived from the Bitcoin scripting language that is a stack based language but not turning-complete. The Ethereum [32] is another cryptocurrency to build the next-generation distributed applications which supports the smart contract. Ethereum offers Turning-complete languages with more expressive expressions. Solidity[2] is the most widely developed language for writing smart contract.

4. Blockchain-based fair data trading protocol

4.1. Overview

The protocol works as follows. First, the market manager initializes the system parameters. Then, the data provider and the data consumer register with the market manager as new users. When the data provider registers on the market manager, he has to pay deposit to the market manager. Next, the data provider publishes the topic lists on the blockchain. The data consumer searches on the blockchain and requests data about certain topics. The data provider sends the encrypted data to the data consumer. After that, the data consumer randomly challenges some data blocks and the data provider responds the challenged content to the data consumer. Then, the data consumer evaluates the content and decision making by taking advantage of the distance metric learning technique. If the data consumer responds to purchase the data, the data provider sends the first DAPS signature to the data consumer. The data consumer publishes contingent payment smart contract to blockchain network. Finally, the data provider calls smart contract and sends the second signature of DPAS to blockchain to receive the payment. The data consumer can extract the secret key by using two DAPS signatures to decrypt the ciphertext. If the data consumer is unable to decrypt all ciphertexts, he complains to the market manager. The manager can reveal the identity of the data provider and punish him by transferring his deposit to the data consumer. We assume the deposit is more than the real payment and the market manager does not collude with the data consumer. In the system, each entity has his/her ECDSA key pair (PK, SK) and the $Sign_{SK}(m)$ denotes the ECDSA signature on message m.

4.2. The details of protocol

The framework of the proposed protocol is shown in Fig. 3. The data are divided into independent chunks and we adopt hybrid encryption by combining AES [9] (AES.enc denotes the AES encryption algorithm and AES.dec denotes the AES decryption algorithm on data m) with public key encryption. The public key encryption is used to encrypt the secret key of AES as $C = (C_1 = AES.enc_k(m), C_1' = Enc_{PK}(k))$.

- *Setup*: The market manager runs RS.Setup to generate the public parameters $pp = (G, G_T, e, B_0, g, h, u', u_1, u_2, \cdots, u_k, H)$, and the accountable key ak. Then, he calls DAPS.Setup to generate cyclic group G_1 and chooses generator g_1 and common reference string crs. Let H_1 is a collision-resistent hash function.
- *Register*: The data provider and the data consumer register to the market manager. The data provider chooses a random $x \in \mathbb{Z}_n$ and calls RS.KGen to generate key pair $(PK_1 = g^x, SK_1 = x)$. Then, he calls DAPS.KGen to generate DAPS key pair $SK_2 = (sk_{\prod}, (a, b))$ and $PK_2 = (pk_{\prod}, pk_1, crs, Y, Z)$. He submits $(PK_1, PK_2, \mathcal{ID}_p)$ and the deposit to the market manager. The manager maintains a list L to record the identity $\mathcal{ID}_p, PK_1, PK_2$. The data consumer generates his ECDSA key pair (PK_3, SK_3) and submits $(\mathcal{ID}_c, PK_3)$ to the manager. Then, the manager updates list L.
- *Publish*: The data provider builds a topic list L'. Then, he randomly chooses the public key set R in list L and computes $H(L', R) \longrightarrow (m_1, m_2, \cdots, m_k)$, generates a ring signature σ on L' and embeds σ to the topic transaction T_P.
- *Query*: The data consumer searches on the blockchain the topic he is interested in. Then, the data consumer queries the content of the topic in list L'. The data provider chooses AES secret key $(k_1, \cdots, k_n)$ to encrypt message chunks $(m_1, \ldots, m_n)$. Then, uses PK_2 to encrypt the secret key $(k_1, \ldots, k_n)$. The data provider forwards the ciphertext $C = (C_1, \ldots, C_n) = ((AES.enc_{k_1}(m_1), Enc_{PK_2}(k_1)), \ldots, (AES.enc_{k_n}(m_n), Enc_{PK_2}(k_n)))$ to the data consumer. When the data consumer receives ciphertext C, he randomly challenges the subset of message chunks, that is $i \in [1, \ldots, n]$ in $(m_1, \ldots, m_n)$, and sends the challenge subset to the data provider. The data provider returns the challenged unencrypted pieces of message and the corresponding AES secret key as a correctness proof to the data consumer. Upon receiving the proof, the data consumer uses the AES secret key to encrypt the pieces of message and then compares it with the ciphertext. If the proof is valid, the data consumer uses similarity learning to make a decision whether to purchase the data or not.

[2] http://solidity.readthedocs.io/en/latest.

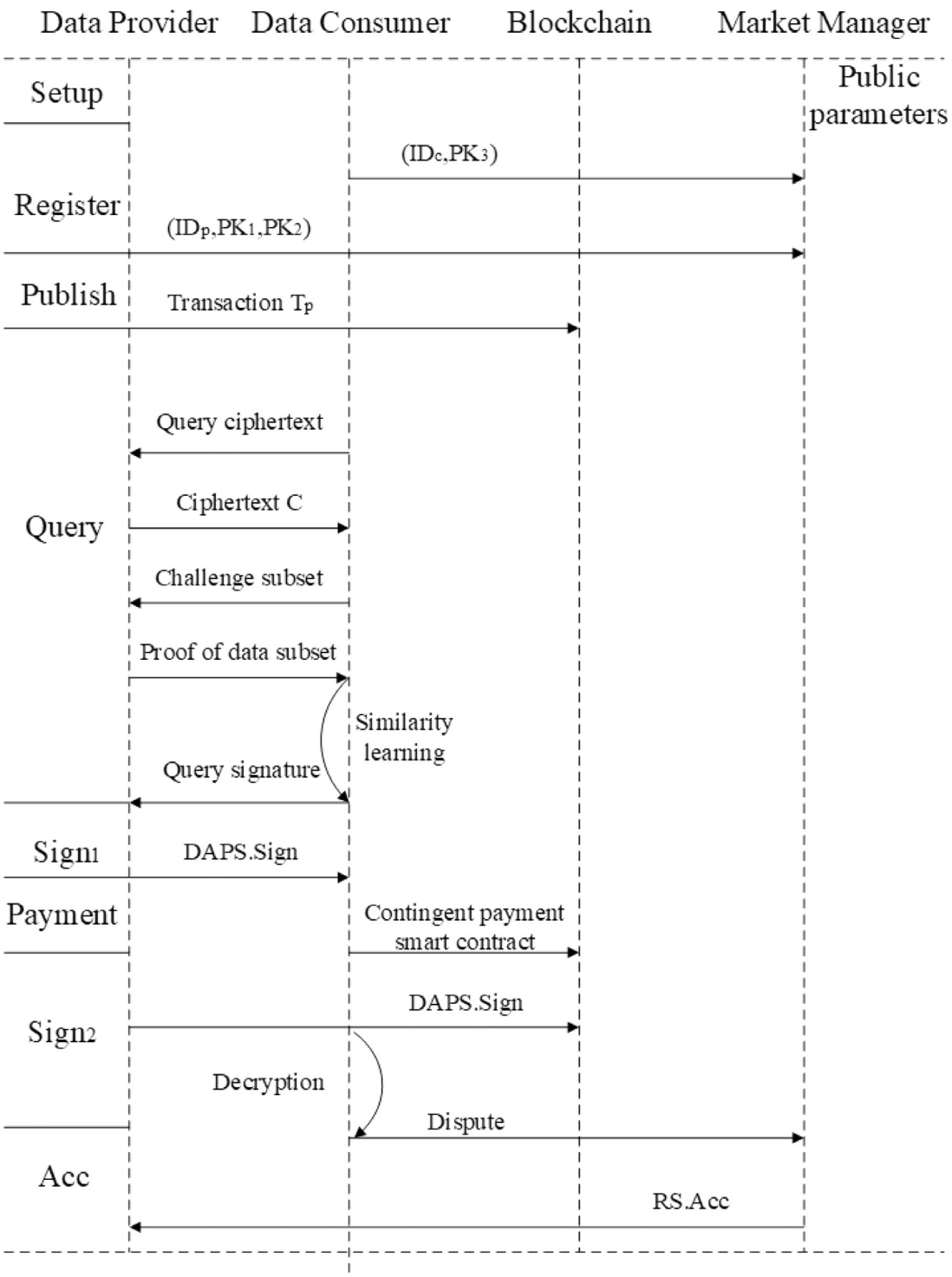

Fig. 3. The framework of the proposed protocol.

- *Sign*$_1$: If the *Query* algorithm returns 1, the data provider calls DAPS.Sign to generate the first DAPS signature $\sigma_1 = (\sigma', z, \pi)$. The data provider encrypts σ_1 with the data consumer's public key PK_3 and sends it to the data consumer. Then, the data consumer decrypts and verifies the signature σ_1.
- *Payment*: If σ_1 is valid, the data consumer conducts a contingent payment smart contract shown in Table 1 and dummyTXdummy- publishes it to blockchain network. Here, $ denotes the value of payment. The data consumer sets the limitation time T as contingent payment time epoch.

Table 1
Contingent payment smart contract.

Function payment(): 1) The data consumer pays $. 2) The data consumer sets limitation time T.
Function transfer(): 1) Assert current time $T_1 < T$. a) Call *DAPS.Ver*. b) Send $ to data provider. 2) Assert current time $T_1 > T$. a) Send $ to data consumer.

- $Sign_2$: The data provider generates a new DAPS signature σ_2. Then, he calls smart contract function Transfer() to conduct transaction T_{Sign}. Once the transaction T_{Sign} is included in blockchain, the data consumer uses σ_1 and σ_2 to extract the secret key SK_2 corresponding to public key PK_2 and utilizes SK_2 to decrypt the ciphertext.
- *Acc*: If the data consumer cannot decrypt all the ciphertexts, he complains to the market manager. Then, the market manager uses the accountable key *ak* to reveal the identity of the data provider in the transaction T_p and transfers the data provider's deposit to the data consumer.

Correctness. If σ_1 and σ_2 are valid, the data consumer can extract the secret key SK_2. Then, he uses SK_2 to decrypt the ciphertext in *Query* phase. The data consumer is able to recover $SK_2 = sk_{\prod} = z_1 \frac{p_2}{p_2 - p_1} + z_2 \frac{p_1}{p_1 - p_2}$, decrypts $k_i = Dec(SK_2, C_i)$ and $m_i = AES.dec_{k_i}(C_i)$.

According to *Acc* algorithm, the data consumer can obtain the deposit when he is unable to decrypt all ciphertexts. The manager can reveal the identity of data provider and punish him by transferring his deposit to the data consumer. If there is no dispute, the data provider can take his deposit back.

5. Security analysis

In this section, we analyze the security of the blockchain-based fair data trading protocol. The security of the proposed protocol is guaranteed by following lemmas.

Lemma 1. *The proposed blockchain-based fair data trading protocol achieves completeness.*

Proof. If the data provider honestly follows the data trading protocol described in Section 4.2, then from the completeness of the underlying DAPS scheme, the data consumer can always recover the secret key and decrypt the ciphertext to obtain the data, and at the same time, the provider can receive the payment. □

Lemma 2. *The proposed blockchain-based fair data trading protocol achieves confidentiality.*

Proof. As for outside adversary such as the market manager, the data is encrypted by PK_2 and AES, it is hard to recover the plaintext without the secret key. The DAPS signature σ_1 is encrypted and sends to the data consumer. The outside adversary can obtain the σ_2 from the blockchain, but he cannot extract the secret key without the σ_1. As for the inside adversary, the data consumer without payment for data, he is unable to obtain σ_2. Without σ_2, the data consumer can not extract the secret key to decrypt the ciphertext. □

Lemma 3. *The proposed blockchain-based fair data trading achieves anonymity if SGH assumption holds.*

Proof. In our protocol, the anonymity of the data provider is guaranteed by ring signature. We borrowed the security model defined in [29], which is a game on anonymity against full key exposure between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$. Here, we prove the security of our protocol in game-based framework. We denote the advantage of $\mathcal{S}$ in $Game_i$ by $Adv_{\mathcal{S}}^{gamei}$. □

Game 0: Game 0 is the original game defined in [29]. Specifically, $\mathcal{S}$ generates the group G with order n. g, h are the generators of group G. $\mathcal{S}$ chooses a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^k$ and calls the *RS.Setup* algorithm to obtain the system parameters $pp = (G, G_T, e, B_0, g, h, u', u_1, u_2, \ldots, u_k, H)$. $\mathcal{S}$ runs *RS.KGen* for k times to obtain key pairs $\{pk_j, a_j\}_{j=1}^k$ for the ring members. Then, $\mathcal{S}$ sends adversary $\mathcal{A}$ the system parameters pp, ring public key $\{pk_j\}_{j=1}^k$. The $Game_0$ is the original scheme, we have

$$Adv_{\mathcal{S}}^{game0} = Adv_{\mathcal{A}}^{Ano}.$$

Game 1: Game 1 is the same as Game 0, with one difference. h is the generator in group G_q rather than group G. If there is a difference for $\mathcal{A}$ between the advantage in Game 0 and Game 1, we can define a new algorithm $\mathcal{C}$ to break SGH problem, for which we have

$$\begin{aligned} Adv_{\mathcal{C}}^{SGH} &= |Pr[\mathcal{S}=1|h\in G_q] - Pr[\mathcal{S}=1|h\in G]| \\ &= \tfrac{1}{2}|2(Pr[\mathcal{S}=1|h\in G_q]-1) - (2Pr[\mathcal{S}=1|h\in G]-1)| \\ &= \tfrac{1}{2}|Adv_{\mathcal{S}}^{game0} - Adv_{\mathcal{S}}^{game1}| \end{aligned}$$

Consider the challenge $(\sigma_1, \sigma_2, \sigma_3, \sigma_4, c_1, \ldots, c_k, \pi_1, \ldots, \pi_k)$ in the security model, for each j, $c_j = (pk_j/B_0)^{f_j} h^{t_j}$, h is the generator of G, there exist τ_0, $\tau_1 \in Z_n$, such that $c_j = (pk_j/B_0)^{f_j} h^{\tau_0} = h^{\tau_1}$, so we have

$$(\pi_j | f_j = 0) = ((pk_j/B_0)^{-1} h^{\tau_0})^{\tau_0} = ((pk_j/B_0) h^{\tau_1})^{\tau_1} = (\pi_j | f_j = 1)$$

for each pair (c_j, π_j). The adversary $\mathcal{A}$ can not obtain information from this part of signature. Now, we consider the value σ_1, σ_2, σ_3, σ_4, which contains no information of the signer. The value σ_1 satisfies $e(\sigma_1, g) \cdot e(\sigma_4, u' \prod_{j=1}^{k} u_j^{m_j}) = e(\sigma_2, B_0 C)$. Let $h = g^w$, we have

$$\begin{aligned} \sigma_1 &= g^{r_0 a_i} \cdot (u' \prod_{j=1}^{k} u_j^{m_j})^{r_1} \cdot h^{r_0 t} \\ &= g^{r_0 a_b + w r_0 t} \cdot (u' \prod_{j=1}^{k} u_j^{m_j})^{r_1} \\ &= g^{c} \cdot (u' \prod_{j=1}^{k} u_j^{m_j})^{r_1}, \text{where } c = r_0 a_b + w r_0 t \end{aligned}$$

This value gives no information of the secret key sk_{j_0}, sk_{j_1} used to generate the signature, and adversary $\mathcal{A}$ can only guess b. Thus, we have $Adv_{\mathcal{S}}^{game1} = 0$ and $Adv_{\mathcal{A}}^{Ano} \le 2Adv_{\mathcal{C}}^{SGH}$. If the $Adv_{\mathcal{A}}^{Ano}$ is non-negligible, so is $Adv_{\mathcal{C}}^{SGH}$.

Lemma 4. *The proposed blockchain-based fair data trading protocol achieves availability.*

Proof. If the data provider executes the data trading honestly, then the honest data consumer can use the similarity learning to make the decision that whether to purchase the data. A malicious data provider who gives the invalid encrypted data or the secret key, he will be punished. As for the malicious data consumer, he can not obtain the secret key to decrypt the ciphertext. □

Lemma 5. *The proposed blockchain-based fair data trading protocol achieves fairness.*

Proof. First, we consider the data provider is malicious, in the sense that he can get payment without submitting the second DAPS. In this case, according to the contingent payment smart contract, the miner can not verify the transaction and the data consumer can withdraw the payment in the limitation time T. The data provider can get payment without providing the trustful data and the deposit. In this case, the data provider does not provide the trustful data, the data consumer challenges some data blocks randomly and uses the similarity learning to make the decision. If the data consumer has generated contingent payment smart contract but the secret key is error, he can turn to the market manager to reveal the identity of the data provider in list L. Then, the market manager pays the deposit of data provider to consumer and the data provider does not obtain deposit back. In this case, a malicious data provider gets a contradiction. The probability of success for a malicious data provider is negligible.

Then, we consider the case that the data consumer is malicious and the data provider is honest. The data consumer gets the data but does not pay the data. The data provider never submits the DAPS to data consumer without contingent payment smart contract. So, the data consumer is unable to extract the secret key to decrypt. We say that is contradiction that he gets data without payment. The data consumer can cheat with only negligible probability. Therefore, the proposed blockchain-based fair data trading protocol achieves fairness. □

Lemma 6. *The proposed blockchain-based fair data trading protocol achieves accountability.*

Proof. If the data provider is malicious, the market manager can call RS.Acc algorithm, for $1 < j < k$, if $(c_j)^{ak} \cdot B_0 \stackrel{?}{=} pk_j$, and output the signer's public key pk_j. Then, he searches the list L to reveal the identity of the data provider. The manager pays the deposit of data provider to the data consumer and the data provider does not get the deposit back. □

6. Performance evaluation

In this section, we report performance evaluation of our protocol with smart contract in Solidity. Since Solidity does not provide the application programme interface (API) of RS and DPAS for the time of research, we will separately quantify the time cost of cryptographic algorithms and the gas cost of smart contract. We execute cryptographic algorithms with the pairing-based library JPBC 2.0.0[3] for our simulation which the Type A_1 pairing on the curve $y^2 = x^3 + x$. The Type A_1 pairing supports the composite order group operation.

[3] http://gas.dia.unisa.it/projects/jpbc/index.html#.VTDrLSOl_Cw.

Table 2
The time cost of RS algorithms.

RS	Size of ring	Setup	KGen	Sign	Ver	Acc
Max time	10	2.147 s	2.681 s	3.716 s	3.768 s	0.287 s
Min time	10	1.430 s	2.208 s	3.291 s	3.038 s	0.113 s
Average time	10	1.827 s	2.285 s	3.397 s	3.241 s	0.229 s

Table 3
The time cost of DAPS algorithms.

DAPS	Setup	KGen	Sign	Ver	Ext
Max time	0.007 s	0.134 s	0.146 s	0.165 s	0.266 s
Min time	0.002 s	0.059 s	0.084 s	0.088 s	0.175 s
Average time	0.004 s	0.070 s	0.092 s	0.093 s	0.186 s

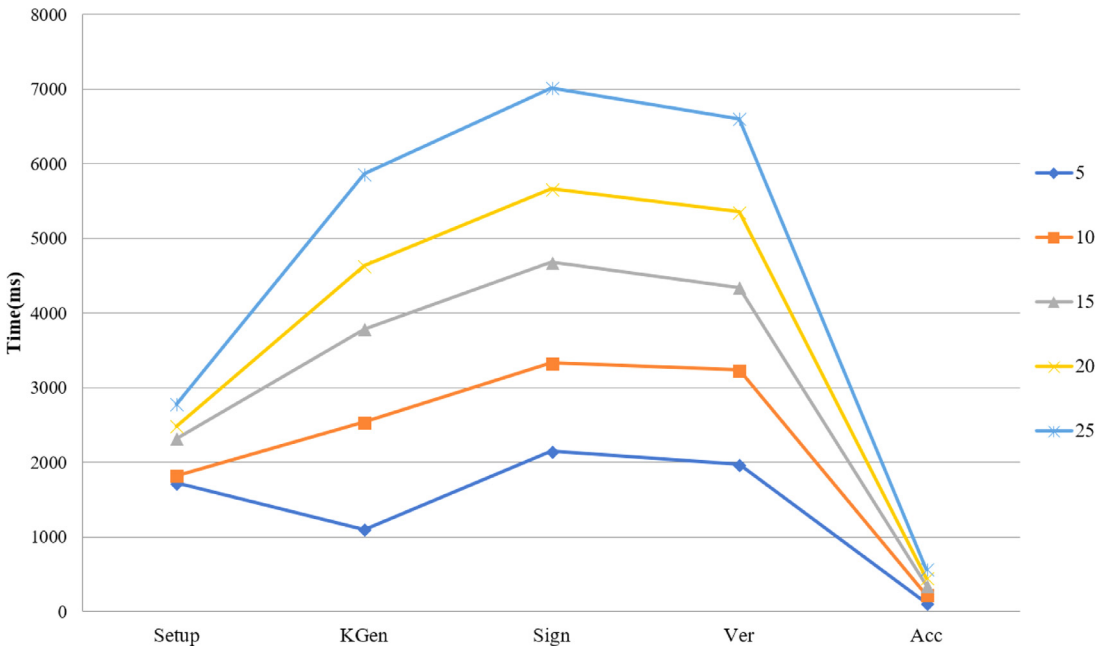

Fig. 4. The time cost of RS on different size of ring.

The simulation platform is Intel(R) Core(TM) i5-2450M CPU 2.50 GHz 4.00GB RAM and Windows 7 operate system. The time cost of cryptographic algorithms is shown in Table 2 (the size of ring is 10) and Table 3 with the average of 100 runs. We evaluate the time cost of RS on different size of ring in Fig. 4. See Table 2 and dummyTXdummy-Fig. 4 for more details. The RS.Setup algorithm generates the composite order group, picks some random values and computes a modular exponentiation in G, which costs 1.827 s. The RS.KGen algorithm costs 2.285 s, which includes the time to generate ten key pairs $(pk_i, sk_i)(1 \le i \le 10)$. If a data provider only generates one key pair (pk, sk), it costs 0.285 s. The RS.Sign algorithm and RS.Ver algorithm have different time cost with different ring size in which the data provider can find the trade-off between anonymity and time cost by choosing a proper ring size. We can see that the RS.Acc algorithm is extremely fast, thus the market manager can achieve accountability efficiently. From Table 3, we can see the DAPS.Setup algorithm and the DAPS.KGen algorithm are cheap. The DAPS.Setup algorithm chooses the generator in G_1 and generates the system parameters, which costs 0.004 s. The DAPS.KGen algorithm performs modular exponentiation in G_1, which costs 0.070 s. The DAPS.Sign algorithm and the DAPS.Ver algorithm are expensive since it includes to generate and verify a non-interactive zero-knowledge proof. The DAPS.Ext algorithm contains two DAPS.Ver algorithms, which costs 0.186 s.

We deploy the contingent payment smart contract in our protocol by Solidity. It executes on a local computer based on Web3j and the estimated gas cost to deploy contract is provided in Table 4.

Table 4
Gas cost of the smart contract.

Function	Transaction gas	Execute gas	Gas cost(ether)
Deploy contract	372675	247531	0.01240412
Deposit	104926	82054	0.0037396
Payment	42073	20609	0.00125364
Transfer	44469	21597	0.00132132

7. Conclusion and future work

In this paper, we propose a new blockchain-based fair data trading protocol in big data market, to enhance privacy, availability and fairness of data trading. The advantage of blockchain infrastructures is removing single-point-failure of big data market. We enhance anonymity of data provider and extend DAPS to data trading for fairness. At the same time, we use similarity learning to enhance the availability of trading data. Finally, We implement the protocol with smart contract in Solidity to demonstrate the validity of the blockchain-based fair data trading protocol.

Acknowledgement

This work was supported by National Key R&D Program of China (2017YFB0802000), National Natural Science Foundation of China (61872229), NSFC Research Fund for International Young Scientists (61750110528), National Cryptography Development Fund during the 13th Five-year Plan Period (MMJJ20170216) and Fundamental Research Funds for the Central Universities(GK201702004, 2018CBLY006).

References

[1] R. AlTawy, M. ElSheikh, A.M. Youssef, G. Gong, Lelantos: a Blockchain-Based Anonymous Physical Delivery System, Cryptology ePrint Archive, Report, 2017/465, 2017. http://eprint.iacr.org/2017/465
[2] M. Balazinska, B. Howe, D. Suciu, Data Markets in the Cloud: An Opportunity for the Database Community, in: Proceedings of the VLDB Endowment, volume 4, 2011, pp. 1482–1485.
[3] A. Bellet, A. Habrard, M. Sebban, A Survey on Metric Learning for Feature Vectors and Structured Data, 2013, arXiv:1306.6709,
[4] D. Boneh, E.J. Goh, K. Nissim, Evaluating 2-DNF Formulas on Ciphertexts, in: Theory of Cryptography Conference (TCC), Springer, 2005, pp. 325–341.
[5] M. Campanelli, R. Gennaro, S. Goldfeder, L. Nizzardo, Zero-Knowledge Contingent Payments Revisited: Attacks and Payments for Services, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS), ACM, 2017, pp. 229–243.
[6] J. Chen, Y. Xue, Bootstrapping a Blockchain Based Ecosystem for Big Data Exchange, in: 2017 IEEE International Congress on Big Data (BigData Congress), IEEE, 2017, pp. 460–463.
[7] A. Chiesa, M. Green, J. Liu, P. Miao, I. Miers, P. Mishra, Decentralized Anonymous Micropayments, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2017, pp. 609–642.
[8] C. Systems, Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper, 2017. Online document. http://www.tinyurl.com/zzo6766.
[9] J. Daemen, V. Rijmen, The Design of Rijndael: AES-the Advanced Encryption Standard, Springer Science & Business Media, 2013.
[10] S. Delgado-Segura, C. Pérez-Sola, G. Navarro-Arribas, J. Herrera-Joancomart, A fair protocol for data trading based on bitcoin transactions, Fut. Gene. Comput. Syst. (2017), doi:10.1016/j.future.2017.08.021.
[11] D. Derler, S. Ramacher, D. Slamanig, Short Double-and N-times-authenticationpreventing Signatures from ECDSA and More, Cryptology ePrint Archive, Report, 2017/1203, 2017. http://www.eprint.iacr.org/2017/1203.
[12] A. Escala, J. Herranz, P. Morillo, Revocable Attribute-Based Signatures with Adaptive Security in the Standard Model, in: International Conference on Cryptology in Africa, Springer, 2011, pp. 224–241.
[13] S. Goldfeder, J. Bonneau, R. Gennaro, A. Narayanan, Escrow Protocols for Cryptocurrencies: How to Buy Physical Goods Using Bitcoin, in: International Conference on Financial Cryptography and Data Security, Springer, 2017, pp. 321–339.
[14] D. Johnson, A. Menezes, S. Vanstone, The elliptic curve digital signature algorithm (ECDSA), Int. J. Inf. Secur. 1 (1) (2001) 36–63.
[15] T. Jung, X.Y. Li, W. Huang, J. Qian, L. Chen, J. Han, J. Hou, C. Su, Accounttrade: Accountable Protocols for Big Data Trading against Dishonest Consumers, in: IEEE Conference on Computer Communications(INFOCOM), IEEE, 2017, pp. 1–9.
[16] H. Kopp, C. Bösch, F. Kargl, Koppercoin-a Distributed File Storage with Financial Incentives, in: International Conference on Information Security Practice and Experience, Springer, 2016, pp. 79–93.
[17] H. Kopp, D. Mödinger, F. Hauck, F. Kargl, C. Bösch, Design of a Privacy-Preserving Decentralized File Storage with Financial Incentives, in: IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), IEEE, 2017, pp. 14–22.
[18] A.B. Kurtulmus, K. Daniel, Trustless Machine Learning Contracts, Evaluating and Exchanging Machine Learning Models on the Ethereum Blockchain, 2018 arXiv:1802.10185.
[19] O. Leiba, Y. Yitzchak, R. Bitton, A. Nadler, A. Shabtai, Incentivized Delivery Network of IoT Software Updates Based on Trustless Proof-of-Distribution, 2018, arXiv:1805.04282.
[20] F. Liang, W. Yu, D. An, Q. Yang, X. Fu, W. Zhao, A Survey on Big Data Market: Pricing, Trading and Protection, 6, IEEE Access, 2018, pp. 15132–15154.
[21] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, 2008, http://pdos.csail.mit.edu/6.824/papers/bitcoin.pdf.
[22] C. Niu, Z. Zheng, F. Wu, X. Gao, G. Chen, Trading Data in Good Faith: Integrating Truthfulness and Privacy Preservation in Data Markets, in: IEEE 33rd International Conference on Data Engineering (ICDE), IEEE, 2017, pp. 223–226.
[23] S. Parameswaran, K.Q. Weinberger, Large Margin Multi-task Metric Learning, in: Advances in Neural Information Processing Systems, 2010, pp. 1867–1875.
[24] B. Poettering, D. Stebila, Double-Authentication-Preventing Signatures, in: ESORICS, Part I, 8712, Springer, 2014, pp. 436–453.
[25] B. Poettering, D. Stebila, Double-authentication-preventing signatures, Int. J. Inf. Secur. 16 (1) (2017) 1–22.
[26] R.L. Rivest, A. Shamir, Y. Tauman, How to Leak a Secret, in: International Conference on the Theory and Application of Cryptology and Information Security, Springer, 2001, pp. 552–565.
[27] R.L. Rivest, A. Shamir, Y. Tauman, How to Leak a Secret: Theory and Applications of Ring Signatures, in: Theoretical Computer Science, Springer, 2006, pp. 164–186.

[28] C.P. Schnorr, Efficient Identification and Signatures for Smart Cards, in: Conference on the Theory and Application of Cryptology, Springer, 1989, pp. 239–252.
[29] H. Shacham, B. Waters, Efficient Ring Signatures without Random Oracles, in: International Workshop on Public Key Cryptography (PKC), Springer, 2007, pp. 166–180.
[30] N. Szabo, Smart Contracts, 1994, http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart.contracts.html.
[31] W.T. Wang, Y.S. Wang, E.R. Liu, The stickiness intention of group-buying websites: the integration of the commitment-trust theory and e-commerce success model, Inf. Manage. 53 (5) (2016) 625–642.
[32] G. Wood, Ethereum: a Secure Decentralised Generalised Transaction Ledger, Ethereum Project Yellow Paper, 2014. Online document. http://www.cryptopapers.net/papers/ethereum-yellowpaper.pdf.
[33] L. Yang, R. Jin, Distance Metric Learning: A Comprehensive Survey, Michigan State Universiy, 2(2), 2006, p. 4.
[34] P. Yang, K. Huang, C.L. Liu, Geometry preserving multi-task metric learning, Mach. Learn. 92 (1) (2013) 133–175.
[35] J. Zhou, F. Tang, H. Zhu, N. Nan, Z. Zhu, Distributed Data Vending on Blockchain, 2018. arXiv:1803.05871.