Blockchain-Cloud Transparent Data Marketing: Consortium Management and Fairness

Dongxiao Liu, *Member, IEEE*, Cheng Huang, *Member, IEEE*, Jianbing Ni, *Member, IEEE*, Xiaodong Lin, *Fellow, IEEE*, and Xuemin Sherman Shen, *Fellow, IEEE*

Abstract—Data are generated by Internet of Things (IoT) devices and centralized at a cloud server, that can later be traded with third parties, i.e., data marketing, to enable various data-intensive applications. However, the centralized approach is recently under debate due to the lack of (1) transparent and distributed marketplace management, and (2) marketing fairness for both IoT users (data sellers) and third parties (data buyers). In this paper, we propose a Blockchain-Cloud Transparent Data Marketing (*Block-DM*) with consortium management and executable fairness. First, we introduce a hybrid data-marketing architecture, where the cloud acts as an efficient data management unit and a consortium blockchain serves as a transparent marketing controller. Under the architecture, consent-based secure data trading and identity privacy for data owners are achieved with the distributed credential issuance and threshold credential openings. Second, with a consortium committee, we design a fair on/off-chain data marketing protocol. By financial incentives and succinct 'commitments' of marketing operations, the protocol can achieve the marketing fairness and effective detection of unfair marketing operations. We demonstrate the security of *Block-DM* with thorough analysis. We conduct extensive experiments with a consortium blockchain network on Hyperledger Fabric to show the feasibility and practicality of *Block-DM*.

Index Terms—Data marketing, blockchain, cloud computing, privacy, fairness

1 Introduction

The prevalence of the Internet of Things (IoT) has made the generation and collection of IoT data at an unprecedent rate. Such wealth of user data is of great trading value between data owners for pursuing economic benefits and third parties for developing data-intensive applications [1], [2]. However, it is cost-ineffective for data owners to manage the data trading locally due to the ever-increasing volume of the IoT data and the high demand for data transmission rate. Therefore, it is a promising solution for data owners to store and trade the IoT data at a powerful data center, i.e., cloud server [3]. By doing so, the data owners can enjoy the data services in a flexible and economic manner [4], [5], [6], [7].

There are essential requirements on realizing the full potential of the cloud-based data marketing: effective management and executable fairness. First, data marketing operations over the cloud should be effectively managed such that data confidentiality [8] and user identity privacy [9] should be preserved. More specifically, privacy regulations (e.g., General Data Protection Regulation (GDPR) [10]) require the data marketing to be transparent and reliable, where users should have rights to obtain information and control over their IoT data [11], [12]. Second, the marketing fairness guarantees (1) IoT users are paid well for selling their data over the cloud, and (2) third parties pay only if they receive the right data [13]. To this end, the cloud server is a centralized platform where users must rely on its trustworthiness to guarantee the data marketing transparency and reliability [1], [14], and the marketing fairness [15].

Blockchain [16] is a distributed ledger that is maintained by a peer-to-peer network with immutable on-chain storage and verifiable state updates, which can be utilized as a transparent and reliable 'controller' of the data trading [17], [18], [19], [20] or data processing [21], [22], [23]. With the smart contract technique [24], data owners, third parties, and the cloud servers can negotiate a trustworthy data usage agreement, that enforces consent-based access control over the IoT data [25] and records critical data sharing instances as provenance evidence [26] to manage the marketing fairness [27], [28]. At the same time, the blockchain-based data-marketing solution increases the consumer confidence by building a more trustworthy cloud ecosystem. However, there are still unresolved challenges on building the blockchain-based data marketing.

Many existing works adopt an on-chain marketing model [29], [30], [31], where all the data are stored or shared via the blockchain. This model can find some practical applications if the data volume is small, such as sharing of a secret value [29]. To reduce the on-chain storage and updating cost, an on/off-chain marketing model is investigated, where an off-chain cloud server stores the large-size data, the blockchain manages the data access control [5], [32], [33], [34], [35], and a single entity is utilized to manage IoT users' identities. At the same time, the marketing fairness issue in the on/off-chain

- *Dongxiao Liu, Cheng Huang, and Xuemin Sherman Shen are with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada. E-mail: {dongxiao.liu, cheng.huang, sshen}@uwaterloo.ca.*
- *Jianbing Ni is with the Department of Electrical and Computer Engineering, Queen's University and Ingenuity Labs Research Institute, Kingston, ON K7L 3N6, Canada. E-mail: jianbing.ni@queensu.ca.*
- *Xiaodong Lin is with the School of Computer Science, University of Guelph, Guelph, ON N1G 2W1, Canada. E-mail: xlin08@uoguelph.ca.*

Manuscript received 28 Jan. 2021; revised 20 Dec. 2021; accepted 31 Jan. 2022. Date of publication 14 Feb. 2022; date of current version 11 Nov. 2022.
(Corresponding author: Dongxiao Liu.)
Recommended for acceptance by J. Chen.
Digital Object Identifier no. 10.1109/TC.2022.3150724

model with rational data-marketing entities (data owner, data buyer, and cloud server) needs more research attention. Compared with the on-chain marketing model where communications among marketing entities are conducted via a public channel, the fairness issue becomes more challenging due to the use of an off-chain cloud storage.

In this paper, we propose a Blockchain-Cloud Transparent Data Marketing *(Block-DM)* to tackle the mentioned challenges. First, we introduce a set of supervising nodes to form an efficient consortium blockchain as a transparent and reliable "controller" for the cloud-based data marketing. Second, we utilize the supervising nodes to manage anonymous credentials for data owners with distributed credential issuance and threshold openings. Third, we carefully design the on/off-chain communications for the data owner, the third party, and the cloud server with financial incentives and succinct 'commitments' of honest behavior. The contributions of this paper are as follows:

- *Consortium Management:* We design a transparent and reliable data marketing architecture with the integration of the cloud server and the consortium blockchain, where the former acts as an efficient data storage unit and the latter serves as a transparent controller. *Block-DM* achieves right-to-be-informed, right-to-control, and conditional identity privacy for IoT users in the data trading over the cloud with a distributed committee.
- *Executable Fairness:* We design a fair on/off-chain data marketing protocol. With financial incentives and succinct commitments of correct marketing behavior, marketing entities are motivated to behave correctly, where honest users are well-paid and honest third parties get the right data.
- *Thorough Evaluation:* We formulate and achieve *consortium management* and *executable fairness* with detailed security analysis. We build a real-world consortium blockchain based on Hyperledger Fabric and conduct extensive experiments on multiple datasets to show that *Block-DM* achieves a feasible implementation cost and practical efficiency.

The remainder of the paper is organized as follows. We investigate related works in Section 2. In Section 3, we formulate system model and security model of *Block-DM*, and present security goals and design goals. Preliminaries and detailed constructions of *Block-DM* are presented in Sections 4 and 5, respectively. We demonstrate security properties in Section 6 and performance efficiency of *Block-DM* in Section 7. In Section 8, we conclude this paper.

2 Related Work

In terms of the system model of the data marketing, existing works can be roughly classified into two types: on-chain marketing and on/off-chain marketing.

In the on-chain model, the data are stored or shared directly via the blockchain [29], [31]. In this paradigm, the data are usually encrypted before being outsourced onto the blockchain. To achieve secure sharing of the data, the data owner not only defines access policies on the smart contract, but also manages the decryption of the data. Specifically, a data sharing framework is proposed in [31], where a service provider helps customers to manage the encrypted data and decryption key on the blockchain. In Calypso [29], a data owner can write an encrypted secret onto the blockchain for sharing, where the decryption management is delegated to a key-management committee. Gunasinghe *et al.* [30] utilize the blockchain to build a digital identity exchange framework among financial institutions. Specifically, a financial institution must obtain a user's consent for sharing the user's identity. The zero-knowledge succinct non-interactive argument of knowledge (zk-snark) technique [36] is used to prove the user's ownership of the identity asset on the blockchain, which often requires a trusted setup of the public parameters. The on-chain marketing model is suitable for sharing small data, such as a secret [29] or a digital identity asset [30]. For the large-size data, the on-chain model can incur expensive storage and computing costs for blockchain nodes.

In the on/off-chain model, an external (cloud) server is introduced for data marketing services [6]. Considering the GDPR requirements for the cloud-based data services [11], the blockchain can serve as a log system to manage general data operations on the cloud [5], [27], [37]. Such solutions usually require that the cloud server correctly record data operations onto the blockchain. For the data marketing operations, attribute-based encryption (ABE) technique can be utilized to achieve one-to-many data sharing in vehicular networks [34]. Specifically, the data owner outsources the encrypted data to the cloud server and uses the blockchain as a broadcast channel to publish access policies based on ABE, which makes the data access control transparent on the blockchain. The pioneering effort to build a blockchain-based data management with an on/off-chain model and GDPR-compliance is explored in [32]. The blockchain mainly serves as an access manager and log system of data operations, which requires a resource server (RS) to correctly manage the off-chain storage [32]. PrivySharing is a blockchain-based asset sharing scheme for smart cities [25]. It utilizes the membership service provider (MSP) in Hyperledger Fabric [16] to manage user identities and the channel-based data storage model to manage user data.

In terms of the GDPR requirements, most existing works enable consent-based data marketing: the data owners are informed of data requests (R2I) on the blockchain and can choose if the access is granted to the data requestor (R2C). Some existing works delegate the control capability of the data owner to an external committee [29] or an access structure on the contract [34]. For the identity privacy of the data owner, it is often required to have a certificate authority to provide certificate-based pseudonymity for IoT users. At the same time, a cloud server is required to correctly provide the storage management, where more research attention should be directed on addressing fairness issues in the on/off-chain model. In Table 1, compared with the existing works, *Block-DM* achieves the GDPR compliance (right to be informed, right to control, and strong identity privacy) with a distributed consortium. Under a practical but complex on/off-chain marketing model, *Block-DM* addresses the fairness challenges with the consideration of on/off-chain communications and various behaviors of marketing entities.

TABLE 1
Summary of Blockchain-Based Data Marketing

	System Model	R2I	R2C	Identity Privacy for Users	Identity Manager	Fairness
[31]	On-chain	✓	Delegated	Pseudonym	PKI	N/A
[29]	On-chain	✓	Delegated	Anonymity	Self-sovereign Identity	✓
[30]	On-chain	✓	✓	Pseudonym	CA	N/A
[25]	Hyperledger Fabric	✓	✓	N/A	MSP	N/A
[34]	On/off-chain	✓	Delegated	Pseudonym	CA	N/A
[32]	On/off-chain	✓	✓	Certificate	CA	N/A
Block-DM	On/off-chain	✓	✓	Conditional anonymity: distributed issuance & threshold opening	Consortium Committee	✓

** "on-chain" means digital assets for sharing are fully stored on the blockchain; "on/off-chain" means an external storage unit is used for storing digital assets; "R2I", right to be informed; "R2C", right to control; N/A, not applicable; CA, certificate authority; PKI, public key infrastructure; MSP, membership service provider.*

3 Problem Formulation

3.1 System Model

In *Block-DM*, there are four entities in the data marketing framework: Supervising Authority (SA), Data Subject (DS), Data Controller (DC), and Third Party (TP).

- SA comprises of a set of independent supervising nodes, such as regulatory authorities. SA is responsible for setting up the public parameters, maintaining a consortium blockchain architecture of the *Block-DM*, and assigning anonymous credential for DS.
- DS refers to users of IoT systems [38], e.g., an administrator of many IoT devices. DS can have more computing, storage, and communication resources than resource-constraint IoT devices but much less resources than a powerful cloud server. DS generates a large amount of data that are of great interest to TP for application developments. Instead of managing the data locally, DS relies on DC for data management and marketing services.
- TP is a third-party entity that is interested in the IoT data of DS. For example, TP can be a technology company that provides smart home solutions. By collecting runtime data from different IoT devices, TP can obtain design insights that may lead to future product developments.
- DC manages the data marketing between DS and TP. *Block-DM* breaks the role of DC into two parts: Cloud Server (CS) and BlockChain (BC). (1) CS is a storage and computing entity that provides data storage, transmission, and marketing services for DS and TP; (2) BC is a consortium ledger maintained by SA. SA enforces DS-consent-based data marketing over IoT data on a smart contract and records operations of $DS/CS/TP$ in the marketing process.

In Fig. 1, *Block-DM* consists of 5 phases. (1) System Setup: SA initializes a consortium blockchain network with smart contracts for data marketing. SA also sets system public parameters for issuing DS credentials and conducting data marketing; (2) DS Registration: DS registers him/herself at SA to obtain an anonymous credential. Note that the credential is issued by a set of supervising nodes collaboratively and is used for conducting data marketing; (3) Data Listing: DS constructs a data item including encrypted data, data description, data digest, and a set of data commitments. DS stores the data item on CS and uploads commitments of the data item onto the smart contract. CS verifies the correctness of the commitments and confirms the data item on the smart contract if the verifications pass; (4) Data Trading: TP retrieves the data item of interests from CS and corresponding commitments from BC. If the commitments match the data item, TP sends a data request to BC with its deposit. If DS approves the data request, DS sends a confirmation to the smart contract with a new proof. After that, TP retrieves the confirmation message, verifies the correctness of the retrieved data, and pays the fee to DS and CS if the retrieved data is correct; (5) Tracing. SA reveals the true identity of DS for the data marketing if necessary.

3.2 Security Model and Goals

CS is a multi-sector commercialized organization. Due to the profit consideration or lack of public transparency, CS may not always correctly conduct a data-marketing instance [11]. BC is a secure and distributed infrastructure, that provides immutable storage and automatic contract executions. TP is a rational party and would like to pay for the data marketing service if the correctness of the data item is guaranteed. DS is a rational IoT user that would like to trade the IoT data if marketing requirements and fair payments are achieved [13], [39]. SA consists of a set of secure supervising nodes.

In *Block-DM*, the following marketing requirements of DS should be achieved:

- Right to be informed: DS should be aware of: (1) data item (of the DS) to be traded; (2) receipt (TP) of the data item; (2) means of usage of the data item.

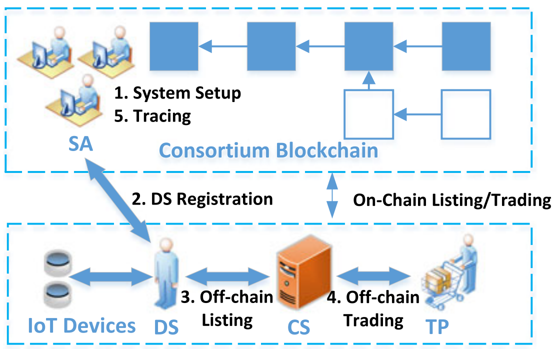


Fig. 1. Block-DM model.

TABLE 2
Abbreviations

IoT	Internet of Things
GDPR	General Data Protection Regulation
SA	Supervising Authority
DC	Data Controller
TP	Third Party
CS	Cloud Server
BC	Blockchain
ZKP	Zero-knowledge Proof
PS	Pointcheval-Sanders
$PVSS$	Publicly Verifiable Secret Sharing

TABLE 3
Notations

$\mathbb{G}$	Multiplicative groups
$\mathbb{Z}_p$	A ring of integers of a prime order p
λ	Security parameter
H	A hash function: $\{0,1\}^* \rightarrow \{0,1\}^{512}$
H'	A hash function: $\{0,1\}^* \rightarrow \mathbb{Z}_p$
H_0	A hash function: $\mathbb{Z}_p \rightarrow \mathbb{Z}_p \times \mathbb{G}_1$
H_1	A hash function: $\{0,1\}^* \rightarrow \mathbb{G}_1^n$
H_2	A hash function: $\mathbb{G}_1 \rightarrow \{0,1\}^{256}$
H_3	A hash function: $\{0,1\}^* \rightarrow \mathbb{G}_1$
$g, \tilde{g}$	Generators $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$

- Right to agree/reject: DS should have control of their data [1]. More superficially, DS should be able to agree or reject the sharing of the data over CS.
- Identity privacy: DS's real identity should be concealed in the data listing and trading. At the same time, identity privacy of DS can be removed if any commitment generated by DS is found incorrect.

Under the system and security models, *Block-DM* should achieve the following security goals:

- Consortium Management: (1) Right to be informed and right to agree/reject of DS should be achieved in a transparent and secure manner; (2) Identity privacy of DS should be preserved in a distributed manner for the data marketing process over the cloud.
- Executable Fairness: (1) DS only reveals the data to TP if DS receives fair payment; (2) TP only pays DS if TP receives the correct data. More specifically, "correct data" means that TP receives the same encrypted data as DS commits on BC; (3) Identity privacy of DS can be revealed by SA when necessary.

Block-DM aims at ensuring the above security goals for the data marketing process in Fig. 1 by designing a hybrid controller architecture with CS and BC. That is, we utilize a consortium blockchain network to control data marketing over the cloud and provide evidential records (commitments) for marketing operations for resolving later disputes. However, other data-marketing operations are not considered in *Block-DM*, such as arbitrary re-shares or usage of data after TP finishes a marketing instance. Moreover, we consider rational participants, who are motivated to behave correctly in the marketing process with financial incentives. Moreover, other attacks beyond the marketing process, such as intentional lock of funds, are not considered in this paper.

For identity privacy, we take the notion of *anonymity* from group signatures. That is, without the help of SA, there is no efficient adversary that can determine a true identity of DS (the identity information when DS registers itself at SA) from valid anonymous signatures generated by DS. Cross-layer linking attack or other side-channel leakages on DS identity are not considered in this paper, which is of independent research interests.

3.3 Design Goals

Block-DM should achieve the following design goals:

- Security: Security goals of *Block-DM* should be guaranteed, including consortium management and executable fairness.
- Efficiency: The additional overhead of the blockchain-based controller should be efficient. That is, on-chain computational and storage cost should be feasible compared with the traditional cloud-based data marketing.
- Implementation: *Block-DM* should be implemented and tested in a real-world consortium blockchain platform for comprehensive benchmarks.

4 Preliminaries

In this section, we present building blocks of *Block-DM*. Notations are shown in Table 3.

4.1 Cryptographic Notations

$\mathbb{G} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ is a set of multiplicative groups over elliptic curves. $\mathbb{G}$ is equipped with a prime order p and an asymmetric bilinear pairing $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. $\mathbb{Z}_p$ is a ring of integers with order p. λ is a security parameter and is often used implicitly. $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$, $H_0 : \{0,1\}^* \rightarrow \mathbb{Z}_p \times \mathbb{G}_1$, $H_1 : \{0,1\}^* \rightarrow \mathbb{G}_1^n$, and $H_2 : \mathbb{G}_1 \rightarrow \{0,1\}^{256}$ are collision-resist hash functions, which can be modeled as random oracles. $g \in \mathbb{G}_1$ and $\tilde{g} \in \mathbb{G}_2$ are two random generators. We use ˜ to indicate an element is chosen from $\mathbb{G}_2$.

4.2 ElGamal Encryption

ElGamal encryption [40] is a public-key encryption scheme that consists of three algorithms:

- $KeyGen(\mathbb{G}) \rightarrow (sk, pk)$. Choose a random number $sk \in \mathbb{Z}_p$ and compute $pk = g^{sk}$.
- $Enc(m, pk) \rightarrow c$. Take $m \in \mathbb{Z}_p$ as input. Choose a random number $r \in \mathbb{Z}_p$ and compute $c = (c_1, c_2) = (g^r, pk^r g^m)$.
- $Dec(c, sk) \rightarrow g^m$. Compute $c_2/c_1^{sk} = g^m$.

ElGamal encryption is secure in a group. *Block-DM* utilizes the ElGamal encryption to encrypt the secret shares of DS-chosen asymmetric identity key and symmetric keys for file encryptions.

4.3 Zero-Knowledge Proof (ZKP)

In ZKP [41], a prover can convince a verifier that the prover holds a secret satisfying a public relation without exposing the secret. ZKP has many important applications, such as digital signature and anonymous credentials [42]. In *Block-DM*, we consider the algebraic relations in the elliptic curve groups with the secrets only in the single discrete logarithm. For example, a siimple ZKP can be written as follows [43], [44]

$$\pi = ZKP\{(s) : y = g^s\}, \tag{1}$$

where $s \in \mathbb{Z}_p$ is a secret, $y, g \in \mathbb{G}_1^2$ are public parameters, and π is the generated proof. A non-interactive protocol for the above relation consists of two algorithms:

- *Prove*. The prover chooses a random number $r \in \mathbb{Z}_p$ for the secret s and computes $Y = g^r$, $c = H'(y, Y)$, $ck = r - c \times s$. The prover sends y, Y and ck to the verifier.
- *Verify*. The verifier computes $c = H'(y||Y)$ and checks $y^c g^{ck} \stackrel{?}{=} Y$.

For different relations (linear combinations or quadratic polynomials) with multiple secrets on the exponents of the generators, similar constructions can be realized by choosing different images (random numbers) for different secrets. ZKP is *complete* for honest verifiers; ZKP is *sound* that dishonest provers cannot forge a valid proof efficiently.

4.4 PS Signature

PS (Pointcheval-Sanders) signature [42] is a short signature scheme with a randomized verification mechanism. This property makes it suitable for constructing anonymous credentials, where proof of knowledge of signatures is required. In *Block-DM*, we adopt the multi-signer variation [45] of the original PS signature, denoted as MPS signature. Given the public parameters, the multi-signer scheme consists of the following algorithms:

- $MPS.KeyGen(\mathbb{G}, g, \tilde{g}) \rightarrow (\{sk_i, \widetilde{pk_i}\}_{i\in[n]}, \widetilde{vk})$
  The algorithm takes into the public parameters pp and generates private/public key pairs for n signers. For the signer indexed by $i \in [n]$, generate the private key $sk_i = (x_i, y_{i,1}, y_{i,2}) \in \mathbb{Z}_p^3$, and compute the public key $\widetilde{pk_i} = (\tilde{X}_i, \tilde{Y}_{i,1}, \tilde{Y}_{i,2}) = (\tilde{g}^{x_i}, \tilde{g}^{y_{i,1}}, \tilde{g}^{y_{i,2}})$. Compute $R = H_1(\widetilde{pk_1}, \widetilde{pk_2}, \ldots, \widetilde{pk_n}) = (r_1, r_2, \ldots, r_n) \in \mathbb{Z}_p^n$ and the aggregated verification key $\widetilde{vk}$ are as follows:

$$\widetilde{vk} = (\tilde{X}_A, \tilde{Y}_{A,1}, \tilde{Y}_{A,2}) = \left(\prod_{i=1}^n \tilde{X}_i^{r_i}, \prod_{i=1}^n \tilde{Y}_{i,1}^{r_i}, \prod_{i=1}^n \tilde{Y}_{i,2}^{r_i}\right). \tag{2}$$

- $MPS.Sign(pp, sk_i, m) \rightarrow \pi_i$
  The algorithm takes into a message $m \in \mathbb{Z}_p$ and outputs a signature π_i. Compute $H_0(m) \rightarrow (m', h)$ and $\pi_i = (m', h, \pi_{i,2})$, where $\pi_{i,2} = h^{x_i + y_{i,1}m + y_{i,2}m'}$. Note that, two messages are taken as inputs, which is slightly different from the original MPS signature.
- $MPS.Aggregate(\{\pi_i\}_{i\in[n]}, R) \rightarrow \pi_A$
  The algorithm first takes into the signatures of the message from n signers and checks that $\{\pi_i\}_{i\in[n]}$ are signatures on the same message. R can be computed similarly to the $KeyGen$ algorithm. Then, the algorithm computes $\pi_{A,2} = \prod_i^n \pi_{i,2}^{r_i} = h^{\sum_i^n x_i r_i + m \sum_i^n y_{i,1} r_i + m' \sum_i^n y_{i,2} r_i}$. The algorithm outputs an aggregated signature of the message m as

$$\pi_A = (m', h, \pi_{A,2}). \tag{3}$$

- $MPS.Verify(\pi_A, m, \widetilde{vk}) \rightarrow \{0, 1\}$
  The algorithm parses the π_A as $(m', h, \pi_{A,2})$. Compute $H_0(m) \rightarrow (m', h)$ and check consistency with π_A. Check $h \neq 1_{\mathbb{G}_1}$ and the following equation

$$e(h, \tilde{X}_A \tilde{Y}_{A,1}^m \tilde{Y}_{A,2}^{m'}) \stackrel{?}{=} e(\pi_{A,2}, \tilde{g}). \tag{4}$$

The MPS signature scheme can be utilized to generate anonymous credentials for DS. The supervising nodes act as the singers and collaboratively sign the blinded secret id of the DS. Later, the DS can aggregate the signatures as the anonymous credential. DS can prove validity of the credential by proving the knowledge of signature following the ZKP technique. Specifically, the prover demonstrates the knowledge of m, m' in Eq. (4), which is a ZKP protocol over elliptic groups.

4.5 Publicly Verifiable Secret Sharing (PVSS)

A (t, n) PVSS [46], [47] scheme enables a dealer holding a secret s to share the secret with n participants $(P_1, P_2, \ldots, P_n)$, where t-out-of-n participants can later combine the shares and recover the secret. To prevent a dishonest dealer to cheat on generating shares, PVSS scheme should support commitments of the shares and public verifiability of the committed values. Specifically, PVSS scheme consists of the following algorithms:

- $PVSS.Setup(\mathbb{G}, \tilde{g}_1) \rightarrow (\{psk_i, \widetilde{ppk_i}\}_{i\in[n]})$
  Each participant P_i chooses a random number $psk_i \in \mathbb{Z}_p$ as the private key and outputs a public key $\widetilde{ppk_i} = \tilde{g}_1^{psk_i}$. Note that, $(psk_i, \widetilde{ppk_i})$ is a private/public key pair of ElGamal encryption.
- $PVSS.Share(s, \{\widetilde{ppk_i}\}_{i\in[n]}, g, \tilde{g}_1, \tilde{g}_2) \rightarrow (\{E_i, \pi_{p_i}\}, H_s, \{A_j\})$
  The dealer picks random numbers $(a_1, \ldots, a_{t-1}) \in \mathbb{Z}_p^{t-1}$ and sets $a_0 = s$, where s is the secret to share. The dealer constructs a polynomial $P(x) = \sum_{j=0}^{t-1} a_j x^j$. For each $a_j, j \in [1, t-1]$, the dealer computes and publishes $A_j = g^{a_j}$. The dealer computes $H_s = g^s$. For each participant, the dealer chooses a random number $r_i \in \mathbb{Z}_p$ and computes

$$\begin{aligned} E_{i,1} &= \tilde{g}_1^{r_i}, E_{i,2} = \widetilde{ppk_i}^{r_i} \tilde{g}_2^{P(i)}, F_i = (E_{i,1}, E_{i,2}), \\ \pi_{p_i} &= ZKP\{(r_i) : E_{i,1} = \tilde{g}_1^{r_i} \wedge \\ &\quad e(g, E_{i,2}/\widetilde{ppk_i}^{r_i}) = e(H_s \prod_{j=1}^{t-1} A_j^{i^j}, \tilde{g}_2)\}. \end{aligned} \tag{5}$$

  The dealer broadcasts $(\{E_{i,1}, E_{i,2}, \pi_{p_i}\}, H_s, \{A_j\})$ to all participants.
- $PVSS.Verify(\pi_{p_i}, E_{i,1}, E_{i,2}) \rightarrow \{0, 1\}$
  Each participant P_i checks that the correctness of the proof π_{P_i} using the ZKP technique.
- $PVSS.Recover(\{psk_i, F_i\}_{i\in[t]}) \rightarrow \tilde{g}_2^s$
  Denote a set of t participants as SO_t. Each participant P_i in SO_t with the secret key psk_i computes

$$\pi_{F_i} = ZKP\{(psk_i) : E_i' = E_{i,2}/E_{i,1}^{psk_i} \wedge \widetilde{ppk_i} = \tilde{g}_1^{psk_i}\}. \tag{6}$$

  An opener collects t copies of $\{E_i', \pi_{F_i}\}$. If all π_{F_i} are correct, the opener computes the follows with t collected E_i'

$$\prod_{P_i \in SO_t} (E_i')^{\lambda_i} = \tilde{g}_2^s, \; \lambda_i = \prod_{P_j \in SO_t, j\neq i} j/(j-i). \tag{7}$$

PVSS scheme finally reconstructs the secret s over the generator $\tilde{g}_2$. This property is later utilized to construct the opening material for the anonymous credential from MPS signature.

4.6 Smart Contract

Smart contract [24] in a consortium blockchain stores codes on the blockchain storage and specifies the terms and actions for involved parties. (1) Involved parties can be blockchain nodes in the consortium. (2) Terms refer to the states and conditions encoded in the script language of the smart contract, e.g., Chaincode in Hyperledger Fabric. (3) Actions include read/write blockchain storage and transfer digital assets between blockchain nodes.

The involved parties can interact with the smart contract by sending transactions to the contract address. A confirmed transaction by the blockchain network changes the state of the contract. Benefiting from the transparency and transaction robustness of the blockchain network, the state change of the smart contract is trustworthy and publicly verifiable, which makes it suitable to enforce consortium management and executable fairness.

5 Blockchain-Cloud Transparent Data Marketing

5.1 Overview

For the consortium management of user identities, we enable the multi-signer aggregation form [45] of the PS signature [42] for credential generation. Moreover, publicly verifiable secret sharing (PVSS) scheme [47] is integrated to securely share the opening token to the committee members [45]. Moreover, *Block-DM* also enables DS to efficiently prove that a confirmation message is sent from the same user that creates the data item. For the executable fairness, we introduce the cloud server as an off-chain storage in *Block-DM*. Compared with a trading protocol [13] without CS, off-chain communications with CS in *Block-DM* are not publicly verifiable, which brings design challenges for the trading fairness. As a result, we let DS generate succinct commitments and efficient proofs for data items to be stored on-chain, that are only valid after a confirmation by CS in the data listing. Based on that, we design an on/off-chain trading protocol with financial incentives and effective verifications of marketing behavior. With a careful design of message flows in the trading, clear responsibility of marketing players can be determined in case of marketing disputes.

Block-DM consists of five phases: System Setup, DS Registration, Data Listing, Data Trading, and Tracing. In System Setup, SA initializes public system parameters. In Registration, DS obtains an anonymous credential from SA. Specifically, DS chooses a secret key and shares it with SA using PVSS. SA verifies the correctness of the shares, blindly signs the secret key, and sends the signatures to DS. DS aggregates the received signatures to obtain an anonymous credential using $MPS.Aggregate$. In Data Listing, DS processes data items, stores the encrypted data items on the CS, and lists the data descriptions on the blockchain. In Data Trading, DS, CS and TP conduct the data sharing via a data-marketing smart contract. In Tracing, the identity privacy of misbehaving DS is removed by SA. For illustrative simplicity, *Block-DM* makes the following assumptions:

- Secure and authenticated off-chain (synchronous boradcast) channels are set up between SA, CS, and TP. While secure and authenticated off-chain (synchronous broadcast) channels are set up between DS and SA, anonymous and secure communication channels are set up between DS and CS.
- A secure consortium blockchain network is set up and maintained by SA. Each supervising node, CS, and TP are assigned with a consortium blockchain membership by SA. That is, communications between CS, TP and BC are secure and authenticated.
- *Block-DM* abstracts a blockchain with the following functionalities: (1) It can receive and verify transactions from (anonymous) nodes under pre-defined conditions; (2) It provides global and reliable time stamps for transactions; (3) It has a marketing contract that can freeze/unfreeze funds or transfer funds when certain conditions are met, such as time-lock conditions; (4) It has an immutable storage that can check uniqueness of various on-chain identifiers, such as file, session, public key, and user identity.

5.2 System Setup

A set of n supervising nodes $SA = (S_1, S_2, \ldots, S_n)$ agree on system public parameters

$$pp = (\lambda, \mathbb{G}, e, p, g, \tilde{g}). \tag{8}$$

$\mathbb{G}$ is an elliptic group with an asymmetric paring e and a prime order p. g and $\tilde{g}$ are chosen securely and uniformly. SA generates public keys of each supervising node as follows:

$$\begin{aligned} &MPS.KeyGen(\mathbb{G}, g, \tilde{g}) \rightarrow (\{sk_i, \widetilde{pk_i}\}_{i\in[n]}, \widetilde{vk}),\\ &\widetilde{vk} = (\tilde{X}_A, \tilde{Y}_{A,1}, \tilde{Y}_{A,2}),\\ &PVSS.Setup(\mathbb{G}, \tilde{g}) \rightarrow \{psk_i, \widetilde{ppk_i}\}_{i\in[n]}. \end{aligned} \tag{9}$$

SA also generates and publishes a public key pk_s for a threshold ($\geq t$) ElGamal encryption using a distributed key generation protocol [48]. Note that, each individual supervising node must additionally prove possession of its secret key for $MPS.KeyGen$, $PVSS.Setup$ and the distributed encryption key with a ZKP, which is verified by all supervising nodes. Finally, each S_i obtains (sk_i, psk_i) and publishes $\widetilde{pk_i}$ and $\widetilde{ppk_i}$.

5.3 Registration

TP registers her/himself at SA with a public key pk_t for a digital signature scheme, e.g., ECDSA, and a blockchain membership mem_t. CS is authenticated by SA with a public key pk_c and a consortium blockchain membership mem_c. As shown in Fig. 2, DS registers her/himself at SA with a unique identity id_s as follows:

(1) Credential Request:

DS randomly chooses a secret key $sk_s \in \mathbb{Z}_p$ and computes

$$\begin{aligned} &H_0(id_s) = (m', h),\\ &PVSS.Share(sk_s, \{\tilde{ppk_i}\}_{i\in[n]}, h, \tilde{g}, \tilde{Y}_{A,1}), \end{aligned} \tag{9}$$

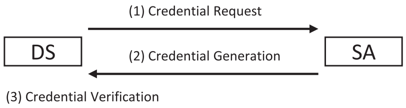


Fig. 2. Overview of registration.

where $\tilde{g}, \tilde{Y}_{A,1}$ are generators from public parameters and aggregated verification keys of MPS setup phase. Note that, $PVSS.Share$ is used to share the $\tilde{Y}_{A,1}^{sk_s}$ over supervising nodes, for user tracing. Particularly, DS computes a polynomial $P(x)$ with coefficient commitments $\{A_j\}$ and $H_s = h^{sk_s}$. For each supervising node S_i, DS chooses a random number $r_i \in \mathbb{Z}_p$ and computes shares on $\tilde{Y}_{A,1}^{sk_s}$ as follows:

$$F_i = (E_{i,1}, E_{i,2}) = (\tilde{g}^{r_i}, \widetilde{ppk}_i^{r_i} \tilde{Y}_{A,1}^{P(i)}), \tag{11}$$

where a proof π_{p_i} is also generated. DS broadcasts $\{A_j\}_{j \in t-1}$, H_s, id_s and $\{F_i, \pi_{p_i}\}_{i \in [1,n]}$ to all S_i. Note that, $g_{sk} = g^{sk_s}$ and a ZKP of sk_s in H_s and g_{sk} are also generated and broadcasted to all S_i, which is used for the security proof [45].

(2) Credential Generation:

Upon receiving $(\{A_j\}, H_s, id_s, \{F_i, \pi_{p_i}\})$, S_i first checks if id_s has existed. If not, S_i conducts

$$\begin{aligned} &(m', h) = H_0(id_s), \\ &PVSS.Verify(\pi_{p_i}, E_{i,1}, E_{i,2}). \end{aligned} \tag{12}$$

Note that the $PVSS.Verify$ uses $h, \tilde{g}, \tilde{Y}_{A,1}, \{A_j\}$, and $\widetilde{ppk}_i$ as generators. If the verification fails, S_i stops the credential issuance process. Otherwise, S_i computes

$$\pi_{i,2} = h^{x_i + y_{i,2} m'} H_s^{y_{i,1}}, \tag{13}$$

to obtain a blind signature $\pi_{i,2}$ on H_s, where $(x_i, y_{i,1}, y_{i,2})$ are secret keys of S_i. S_i sends $\pi_i = (m', h, \pi_{i,2})$ to DS via a secure channel.

(3) Credential Verification:

Upon receiving all π_i from SA, DS computes

$$\begin{aligned} &R = H_1(\widetilde{pk}_1, \widetilde{pk}_2, \ldots, \widetilde{pk}_n) = (r_1, r_2, \ldots, r_n), \\ &MPS.Aggregate(\{\pi_i\}_{i \in [n]}, R) \rightarrow \pi_A = (m', h, \pi_{A,2}), \end{aligned} \tag{14}$$

where $\widetilde{pk}_i$ is the public key of S_i. Then, DS verifies the correctness of π_A with $MPS.Verify(\pi_A, sk_s, \widetilde{vk})$, where $\widetilde{vk}$ is the aggregated verification key. Note that, here $H_0(id_s) \rightarrow (m', h)$, which is slightly different compared with $MPS.Verify$. If the verification passes, DS stores the anonymous credential π_A at the local storage. Otherwise, DS reports to SA that some signature is not correctly computed. SA can verify each individual signature to determine the misbehaving one.

5.4 Data Listing

In the following, we present the details of the data listing protocol as shown in Fig. 3.

(1) Data Item Generation:

DS chooses a file, $File$, for the data marketing, with a globally unique identifier id_f and a description D_F of keywords, price, and so on. DS chooses a key $K_s \in \mathbb{Z}_p$ and encrypts the file as $AES(File)_{H_2(g^{K_s})}$, e.g., using AES-256 encryption under the key $H_2(g^{K_s})$. DS chooses a random number $r_f \in \mathbb{Z}_p$ and encrypts K_s under the distributed public encryption key pk_s of SA as

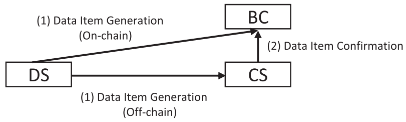


Fig. 3. Overview of data listing.

$$E_f = (E_{f,1}, E_{f,2}) = (g^{r_f}, pk_s^{r_f} g^{K_s}). \tag{15}$$

DS chooses another public/private key pair (pk_f, sk_f) of the ElGamal encryption for the file and sets a payment address $addr_f$. Note that, pk_f is for TP to encrypt a usage description of the file id_f. DS can have multiple payment addresses on the blockchain and the proof of possession of the secret key sk_f can be required in certain cases. DS computes

$$\begin{aligned} &\pi_f = ZKP\{(r_f, K_s) : E_{f,1} \wedge E_{f,2}\}, \\ &H_f = H(id_f || AES(File) || D_F). \end{aligned} \tag{16}$$

π_f is the ZKP to demonstrate the correct encryption of K_s [49] and H_f is the authentication code of the data item. DS chooses a linkage secret r_l and computes a linkage token $T_l = H_3(id_f)^{r_l}$. DS sets m_f as follows:

$$m_f = (id_f, H_f, E_f, \pi_f, pk_f, addr_f, T_l). \tag{17}$$

With his/her anonymous credential $\pi_A = (m', h, \pi_{A,2})$, DS chooses a random number $r \in \mathbb{Z}_p$ to generate an anonymous signature on m_f as follows:

$$\begin{aligned} &(\pi'_1, \pi'_2) = (h^r, \pi^r_{A,2}), \\ &\pi_{fs} = ZKP\{(sk_s, m', r_l) : T_l = H_3(id_f)^{r_l} \wedge \\ &e(\pi'_1, \tilde{X}_A) e(\pi'_1, \tilde{Y}_{A,1})^{sk_s} e(\pi'_1, \tilde{Y}_{A,2})^{m'} = e(\pi'_2, \tilde{g})\}. \end{aligned} \tag{18}$$

This is essentially a ZKP protocol for secrets sk_s, m' and r_l, and m_f should be included into the generation of a challenge c of the ZKP. DS sends m_f and π_{fs} to the marketing contract. The contract verifies the uniqueness of id_f and the correctness of π_f, π_{fs}. The contract stores m_f if all checks pass.

(2) Data Item Confirmation:

After m_f is accepted in the marketing contract, DS constructs $m_c = (id_f, AES(File), D_F)$ and a proof of knowledge of r_l in T_l for m_c [50], denoted as π_c. DS sends (m_c, π_c) to CS.

Upon receiving (m_c, π_c) from DS, CS retrieves m_f from BC. CS checks that m_f, m_c have the same id_f and $tL = H_3(id_f)^{r_l}$, and π_c is correct. CS aborts the message if they are not consistent. Otherwise, CS checks the following equation

$$H(id_f || AES(File) || D_F) \stackrel{?}{=} H_f \tag{19}$$

If the check passes, CS sends a confirmation to the marketing contract for the data item id_f. After the confirmation,

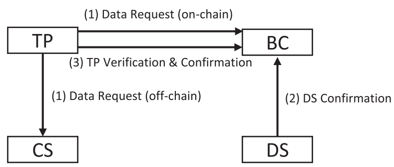


Fig. 4. Overview of data trading.

the data item id_f is valid for the data trading. If $H(id_f||AES(File)||D_F) \neq H_f$, CS sends m_c and π_c to SA. SA checks the consistency between m_f and m_c. If the consistency check passes, π_c is valid, but $H(id_f||AES(File)||D_F) \neq H_f$, SA reveals DS identity from π_{fs} and the data item id_f will be marked invalid. $H_f \in m_f$ is stored on the blockchain with the unique identifier id_f. Note that, CS is motivated to confirm valid data items since it can get payments from later data trading. Moreover, the linkage secret r_l should be kept secret by DS.

5.5 Data Trading

TP interacts with CS and BC for data trading over the confirmed data items as shown in Fig. 4:

(1) Data Request:

TP browses the storage of CS to look for files of interests by obtaining file descriptions D_F. Suppose that TP is interested in $File$ with id id_f, TP downloads $(id_f, AES(File), D_F)$ from CS and m_f with the same id_f from BC to check

$$H(id_f||AES(File)||D_F) \stackrel{?}{=} H_f \tag{20}$$

If the check passes, TP sends a file request to the marketing contract as follows:

$$(sid, id_f, id_t, pk_t, Enc(D_p, pk_f), C_t), \tag{21}$$

where sid is a unique session id, id_t is the ID of TP, pk_t is the public encryption key of TP, D_p is the data usage description, and $Enc(D_p, pk_f)$ is the ElGamal encryption under the public key pk_f of the file id_f. C_t is the predeposit at the contract for the payment of the file. The contract will check the validity of the message using the membership mem_t of TP where pk_t and id_t are certified by mem_t. We require that TP has proven the possession of the secret key for pk_t.

If the above check fails, TP sends the message received from CS to SA. Note that the communications between CS and TP are authenticated and secure. If SA determines the check fails (i.e., $H(id_f||AES(File)||D_F) \neq H_f$) where $H_f \in m_f$ is stored on the blockchain, SA concludes that CS did not correctly confirm the data item id_f and marks the data item as invalid on the marketing contract. Moreover, if TP keeps requiring encrypted files from CS but does not make requests on BC, CS can refuse to provide further services to TP.

(2) DS Confirmation:

Upon seeing the request on BC by the file id id_f, DS decrypts $Enc(D_p, pk_f)$ using sk_f. If either the decryption fails or DS disapproves the data usage in D_p, DS aborts. If DS approves the data usage, DS extracts the linkage secret r_l from its storage and computes

$$\begin{aligned} &g_c = H_3(sid||id_f||id_t), T'_l = g_c^{r_l}, \\ &E_{ft} = (E'_1, E'_2) = (g^{r'_f}, pk_t^{r'_f} g^{K_s}), \end{aligned} \tag{22}$$

where r'_f is a random number from $\mathbb{Z}_p$ and E_{ft} is an encryption of the key K_s under the TP's public key pk_t.

DS constructs a confirmation message $m_{con} = (sid, id_f, id_t, E_{ft}, g_c^{r_l})$ and generates a proof π_{ft} as follows:

$$\begin{aligned} &ZKP\{(r_f, r'_f, K_s, r_l) : E_1 = g^{r_f} \wedge E'_1 = g^{r'_f} \wedge \\ &E_2 = pk_s^{r_f} g^{K_s} \wedge E'_2 = pk_t^{r'_f} g^{K_s} \wedge T'_l = g_c^{r_l} \wedge T_l = H_3(id_f)^{r_l}\}. \end{aligned} \tag{23}$$

$E_f = (E_1, E_2)$ is the original encryption of K_s in m_f and $H_3(id_f)^{r_l}$ is the linkage token in m_f. π_{ft} serves as the evidence that DS includes the same decryption key in both E_f and E_{ft}. Moreover, due to the proof of knowledge of r_l, DS also links this response to the stored m_f on the blockchain storage. Note that, the above proof contains a hash for m_{con} when generating a challenge using ZKP. DS sends m_{con} and π_{ft} to the marketing contract. The contract checks the consistency between the data item, the file request and the DS confirmation message and the validity of π_{ft}. Note that, E_1, E_2 and, T_l are stored on the blockchain located by id_f. If all checks pass, the contract accepts the confirmation message.

(3) TP Verification & Confirmation:

If TP does not receive the confirmation from DS after a certain time when its file request is accepted on the contract, TP can cancel the data request and get the deposit back.

If TP receives a confirmation message from DS on the contract, TP can retrieve m_{con} to get E_{ft}. TP can get g^{K_s} by decrypting E_{ft} using sk_t, where sk_t is the private key of pk_t, and can decrypt the file $AES(File)$ using the key $H_2(g^{K_s})$. There are two cases:

(a) If the decrypted file matches the original description, TP can send (sid, id_f, id_t, pay) to the marketing contract. The contract will check the message sender is the same with that in the data request message and the (sid, id_f, id_t) matches. If the check passes, the contract transfers part of the deposit of TP to $addr$ of DS and the rest of the deposit to CS. Then, the data request with session id sid is finalized.

(b) TP can send a complain message to SA if the decrypted file content $File$ does not match the description D_F. SA checks the consistency between the complain message, data item, data request, and the DS confirmation. If all checks pass, it means that TP gets the encrypted file and decryption key as DS committed. In this case, SA decrypts E_f using their distributed ElGamal decryption keys [51]. SA assesses the file content based on the file description D_F. After the offline checks, each S_i sends a voting message to the marketing contract and a conclusion can be drawn from the majority of the voting messages. If the complain is valid, the contract transfers C_t back to TP, marks the data request as finalized, and marks the data item as invalid. Otherwise, the contract transfers deposit of TP to $addr$ of DS and CS, and marks the data request as finalized.

If DS sends a valid response to the contract but does not receive a confirmation or complain after a time period, SA or CS can send (sid, id_f, id_t, pay) to the contract. The contract checks the consistency between the data request and the received message, and if TP does not send a confirmation or complain timely based on the current block time and the block time when DS confirmation is recoded on the blockchain. If the check passes, the contract will transfer the deposit of TP to $addr$ of DS and CS, and marks the data request as finalized.

Remarks. (1) ZKP technique is used to prove the knowledge of anonymous credentials and the linkage token, which makes it computationally infeasible for an efficient adversary without DS secret key or the linkage token to forge DS

messages. Moreover, CS is motivated to conduct correct operations with incentives in the data marketing contract; (2) CS, TP and SA are assigned with valid membership to communicate with BC. Therefore, we omit the details of message authentications and consistency checks for CS, TP and SA in our trading protocol; (3) The ZKP proof π_{ft} is used to publicly prove that DS provides the same encrypted K_s in the data confirmation as that in the data listing. Alternatively, DS can only provide the encryption (E_{ft}) of K_s under pk_t with a proof. TP decrypts E_{ft} to get K_s and verifies the decryption of $AES(File)$, which changes the public verifiability of the key encryption to the local verifiability. Later, in case of disputes, TP can generate a verifiable decryption of E_f for SA's assessments.

Discussions. (1) We do not consider an assessment of file content by SA as the break of data confidentiality. However, to prevent TP from arbitrarily making complaints about the data content, SA can enforce countermeasures for failing to make correct complaints. An alternative solution that preserves partial data confidentiality against SA is to adopt the proof-of-misbehavior protocol in [13]. While it is efficient to assess the integrity of a file via circuits, it is a non-trial task to assess the content of a file in an efficient and privacy-reserving manner; (2) DS can periodically update data listings with new (short-lived) encryption keys and listing tokens. Moreover, side chains (private channels) can be implemented in the consortium blockchain, where on-chain data access can only be restricted to specific participants; (3) In the data trading, we require DS to generate anonymous transactions to a consortium blockchain. Similar designs can be found in Hyperledger Fabric [52]. For the DS confirmation transaction, we simplify the proof knowledge of anonymous credential with a simple linkage token. Our designs do require modifications of signature verifications in a consortium blockchain; (4) We use a linkage token and the ZKP to build linkage between later messages of DS and the first data listing message. The reason is that DS is not willing to share the linkage secret. An alternative solution is to require DS to prove knowledge of its secret key on a basename in each message [53]; (5) We adopt a key derivation function by encrypting the file encryption key with an ElGamal encryption. Advanced (but complex) key encryption and derivation functions can be used [54] for different types of files.

5.6 Tracing

Block-DM preserves the identity privacy of DS from valid anonymous signatures in the data listing and trading. However, DS may not always behave correctly especially if no accountability is pursued under the anonymity. In *Block-DM*, we consider the following two cases when the identity privacy of DS should be removed:

(a) CS detects that DS sends incorrect file and descriptions compared with the digest in m_f on the marketing contract. That is, $H_f \neq H(id_f||AES(File)||D_F)$, where $H_f \in m_f, (AES(File), D_F) \in m_c$. Note that, if TP later detects $H_f \neq H(id_f||AES(File)||D_F)$ in a confirmed data item, it is considered as the misconduct of CS. (b) TP detects that the content of the file does not match the decryption D_F. Note that there can be other cases where DS should be traced, which needs case-by-case decisions from SA. For example, π_{fs} is valid but π_f is not valid in the data listing.

The first case can be easily detected with the authenticated communication among CS and DS. Please refer to Section 5.4.(2) for details. CS can send the evidence of DS misbehavior to SA. SA verifies the validity of the evidence and randomly selects a set of supervising nodes to open the anonymous signature of DS. For the second case, the content assessment can be conducted by SA using the majority-voting method in the TP complain process.

If any of the above cases are confirmed, SA starts the tracing procedures with a valid anonymous signature $(\pi'_1, \pi'_2) \in \pi_{fs}$. A set of t supervising nodes, SO_t, conduct the following operations:

(1) For each DS with id_s and a secret share $(E_{j,1}, E_{j,2})$ for $S_j \in SO_t$, S_j computes O_{j,id_s} as follows:

$$O_{j,id_s} = e(\pi'_1, E_{j,2}/E_{j,1}^{psk_j}). \tag{24}$$

S_j sets $O_j = \{id_s, O_{j,id_s}\}$ for all registered DS, and broadcasts O_j to all other openers in SO_t via a secure broadcast channel.

(2) When receiving opening credentials from t supervising nodes, a supervising node in SO_t can open the signature (π'_1, π'_2). Specifically, the supervising node computes $\lambda_i = \prod_{S_j \in SO_t \setminus S_i} j/(j-i)$. Then, for each valid id_s in its storage, the supervising node checks the following equations until a match is found

$$H_0(id_s) = (m', h),$$
$$e(\pi'_1, \tilde{X}_A \tilde{Y}_{A,2}^{m'}) \prod_{S_j \in SO_t} O_{j,id_s}^{\lambda_j} \stackrel{?}{=} e(\pi'_2, \tilde{g}). \tag{25}$$

Finally, the true identity of the misbehaving user is recovered by opening nodes. Countermeasures, such as revocation of the user from the system, can be enforced.

6 Security Analysis

6.1 Distributed Anonymous Credential

We utilize the sign-randomize-prove paradigm of the group signature technique from multi-signer PS signature [42], [45], [55]. Specifically, a set of supervising nodes blindly sign on the DS-chosen g^{sk_s} to generate a credential π_A. To prove validity of the credential, DS first randomizes π_A, and then proves knowledge of sk_s, m' for the randomized π'_A. In the distributed setting, the security properties of such a paradigm can be categorized by the following three notions:

n-out-of-n Issuance. A credential π_A is valid iff n supervising nodes collaboratively sign on h^{sk_s}. The credential issuance process is secure and correct due to the following reasons: First, the signing keys of SA are generated individually and proven correct publicly. Unless the secret signing keys of the supervising nodes are leaked to an adversary, the adversary cannot efficiently forge a valid aggregated PS signature on sk_s [42], [45]. Second, after obtaining the signature on sk_s from n supervising nodes, DS can use the aggregated verification key $\tilde{vk}$ to ensure the correctness of the signature. Any individual signature can also be verified using the verification algorithm.

Anonymity. This property ensures that DS can prove valid credential in the data marketing without revealing its true identity. The property requires two folds: First, in the credential issuance phase, DS constructs a blinded element H_s of sk_s for SA to sign. A efficient cannot extract sk_s from H_s. Second, with a valid credential π_A, DS can prove sk_s using the ZKP technique. Particularly, DS chooses a random number r to randomize the original credential π_A. After the credential randomization, DS runs a non-interactive ZKP protocol to prove the knowledge of sk_s and m' following the verification of the aggregated MPS signature in Equ. 4. With the randomization and the ZKP, sk_s and m' are kept secret in the proof of knowledge of the signature. Therefore, the true identity information of a user from an anonymous signature cannot be recovered by an efficient adversary unless a tracing operation is conducted for the signature.

t-out-of-n Openness. This property ensures that any valid anonymous signature from a credential π_A can later be opened to a registered user when a threshold number of supervising nodes correctly compute O_j. The security of the threshold openness requires two folds. First, a verifiable secret share of sk_s is generated in the credential issuance phase, that are essentially an ElGamal encryption of the shares of $Y_{A,1}^{sk_s}$ over the MPS public verification key $Y_{A,1}$. The public shares are proven valid and consistent with H^{sk_s} via the ZKP technique. Due to the security of the PVSS scheme, the supervising nodes only sign on H_s if the shares are correctly computed. Second, an efficient adversary cannot forge a valid MPS signature [45]. Moreover, an efficient adversary cannot forge a valid anonymous signature that opens to a registered user without knowing the corresponding secret key [45]. Once conditions for opening the credential are met, *Block-DM* guarantees that a valid signature can be opened if at least *t-out-of-n* supervising nodes correctly decrypt the encrypted shares in Equ. 24 and the opening authority correctly computes Equ. 25, which is correct from the Shamir's secret sharing technique [46].

6.2 Blockchain Security

Block-DM utilizes the consortium blockchain as a secure provenance ledger of the data marketing process on CS. That is, the blockchain controls the state transitions of the data marketing contract with the following properties [16], [24]: (1) Distributed consensus. The supervising nodes who maintain the blockchain can reach a consistent view of ledger states; (2) Trusted state transition: State transitions of the data marketing contract should be publicly verifiable and a valid state change can be efficiently (timely) updated on the blockchain.

Distributed consensus is achieved with the secure consensus protocols of the consortium blockchain [56]. State transitions of the contract are conducted by sending and verifying the transactions of contract calls. Due to the distributed consensus, the function calls are verified by the supervising nodes. Once valid states are confirmed and stored on the blockchain, it cannot be maliciously changed since the state history is securely chained together via hash functions.

6.3 Consortium Management

The "right to be informed" and "right to agree/object" of DS over the cloud-based data marketing requires three folds. First, DS encrypts data and uploading corresponding information m_f to BC before sending them to CS. Due to the security of the ElGamal encryption, a party without the decryption key cannot decrypt the data, which gives DS the control over the data access. However, re-shares of the data by TP after obtaining access is out of scope of this paper. Second, the data requests of TP are notified to DS by the marketing contract for confirmations in a transparent manner. DS can review the purpose of data usage and only grant access (decryption key) to TP if DS approves the data usage. Third, DS proves knowledge of sk_s and r_l when listing the data. Then, DS includes the session id, file id, TP identity when proving knowledge of r_l in the data trading. This ensures that the confirmation and data listing messages are from the same DS and the confirmation message is intended for a specific data request unless r_l is leaked.

Identity privacy of DS is achieved from the security of the distributed anonymous credential as discussed in Section 6.1.

6.4 Executable Fairness

There are three fairness requirements in Section 3.2.

For the first requirement, the designed marketing protocol requires TP to make deposit when sending the file request to the contract and the deposit are frozen before DS sends an encrypted decryption key to TP [13]. Subsequently, when TP sends a positive confirmation, the deposit is transferred to CS and DS. In case of a purposely delay of a payment confirmation from TP, the marketing contract can transfer TP's deposit to CS and DS after a certain time when TP receives a confirmation message from DS but does not respond. TP cannot deny receiving the confirmation message since the communications happen on the blockchain.

For the second requirement, it is important to verify the 'correctness' of the received data by TP. As we introduce the CS as a data management unit, our design rationales are as follows:

(1) DS generates 'commitments' for data items to be stored on BC. The proof π_f is the commitment of the decryption key and the hash H_f is the digest of a file identifier, an encrypted file and a file description. The proof π_{fs} is to ensure that the data item listed on the blockchain is from an anonymous but authenticated DS. The proof π_{ft} is to build an efficient linkage between the listed data item and a request confirmation from the same DS.

(2) We require TP to first finish the off-chain communications with CS. More specifically, TP retries m_c from CS and verifies the correctness of H_f. After that, TP can make data requests, receive request confirmations, and make payments on the blockchain, which are authenticated and undeniable communications. If TP does not receive the confirmation from the data owner after a certain time when TP makes the request, TP can cancel the request and get back the deposit.

(3) The proofs H_f, π_f, π_{fs}, and π_{ft} are publicly verifiable to ensure that TP receives the exact data and decryption keys committed by DS. However, if TP finds that the decrypted content does not match the file description, an investigation process can be conducted by SA.

For the third requirement of the fairness, if DS misconduct is identified as discussed in Section 5.6, the true identity of DS can be recovered from the anonymous signature due to the *t-out-of-n openness* of the distributed credential.

TABLE 4
Computational Complexity

	Data Listing	Data Trading
DS	$11E_1 + 2E_t + 2P + AES_e$	$12E_1$
TP		$4E_1 + AES_d$
CS	$2E_1$	
BC	$7E_1 + 4E_t + 4P$	$14E_1$

7 Performance Evaluation

7.1 Complexity Analysis

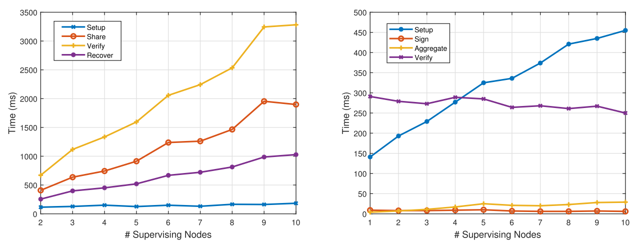
In the following, we give numerical analysis and comparisons between *Block-DM* and the existing works in Table 1.

In the on-chain data marketing model [29], [30], [31], data owners directly store large files on the blockchain. Therefore, the average on-chain storage cost for each file is $n_F * |F|_a$, where the number of full nodes in the blockchain is n_F and the average size of a file is $|F|_a$. In *Block-DM*, large files are stored off-chain with a succinct hash H_f stored on the blockchain, which results in a total $n_F * |H_f| + |F|_a$ on/off-chain storage cost. In practice, n_F can be 100 in a consortium blockchain, $|F|_a$ can be 20 MB, and H_f is only 512 bit. That is, the storage cost in *Block-DM* is significantly reduced compared with that in the on-chain storage model.

In the on/off-chain data marketing model, *Block-DM* further considers the marketing fairness issue without assuming an honest cloud server. The complexity analysis of *Block-DM* is shown in Table 4. We consider cryptographic operations in the marketing process and omit all hash calculations. $E_1/E_2/E_t$ are exponentiation operations in $\mathbb{G}_1/\mathbb{G}_2/\mathbb{G}_T$, P is a paring operation, and AES_e/AES_d is the AES encryption/decryption. From the table, we can see that *Block-DM* incurs succinct overheads for marketing participants. From our experiments in the following sections, we notice that on-chain response time to confirm a transaction is much more expensive than off-chain computation time. Compared with existing works, *Block-DM* requires only 2 transactions for honest participants to finish an instance of data marketing.

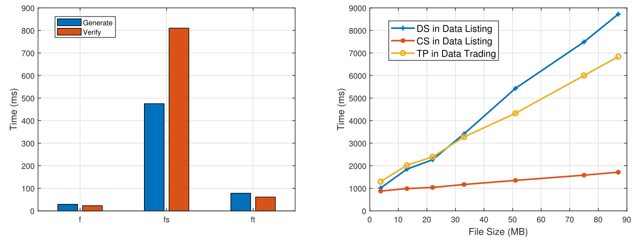
7.2 Off-Chain Overheads

We implement and test the computation overheads for the off-chain operations on a laptop with a 2.30 GHz processor and 8 GB memories. We use the Java Pairing Based Cryptography (JPBC) with type F pairing (BN curve) for cryptographic operations. We set the threshold t to be equal to the number of supervising nodes n and test the computation time by changing n from 2 to 10. We omit the cost of hash to point and hash of messages in generating ZKP.



(a) Publicly Verifiable Secret Sharing (b) Multi-message Signature

Fig. 5. Preliminary overheads.



(a) Proof Generation/Verification (b) Data Listing/Trading

Fig. 6. Off-chain computational overheads.

We adopt a modular strategy to analyze the off-chain overheads. In the registration, DS needs to verifiably share a self-chosen secret with supervising nodes. As a result, the computational cost for DS linearly increases with the number of supervising nodes in Fig. 5a. For the credential verification, it costs around 300 ms in total (aggregation of all signatures and verification of the aggregated signature) as shown in Fig. 5b. For SA, the verification cost of all DS shares linearly increases as shown in Fig. 5a. At the same time, the computation time for a single supervising node to generate a signature remains very low as shown in Fig. 5b.

For the data listing and trading, the core cryptographic overheads are the generation and verification of three proofs π_f, π_{fs}, and π_{ft}, which are shown in Fig. 6a. By comparisons, π_{fs} is more expensive due to the use of pairing operations. However, since the data marketing is not a time-sensitive application, a certain amount of delay can be tolerated.

We further test the hash and encryption/decryption operations on files with different sizes. We use the JAVA native implementation of SHA-512 hash function and AES-256 encryption[1]. Based on the previous results in the experiments, we measure the main computation costs for CS (verifications of hash of the file, π_f and π_{fs}) and DS (generations of an AES key, π_f and π_{fs}, compute a hash and an AES encryption of the file) in the data listing and for TP (verifications of hash of the file and (optional) proofs $\pi_f, \pi_{fs}, \pi_{ft}$, and decryption of the file) in the data trading. In Fig. 6b, the computation overheads linearly increase with the size of the files. In our experiments, the benchmark for the AES-256 on JAVA SE 1.8 is 12 MB/s for the encryption and 16.8 MB/s for the decryption. For a larger file, it can be divided into different file blocks. Specifically, an 1GB file requires roughly 85 seconds for encryption and 61 seconds for the decryption.

7.3 On-Chain Overheads

For on-chain overheads, since we only upload succinct file digests and proofs onto the blockchain, the mainly expensive on-chain operation is the verification of proofs π_f, π_{fs}, and π_{ft}. Therefore, we thoroughly implement and test the verifications of the three proofs on a real-world consortium blockchain network. We set up a Hyperledger Fabric [16] blockchain network on the same laptop, which consists of an orderer running the Raft consensus protocol and a few peer nodes. To support crypto operations in the JAVA-based chaincode, we include JPBC [57] into the chaincode dependencies. To ensure all peers instantiate the same elliptic groups, we include the same curve parameters into the JPBC

1. https://github.com/mkyong/core-java/tree/master/java-crypto

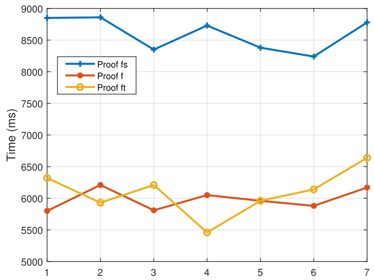


Fig. 7. ChainCode overheads.

jar files. For other parameters, such as generators and verification keys, we directly code them as byte arrays in the chaincode.

The verification of π_f, π_{fs} and π_{ft} is written as a chaincode function. All proofs are generated off-chain and sent to the function for on-chain verifications. All peer nodes in the blockchain network install and approve the contract package and commit the package in the same channel. More specifically, we use the "peer chaincode invoke" to call the verification function and measure the time difference between sending the function call and receiving a response from the blockchain. We run experiments multiple times with the same blockchain network configurations. In Fig. 7, it takes from 5 to 9 seconds to receive a response for a function call. The most expensive verification of π_{fs} takes around 8 seconds. However, compared with the running time shown in Fig. 6a, the response time is mainly decided by the blockchain consensus protocol and network status.

8 Conclusion

In this paper, we have proposed a transparent data marketing architecture with the cloud as a data management unit and the consortium blockchain as a reliable controller. *Block-DM* has achieved the consortium management and the executable fairness in the cloud-based marketing model with a distributed committee. Thorough security analysis and experimental results have shown that *Block-DM* is both secure and practical. The comprehensive design and evaluation of *Block-DM* under privacy requirements may shed light on further research of regulation-compliant data management. In the future, we will explore verifiable data processing at the cloud under privacy regulations.

Acknowledgments

The authors would like to thank Yuxiang Zhang for testing JPBC in the consortium blockchain.

References

[1] R. Herian, "Blockchain, GDPR, and fantasies of data sovereignty," *Law, Innov. Technol.*, vol. 12, no. 1, pp. 156–174, 2020.
[2] X. Shen *et al.*, "Data management for future wireless networks: Architecture, privacy preservation, and regulation," *IEEE Netw.*, vol. 35, no. 1, pp. 8–15, Jan./Feb. 2021.
[3] Q. Zhang, L. T. Yang, and Z. Chen, "Privacy preserving deep computation model on cloud for big data feature learning," *IEEE Transactions on Computers*, vol. 65, no. 5, pp. 1351–1362, May 2016.
[4] O. O. Malomo, D. B. Rawat, andM. Garuba, "Next-generation cybersecurity through a blockchain-enabled federated cloud framework," *The J. Supercomput.*, vol. 74, no. 10, pp. 5099–5126, 2018.
[5] L. Zhu, Y. Wu, K. Gai, and K.-K. R. Choo, "Controllable and trustworthy blockchain-based cloud data management," *Future Gener. Comput. Syst.*, vol. 91, pp. 527–535, 2019.
[6] X. Zheng, R. R. Mukkamala, R. Vatrapu, and J. Ordieres-Mere, "Blockchain-based personal health data sharing system using cloud storage," in *Proc. IEEE 20th Int. Conf. e-Health Netw. Appl. Serv.*, 2018, pp. 1–6.
[7] E. Fernandes, A. Rahmati, J. Jung, and A. Prakash, "Decentralized action integrity for trigger-action IoT platforms," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018.
[8] J. Liang, Z. Qin, J. Ni, X. Lin, and X. Shen, "Practical and secure SVM classification for cloud-based remote clinical decision services," *IEEE Trans. Comput.*, vol. 70, no. 10, pp. 1612–1625, Oct. 2021.
[9] A. Sonnino, M. Al-Bassam, S. Bano, S. Meiklejohn, and G. Danezis, "Coconut: Threshold issuance selective disclosure credentials with applications to distributed ledgers," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2018.
[10] General Data Protection Regulation (GDPR), 2016. Accessed: Apr. 2020. [Online]. Available: https://gdpr-info.eu
[11] S. Shastri, M. Wasserman, and V. Chidambaram, "GDPR anti-patterns: How design and operation of modern cloud-scale systems conflict with GDPR," 2019, *arXiv:1911.00498*.
[12] T. Urban, D. Tatang, M. Degeling, T. Holz, and N. Pohlmann, "Measuring the impact of the GDPR on data sharing in AD networks," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.*, 2020, pp. 222–235.
[13] S. Dziembowski, L. Eckey, and S. Faust, "FairSwap: How to fairly exchange digital goods," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2018, pp. 967–984.
[14] D. Liu, A. Alahmadi, J. Ni, X. Lin, and X. Shen, "Anonymous reputation system for IIoT-enabled retail marketing atop pos blockchain," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3527–3537, Jun. 2019.
[15] M. Li, J. Weng, J.-N. Liu, X. Lin, and C. Obimbo, "BB-VDF: Enabling accountability and fine-grained access control for vehicular digital forensics through blockchain," Cryptology ePrint Archive, Rep. 2020/011, 2020, [Online]. Available: https://eprint.iacr.org/2020/011.
[16] E. Androulaki *et al.*, "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. 13th EuroSys Conf.*, 2018, pp. 1–15.
[17] C. Li, Y. Fu, F. R. Yu, T. H. Luan, and Y. Zhang, "Vehicle position correction: A vehicular blockchain networks-based GPS error sharing framework," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 898–912, Feb. 2021.
[18] Z. Su, Y. Wang, Q. Xu, and N. Zhang, "LVBS: Lightweight vehicular blockchain for secure data sharing in disaster rescue," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 19–32, Jan.–Feb. 2022.
[19] X. Liu, S. X. Sun, and G. Huang, "Decentralized services computing paradigm for blockchain-based data governance: Programmability, interoperability, and intelligence," *IEEE Trans. Services Comput.*, vol. 13, no. 2, pp. 343–355, Mar./Apr. 2020.
[20] Y. Hu, S. Kumar, and R. A. Popa, "Ghostor: Toward a secure data-sharing system from decentralized trust," in *Proc. 17th USENIX Symp. Netw. Syst. Des. Implementation*, 2020, pp. 851–877.
[21] V. Koutsos, D. Papadopoulos, D. Chatzopoulos, S. Tarkoma, and P. Hui, "Agora: A privacy-aware data marketplace," in *Proc. 40th Int. Conf. Distrib. Comput. Syst.*, 2020, pp. 1211–1212.
[22] W. Dai, C. Dai, K.-K. R. Choo, C. Cui, D. Zou, and H. Jin, "SDTE: A secure blockchain-based data trading ecosystem," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 725–737, 2020.
[23] Y. Xiao, N. Zhang, J. Li, W. Lou, and Y. T. Hou, "PrivacyGuard: Enforcing private data usage control with blockchain and attested off-chain contract execution," in *Proc. Eur. Symp. Res. Comput. Secur.*, 2020, pp. 610–629.
[24] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger byzantium version," *Ethereum Project Yellow Paper*, pp. 1–39, 2018.
[25] I. Makhdoom, I. Zhou, M. Abolhasan, J. Lipman, and W. Ni, "PrivySharing: A blockchain-based framework for privacy-preserving and secure data sharing in smart cities," *Comput. Secur.*, vol. 88, 2020, Art. no. 101653.
[26] D. Liu, J. Ni, C. Huang, X. Lin, and X. Shen, "Secure and efficient distributed network provenance for IoT: A blockchain-based approach," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7564–7574, Aug. 2020.

[27] M. Barati and O. Rana, "Tracking GDPR compliance in cloud-based service delivery," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2020.2999559.

[28] K. Nguyen, G. Ghinita, M. Naveed, and C. Shahabi, "A privacy-preserving, accountable and spam-resilient geo-marketplace," in *Proc. 27th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2019, pp. 299–308.

[29] E. Kokoris-Kogias, E. C. Alp, L. Gasser, P. Jovanovic, E. Syta, and B. Ford, "CALYPSO: Private data management for decentralized ledgers," *Proc. Conf. Very Large Data Bases Endowment*, 2020, pp. 586–599.

[30] H. Gunasinghe *et al.*, "PrividEx: Privacy preserving and secure exchange of digital identity assets," in *Proc. World Wide Web Conf.*, 2019, pp. 594–604.

[31] K. Bhaskaran *et al.*, "Double-blind consent-driven data sharing on blockchain," in *Proc. IEEE Int. Conf. Cloud Eng.*, 2018, pp. 385–391.

[32] N. B. Truong, K. Sun, G. M. Lee, and Y. Guo, "GDPR-compliant personal data management: A blockchain-based solution," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 1746–1761, 2020.

[33] M. S. Rahman, A. Al Omar, M. Z. A. Bhuiyan, A. Basu, S. Kiyomoto, and G. Wang, "Accountable cross-border data sharing using blockchain under relaxed trust assumption," *IEEE Trans. Eng. Manag.*, vol. 67, no. 4, pp. 1476–1486, Nov. 2020.

[34] K. Fan *et al.*, "A secure and verifiable data sharing scheme based on blockchain in vehicular social networks," *IEEE Trans. Veh. Technol*, vol. 69, no. 6, pp. 5826–5835, Jun. 2020.

[35] D. Francati *et al.*, "Audita: A blockchain-based auditing framework for off-chain storage," in *Proc. 9th Int. Workshop Secur. Blockchain Cloud Comput.*, 2021, pp. 5–10.

[36] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *Proc. IEEE Symp. Secur. Privacy*, 2013, pp. 238–252.

[37] B.-K. Zheng *et al.*, "Scalable and privacy-preserving data sharing based on blockchain," *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 557–567, 2018.

[38] T. Linden, R. Khandelwal, H. Harkous, and K. Fawaz, "The privacy policy landscape after the GDPR," in *Proc. Privacy Enhancing Technol.*, 2020, pp. 47–64.

[39] C. Lin, D. He, X. Huang, and K.-K. R. Choo, "OBFP: Optimized blockchain-based fair payment for outsourcing computations in cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 3241–3253, 2021.

[40] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. 31, no. 4, pp. 469–472, Jul. 1985.

[41] M. Bellare and O. Goldreich, "On defining proofs of knowledge," in *Proc. Annu. Int. Cryptol. Conf.*, 1992, pp. 390–420.

[42] D. Pointcheval and O. Sanders, "Reassessing security of randomizable signatures," in *Proc. Cryptographers Track RSA Conf.*, 2018, pp. 319–338.

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

[44] T. P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proc. Annu. Int. Cryptol. Con.*, 1991, pp. 129–140.

[45] J. Camenisch, M. Drijvers, A. Lehmann, G. Neven, and P. Towa, "Short threshold dynamic group signatures," in *Proc. Int. Conf. Secur. Cryptogr. Netw.*, 2020, pp. 401–423.

[46] B. Schoenmakers, "A simple publicly verifiable secret sharing scheme and its application to electronic voting," in *Proc. Annu. Int. Cryptol. Conf.*, 1999, pp. 148–164.

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

[48] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, "Secure distributed key generation for discrete-log based cryptosystems," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 1999, pp. 295–310.

[49] C. P. Schnorr and M. Jakobsson, "Security of signed ELGamal encryption," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2000, pp. 73–89.

[50] C.-P. Schnorr, "Efficient identification and signatures for smart cards," in *Proc. Conf. Theory Appl. Cryptol.*, 1989, pp. 239–252.

[51] P.-A. Fouque and D. Pointcheval, "Threshold cryptosystems secure against chosen-ciphertext attacks," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, 2001, pp. 351–368.

[52] D. Bogatov, A. De Caro, K. Elkhiyaoui, and B. Tackmann, "Anonymous transactions with revocation and auditing in hyperledger fabric," *IACR Cryptol. ePrint Arch.*, vol. 2019, 2019, Art. no. 1097.

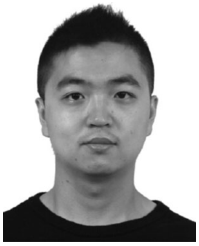
[53] J. Camenisch, M. Drijvers, and A. Lehmann, "Anonymous attestation using the strong diffie hellman assumption revisited," in *Proc. Int. Conf. Trust Trustworthy Comput.*, 2016, pp. 1–20.

[54] H. Krawczyk, "Cryptographic extraction and key derivation: The HKDF scheme," in *Proc. Annu. Cryptol. Conf.*, 2010, pp. 631–648.

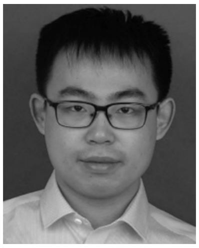
[55] D. Pointcheval and O. Sanders, "Short randomizable signatures," in *Proc. Cryptographers Track RSA Conf.*, 2016, pp. 111–126.

[56] M. Vukolić, "The quest for scalable blockchain fabric: Proof-of-work versus BFT replication," in *Proc. Int. Workshop Open Problems Netw. Secur.*, 2015, pp. 112–125.

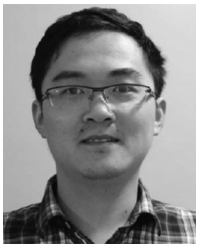
[57] A. De Caro and V. Iovino, "jPBC: Java pairing based cryptography," in *Proc. IEEE Symp. Comput. Commun.*, 2011, pp. 850–855.



Dongxiao Liu (Member, IEEE) received the PhD degree from the Department of Electrical and Computer Engineering, University of Waterloo, Canada, in 2020. He is currently a postdoctoral research fellow with the Department of Electrical and Computer Engineering, University of Waterloo. His research interests include security and privacy in intelligent transportation systems, blockchain, and mobile networks.



Cheng Huang (Member, IEEE) received the BEng and MEng degrees in information security from Xidian University, China, in 2013 and 2016, respectively, and the PhD degree in electrical and computer engineering from the University of Waterloo, ON, Canada, in 2020. He was a project officer with the INFINITUS Laboratory, School of Electrical and Electronic Engineering, Nanyang Technological University till July 2016. His research interests include applied cryptography, cyber security, and privacy in the mobile network.



Jianbing Ni (Member, IEEE) received the BE and MS degrees from the University of Electronic Science and Technology of China, Chengdu, China, in 2011 and 2014, respectively, and the PhD degree in electrical and computer engineering from the University of Waterloo, Waterloo, Canada, in 2018. He is currently an assistant professor with the Department of Electrical and Computer Engineering and Ingenuity Labs Research Institute, Queen's University, Kingston, ON, Canada. His current research interests include applied cryptography and network security, with a focus on cloud computing, smart grid, mobile crowdsensing, and the Internet of Things.

Xiaodong Lin (Fellow, IEEE) received the PhD degree in information engineering from the Beijing University of Posts and Telecommunications, China, and the PhD degree (with Outstanding Achievement in Graduate Studies Award) in electrical and computer engineering from the University of Waterloo, Canada. He is currently a professor with the School of Computer Science, University of Guelph, Canada. His research interests include computer and network security, privacy protection, applied cryptography, computer forensics, and software security.

Xuemin Sherman Shen (Fellow, IEEE) received the PhD degree in electrical engineering from Rutgers University, New Brunswick, NJ, USA, in 1990. He is currently a university professor with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. His research interests include network resource management, wireless network security, Internet of Things, 5G and beyond, and vehicular ad hoc and sensor networks. Dr. Shen is a registered professional engineer of Ontario, Canada, fellow of the Engineering Institute of Canada, fellow of the Canadian Academy of Engineering, fellow of the Royal Society of Canada, foreign member of the Chinese Academy of Engineering, and distinguished lecturer of IEEE Vehicular Technology Society and Communications Society. He was the recipient of Canadian Award for Telecommunications Research from the Canadian Society of Information Theory (CSIT) in 2021, R.A. Fessenden Award in 2019 from IEEE, Canada, Award of Merit from the Federation of Chinese Canadian Professionals (Ontario) in 2019, James Evans Avant Garde Award in 2018 from the IEEE Vehicular Technology Society, Joseph LoCicero Award in 2015 and Education Award in 2017 from the IEEE Communications Society, and Technical Recognition Award from Wireless Communications Technical Committee (2019) and AHSN Technical Committee (2013). He was also the recipient of Excellent Graduate Supervision Award in 2006 from the University of Waterloo and Premier's Research Excellence Award (PREA) in 2003 from the Province of Ontario, Canada. He was the technical program committee chair/co-chair of IEEE Globecom' 16, IEEE Infocom'14, IEEE VTC'10 Fall, IEEE Globecom'07, and the chair of IEEE Communications Society Technical Committee on Wireless Communications. He is the president of IEEE Communications Society. He was the vice president of Technical and Educational Activities, vice president for Publications, Member-at-Large on the Board of Governors, chair of the Distinguished Lecturer Selection Committee, a member of IEEE Fellow Selection Committee of the ComSoc. He was the editor-in-chief of the *IEEE Internet of Things Journal*, *IEEE Network*, and *IET Communications*.