Blockchain-Based Reliable and Privacy-Aware Crowdsourcing with Truth and Fairness Assurance

Haiqin Wu, *Member, IEEE,* Boris Düdder, *Member, IEEE,* Liangmin Wang, Shipu Sun, and Guoliang Xue, *Fellow, IEEE*

***Abstract*—The ubiquity of crowdsourcing has reshaped the static sensor-enabled data sensing paradigm with cost efficiency and flexibility. Still, most existing triangular crowdsourcing systems only work under the centralized trust assumption and suffer from various attacks mounted by malicious users. Although incorporating the emerging blockchain technology into crowdsourcing provides a possibility to mitigate some of the issues, how to concretely implement the crucial components and their functionalities in a verifiable and privacy-aware manner remains unaddressed. In this paper, we present BRPC, a blockchain-based decentralized system for general crowdsourcing. BRPC integrates the confident-aware truth discovery algorithm to provide task requesters with reliable task truths while evaluating each worker's data quality. To mitigate biased evaluation of malicious requesters, we propose a privacy-aware verification protocol leveraging the Threshold Paillier Cryptosystem, with which a certain number of workers can collaboratively verify the evaluation results without knowing any sensory data. Furthermore, we define the three roles of a user and elaborate a comprehensive reputation evaluation model enforced by smart contracts for its trustworthy running. Financial and social incentives are both offered to motivate users' honest participation. Finally, we implement a prototype of BRPC and deploy it on the Ethereum blockchain. Theoretical analyses and experiment results show its security and practicality.**

***Index Terms*—Crowdsourcing, blockchain, truth discovery, verifiability, privacy.**

I. Introduction

THE past few years have witnessed the rapid development and huge commercial value of crowdsourcing in the Internet of Things (IoT) era. As a significant part of sharing economy, crowdsourcing has become a leading paradigm which leverages the *crowds*, *e.g.*, smartphone users, to solve some tough problems, such as large-scale data sensing/collection in IoT and more specialized services. After successful completion, the requesters get their tasks done. Meanwhile, the crowds (*i.e.,* workers) can earn rewards as compensation for their work. Apparently, crowdsourcing brings a win-win situation for both task requesters and workers. Due to these remarkable benefits, numerous crowdsourcing applications have emerged and gained considerable attention in different fields such as environment monitoring, healthcare assistance, transportation, and business.

Traditionally, the crowdsourcing model involves three roles: task requesters, workers, and a centralized crowdsourcing platform, where the requesters can post tasks and the workers are able to search available tasks and work on them. Such a centralized crowdsourcing model is easy for system management and has been prevalent in existing crowdsourcing applications. However, due to its trust-based centralization, many issues [1] cannot be circumvented, including single point of failure, vulnerable to Sybil attacks, DDoS attacks, data privacy disclosure [2], and the notorious "false-reporting" and "free-riding" behaviors [3]. Although cryptography-based privacy preservation ([4]–[7]) and anonymous reputation management model [8], [9] were proposed to solve part of these problems, they are still limited to the centralized setting and would fail to work once the centralized trust is broken.

Recently, the blockchain technology has exhibited promise in constructing decentralized crowdsourcing systems, due to its inherent advantages like decentralization, transparency, and tamper-resistance. Some researches attempted to build general blockchain-assisted frameworks ([1], [10]–[12]) with no fine-grained implementation of some modules or having not fully addressed some crucial concerns in crowdsourcing, *e.g.*, *fine-grained* data quality evaluation with *truth* and *privacy* guarantees, *reliable* and *efficient* reputation assessment, and *fairness* offered to all participants. Recently, a series of researches like blockchain-based quality and reputation evaluation ([13]–[18]), truth discovery/selling ([19], [20]), truthful incentive mechanism ([21], [22]), and anonymous crowdsourcing ([23]–[27]) have been investigating some of the above key issues. Despite their nontrivial contributions, these prior solutions still suffer from the following limitations.

First, existing works using simple methods such as majority voting [1], [23], [24] and cluster median [13] to aggregate the sensory data and decide the task truths only provide coarse-grained data evaluation, which are not accurate enough and fail to capture the differences in worker reliabilities. To deal with malicious users, those relying on miners or workers to evaluate/verify data quality [14], [21], [28] are at the expense of data privacy, as the sensory data is exposed to the miners/workers. Using smart contracts can ensure reliable data evaluation but prior schemes are limited to simple data integrity evalua-

H. Wu and B. Düdder are with the Department of Computer Science, University of Copenhagen, 2100 Copenhagen, Denmark (e-mail: hw@di.ku.dk; boris.d@di.ku.dk).

L. Wang and S. Sun are with the Department of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China, 212013 (e-mail: wanglm@ujs.edu.cn; shipusun@163.com).

G. Xue is with the School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85287 USA (e-mail: xue@asu.edu). The research of Xue was supported in part by NSF grants 1717197 and 2007083.

This work was supported in part by the National Key R&D Program of China under Grant 2017YFB1400700, the National Natural Science Foundation of China under Grant U1736216, and by the Danish Ministry of Education and Science, Digital Pilot Hub and Skylab Digital. The information reported here does not reflect the position or the opinion of the funding agencies.

tion [1] or prone to tolerate costly zero-knowledge proofs ([23], [24]). Moreover, solutions assigning evaluation to the task requesters either fail to consider the malicious behaviors of requesters [18] or have technical flaws in the verification method [15]. It is a non-trivial challenge to provide a trustworthy (*i.e.*, computation verifiable) and accurate data evaluation approach against malicious users without compromising the workers' data privacy in blockchain-based crowdsourcing, due to the transparent characteristic of blockchain and the costly on-chain computations especially for cryptographic operations. Second, using blockchain for reputation management indeed enables a decentralized evaluation and storage of trust values (free from the above drawbacks of centralized management), and meanwhile provides public access to a global trust view for users to choose qualified workers or preferred tasks posted by the reliable requesters. However, existing blockchain-based reputation assessment/management is mainly based on the user's behaviors as a worker, neglecting the user's other roles such as task requester or verifier. Additionally, the reputation is updated as long as a task is completed, which would inevitably bring considerable burden for blockchain with increasing tasks. Developing a comprehensive reputation assessment model with efficient updates in blockchain-powered crowdsourcing systems is the second research challenge. Last but not least, most incentive mechanisms adopted quality-aware reward strategies. Their coarse-grained and inaccurate/biased quality evaluation will impair the reputation assessment accuracy, which is unfair to workers. Such social unfairness may lead to unfairness in reward distribution. The key to ensuring dual fairness lies in addressing the aforementioned two challenges. To the best of our knowledge, there has been no prior work addressing these challenges adequately.

Motivated by this, we leverage the **B**lockchain technology to build a **R**eliable, **P**rivacy-aware, and fair decentralized **C**rowdsourcing system, called BRPC. Apart from the generic blockchain-based crowdsourcing framework, we focus on elaborating how a fine-grained data quality evaluation with accurate truth inference can be verified correctly on encrypted data (the first challenge). For the second challenge, we capture the different roles of a user to formulate a novel reputation evaluation model, which is efficient to update based on a batch of tasks and avoids the frequent on-chain reputation updates. On this basis, an incentive strategy is devised to reward or penalize users, which entitles the users' dual fairness. The main contributions of our work are summarized as follows.

- We present a general decentralized crowdsourcing system, named BRPC, via blockchain. In contrast to existing blockchain-based frameworks, BRPC dives into the concrete crowdsourcing modules and key functionalities against curious and even malicious participants. BRPC achieves system reliability, user privacy, accurate truths, and financial-social fairness of all users.
- Based on BRPC, we employ a confident-aware truth discovery algorithm to estimate the task truths meanwhile evaluating the data quality of each worker. Task requesters are entitled to evaluate the reliability of the workers based on the inferred truths. Notably, we propose a privacy-aware computation verification protocol based on the Threshold Paillier Cryptosystem to avert incorrect/biased evaluation from the malicious requesters. Furthermore, a three-role-based reputation assessment is developed and enforced by smart contracts for its trustworthy execution. Besides social fairness, BRPC also realizes financial fairness with our quality-aware reward allocation and misbehavior-oriented punishment strategies.
- We implement a prototype of BRPC over Ethereum. The experiment results demonstrate its validity and effectiveness with an affordable overhead.

The remainder of this paper is organized as follows. In Section II, we introduce background knowledge. Section III states the research problem. Section IV describes our proposed BRPC system in detail. The security analysis and experimental evaluations are presented in Section V and Section VI, respectively. Section VII reviews some related work. We conclude this paper in Section VIII.

II. Background

Blockchain & Smart Contract. A blockchain is essentially a replicated, immutable, and distributed ledger maintained by a P2P network with a specific consensus mechanism. Each block aggregates a sequence of transactions and is chained to the previous block via a cryptographic hash function. The salient features of blockchain include *decentralization and anonymity*, *transparency and immutability*, and *distributed consensus*.

A smart contract is a kind of self-executing and self-verified digital contract programmed on the blockchain securely, which can be verified by the blockchain nodes. It can be triggered by certain events to carry out predefined functions in the contract.

Confidence-Aware Truth Discovery. In crowdsourcing applications, truth discovery (TD) algorithms effectively solve conflicts among the sensory data submitted by multiple workers, and provide truthful information about sensing tasks. Unlike some simple methods such as taking the *majority* as the "truth", TD captures the different reliability levels of workers, which works by iteratively estimating the truths via weighted aggregation over sensory data among workers and updating the reliability degrees of different workers in the form of weights, until some convergence criterion is reached.

In this paper, aiming at the generic crowdsourcing scenarios, we resort to the representative confidence-aware TD algorithm (CATD) [29] which exhibits better accuracy performance and considers the ubiquitous long-tail phenomenon in crowdsourcing tasks. The rationale behind CATD lies in that if a worker is more reliable, the probability that he gives trustworthy information is higher, and the worker who provides trustworthy information is more reliable. Additionally, if a worker takes more tasks, it is more likely that the worker's weight estimation is closer to his true reliability degree, *i.e.*, the confidence of weight estimation is higher. In contrast, if a user only takes a few tasks, the weight estimation confidence is low. W.L.O.G., let $\mathcal{T}$ and $\mathcal{U}$ be the set of sensing tasks and workers, respectively. The sensory data submitted by a worker $U_i \in \mathcal{U}$ for a task $\tau_j \in \mathcal{T}$ is represented by x_j^i, and the estimated truth for task τ_j is x_j^*. The detailed iterative process in CATD mainly consists of two sub-steps: truth estimation and weight estimation.

1) Truth Estimation: Given all workers' sensory data and the weight ω_i for each worker U_i, the estimated truth for each task τ_j is computed as

$$x_j^* = \frac{\sum_{U_i \in \mathcal{U}_j^*} \omega_i \cdot x_j^i}{\sum_{U_i \in \mathcal{U}_j^*} \omega_i}, \tag{1}$$

where $\mathcal{U}_j^*$ denotes the set of workers participating in task τ_j.

2) Weight Estimation: Given all workers' sensory data and the estimated truth x_j^*, the weight for each worker U_i is estimated as

$$\omega_i = \frac{\chi^2_{\alpha/2, |\mathcal{T}_{U_i}|}}{\sum_{\tau_j \in \mathcal{T}_{U_i}} (x_j^i - x_j^*)^2}, \tag{2}$$

where $\chi^2_{\alpha/2,|\mathcal{T}_{U_i}|}$ is a coefficient used to adjust the worker's weight. Specifically, χ denotes the Chi-squared distribution and α is the significance level (a small constant). $\mathcal{T}_{U_i}$ represents the set of tasks chosen by U_i in the current time slot.

The above iterative procedure does not terminate until some predefined criterion is met, such as reaching a maximum number of iterations I_{max}. Note that, we use CATD algorithm as a building block as it provides a more accurate truth finding solution as demonstrated in [29]. Based on this, we can conduct a fine-grained and reliable data quality evaluation for the workers. Since a worker's weight indicates the reliability of data he submitted, we use it to quantify the data quality, and use worker weight and data quality interchangeably in the latter scheme description (see Section IV-B).

Threshold Variant of Paillier Cryptosystem. In [30], Paillier proposed a new probabilistic asymmetric encryption scheme for public key cryptography. The scheme is known as its attractive additive homomorphic property. Given a public key $pk = (g, n)$ (where $g \in \mathbb{Z}^*_{n^2}$) and ciphertexts of two messages $m_1, m_2 \in \mathbb{Z}_n$, we can derive the ciphertext of m_1+m_2. Concretely, the homomorphic properties are as follows.

$$\mathsf{E}_{pk}(m_1 + m_2) = \mathsf{E}_{pk}(m_1) \cdot \mathsf{E}_{pk}(m_2), \tag{3}$$

$$\mathsf{E}_{pk}(a \cdot m_1) = \mathsf{E}_{pk}(m_1)^a, \tag{4}$$

where $\mathsf{E}_{pk}(m_1) = g^{m_1} r_1^n \ mod \ n^2$, $r_1 \in \mathbb{Z}^*_{n^2}$, and a is constant from set N.

In this paper, we use the threshold variant of the Paillier Cryptosystem to cater for our scenario requirements, with which the ciphertext of worker weight can be jointly decrypted by a sufficient number (*i.e.*, no lower than the threshold) of workers. With this idea, the correctness of the quality evaluation can be ensured by the joint cooperation among workers. Take (t, N)-Threshold Paillier Cryptosystem as an example, there are N shares of the private key sk, which are shared among N entities and each entity has a private key share (also called a piece). For N pieces $sk^1, sk^2, \ldots, sk^N$, at least t pieces are required to collaboratively recover the plaintext encrypted by pk. The decryption process contains two steps: share decryption at each entity and share combination for plaintext recovery. Due to space limitations, we refer the readers to [31] for details.

III. Problem Statement

In this section, we provide a formal definition of the blockchain-based reliable and privacy-aware crowdsourcing problem. Before that, we first present the BRPC system model, and then describe the security threats and security goals.

A. System Model

As illustrated in Fig. 1, BRPC consists of four kinds of roles: task requesters, workers, blockchain nodes, and a decentralized database. The detailed description is as follows.

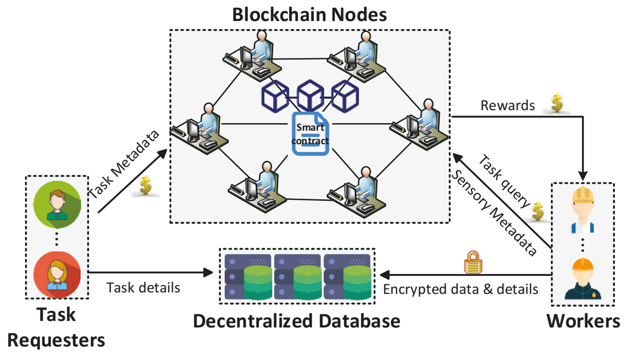

Fig. 1. System architecture.

Task Requesters, identified by $\mathcal{R} = \{R_1, R_2, \ldots, R_M\}$, are task owners who create tasks, recruit a certain number of workers, and reward workers according to their contributed data quality. To efficiently find the qualified workers and stimulate worker's participation, each task with a task identifier contains a task specification, including task time/location, task category, sensing object, data evaluation algorithm, worker's minimum reputation, number of workers, and task budget. Note that, considering the scalability bottleneck of blockchain, as in prior researches [1], [14], [23], [24], we choose to store the task details in the decentralized storage, and only record the task hash on the blockchain. Moreover, to prevent requester's false-reporting attacks, a certain amount of deposit is locked in a designated account when posting tasks, which cannot be redeemed by the requesters before a designated time.

Workers, represented by $\mathcal{U} = \{U_1, U_2, \ldots, U_N\}$, are mobile users who carry powerful mobile devices and compete for different tasks to get rewards. Each worker can search the on-chain tasks and has opportunities to accept tasks as long as he is qualified to work on. If a qualified worker wants to execute a task, he is required to make a deposit on the blockchain as a prevention against Sybil or DDoS attacks. Once assigned, workers collect sensory data and send them to the blockchain. Similarly, we combine on-chain storage with off-chain storage (encrypted data) to relieve the blockchain cost. After the data is evaluated by the requesters, each worker earns the corresponding rewards. Meanwhile, workers executing the same task can collaboratively verify the correctness of the requester's evaluation. Note that, in the crowdsourcing process, a worker can also act as a task requester, and vice versa.

Blockchain Nodes are network peers maintaining identical replicas of the blockchain. Specifically, miners verify transactions, organize all the verified transactions with a Merkle Hash

Tree, and compete for generating new blocks. Task requesters and workers can act as miners if they join in the mining work. In this paper, we consider a public blockchain like Ethereum.

Decentralized Database is a decentralized storage system which employs service peers (*e.g.,* peers in IPFS) to offer storage/computation services by renting out their resources.

In BRPC, w.l.o.g., we assume that time is divided into time slots $T_1, T_2, \ldots$, and there are C task categories, denoted by $\mathcal{C} = \{1, 2, \ldots, C\}$, with different difficulty degrees (indicated by different weights in reputation evaluation). In a time slot T_l, each task requester $R_i \in \mathcal{R}$ posts a set of tasks $\mathcal{T}_{R_i}^l = \{\tau_1, \tau_2, \ldots, \tau_m\}$ with task category $c \in \mathcal{C}$. Each task $\tau_j \in \mathcal{T}_{R_i}^l$ specifies a budget $\mathcal{B}_j$ to collect data from n_j workers whose global reputation and task expertise reputation values are no lower than ε_j^1 and ε_j^2, respectively.

Let $\mathcal{U}_i^l$ be the set of users performing R_i's tasks in time slot T_l. We denote the set of tasks chosen by worker $U_j \in \mathcal{U}_i^l$ and the set of tasks published by requester R_i in T_l as $\mathcal{T}_{U_j}^l$ and $\mathcal{T}_{R_i}^l$, respectively. We have $\mathcal{T}_{U_j}^l \subset \mathcal{T}_{R_i}^l$. Each worker $U_j \in \mathcal{U}_i^l$ submits the encrypted sensory data and R_i obtains the original data after decryption. Due to the diverse reliabilities of workers and the uncertainty of task truths, our first issue is to let R_i discover the most trustworthy information from multiple sensory data and evaluate the reliability of each worker in completing this kind of tasks, *i.e.*, data quality. Considering the possible unreliable evaluation from malicious requesters, the next issue is how to offer a privacy-aware verifiable scheme to ensure the evaluation correctness and the fairness of incentives. Last, since each user may act in multiple roles in different phases, it is a crucial problem to comprehensively model these roles and develop a reliable and decentralized reputation assessment method with efficient update.

B. Threat Model

The security threats are mainly from the internal attackers. Specifically, task requesters may be curious about the true identities of workers and vice versa. Moreover, they may want to learn the sensory data of other requesters' tasks. Similarly, the blockchain nodes may be curious about the sensory data of specific tasks or data submitted by specific workers.

More seriously, task requesters and workers may behave maliciously to maximize their profits. On one hand, malicious requesters may want to collect sensory data without giving payments, or giving unfair rewards to the workers. For example, for two workers submitting data with the same quality, a malicious requester may give an unknown worker lower rewards while giving an acquainted worker higher rewards. Additionally, they may misreport high-quality data as low-quality data (biased/incorrect evaluation), or even repudiate the fact of obtaining the sensory data (false-reporting). On the other hand, malicious workers may attempt to earn rewards with no/less effort (free-riding). They may forge data or submit low-quality data. Moreover, they may create multiple fake identities (Sybil attack) to request tasks for more rewards, or purposely do not submit data on time after choosing many tasks (DDoS), resulting in an insufficient data collection and discouraging the requesters' participation. Some malicious workers may collude with each other to submit low-quality data, try to infer the private key of each requester, or misreport the verification results.

In this paper, we assume that the tasks submitted by a requester in a time slot are executed by at least t workers, in which there are fewer than t colluding workers. These are the basic security requirements in (t, N)-Threshold Paillier Cryptosystem. Additionally, we assume that task requesters and workers have a limited amount of money and the amount of deposit is more than the rewards. For blockchain security, we assume that the majority of blockchain nodes are honest, and attackers do not have the power to control the blockchain.

C. Design Goals

In light of the above security threats, we elicit the following security requirements for our BRPC system.

1) Privacy Protection. For identity privacy, the true identities of requesters and workers should not be revealed to any other system entities. For data privacy, we aim to protect the confidentiality of sensory data. Only the requester can obtain the plaintexts and the estimated ground truths of his tasks.

2) Reliability. From the system's perspective, single point of failure should be mitigated. From the user's perspective, we should guarantee that the task requesters are able to obtain reliable truths for their tasks. To prevent malicious requesters from deliberately misreporting the data quality, we need to offer a verifiable computation solution to validate the correctness of data quality. Moreover, for reputation evaluation of users, we should ensure that the whole evaluation process is public and the correctness is publicly verifiable.

3) Dual Fairness. We consider two kinds of fairness: transaction (*i.e.*, financial) fairness and evaluation (*i.e.*, social) fairness. For the former, requesters should be able to get their deposit back only if the corresponding rewards are given to the workers and their data quality evaluation is correct. The rewards earned by the workers should depend on the submitted data quality (higher quality means more rewards) and workers can get their deposit back only if they follow the designated protocol. Moreover, workers as verifiers can earn equivalent rewards if they correctly report the verification results. For the latter, the evaluation methods of data quality and user reputation are disclosed to the public. All joined workers can verify the data quality in their joined tasks. The trust evaluation should comprehensively consider the different roles of a user.

IV. Blockchain-based crowdsourcing protocol

In this section, we formalize the BRPC system and show how it works to address the reliability, privacy, and fairness challenges in existing crowdsourcing systems.

A. Overview

At a high level, the entire crowdsourcing process of BRPC consists of five phases, as shown in Fig 2. In Phase 1, the system parameters are initialized and each user (worker/task requester) needs to be registered in the system for later participation. In task publication (Phase 2), any requester with task requirements posts tasks with a deposit to the BRPC system via a smart contract in the form of transactions. With the on-chain task state information, workers can query available/valid tasks and are eligible to participate in the tasks if their reputation values satisfy a qualification check function. Similarly, a predefined amount of money from the workers is locked as deposit once the task execution is confirmed. To enable the following privacy-aware verification on the worker side, the requester distributes his private key shares to the participating workers before data submission (Phase 3).

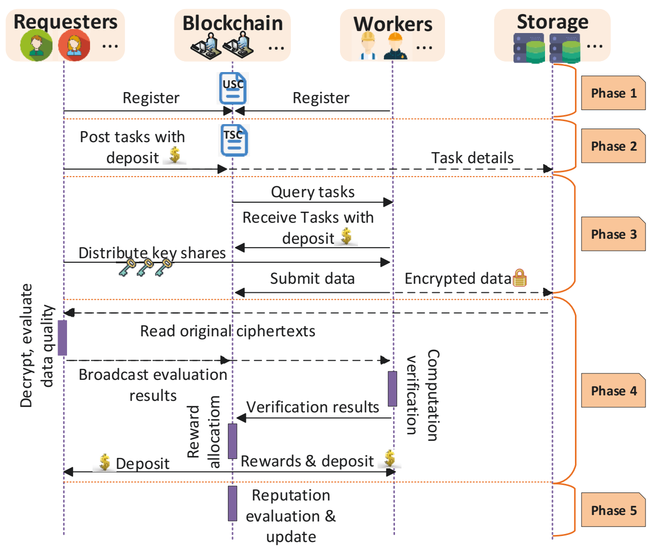

Fig. 2. The interactive process among entities in BRPC.

With all encrypted data submitted, the requester will turn to the CATD algorithm to infer the task truths and evaluate the data quality of each worker after decryption. Meanwhile, workers are incentivized to cooperatively verify the correctness of the evaluation results. For remuneration, each worker is first rewarded according to the quality of their contributed data. Moreover, workers and task requesters will earn extra rewards or lose the deposit, depending on their benign/malicious behaviors (Phase 4). Finally, BRPC uses the smart contract to enforce reliable reputation evaluation and update (Phase 5).

B. Protocol Details

1) System Initialization: This phase consists of two steps: setup and user registration.

In the setup step, we assume that parameters such as the duration of each time slot, C and the corresponding weight of each category $c \in C$ are set by a bootstrap program in BRPC. Moreover, the system also sets the user's initial global reputation and expertise reputation regarding different task categories, based on the average reputation level of humans. To depict the task process, a set of task states $\mathcal{S} = \{$*Pending*, *Available*, *Unavailable*, *QEvaluating*, *Completed*, *Canceled*$\}$ is predefined in a task smart contract (TSC).

In the registration, task requesters and workers register in the BRPC system as users. Specifically, each user i will obtain a key pair (pk_i, sk_i) and a public address A_i, *i.e.*, pseudonym. The initial global and expertise reputation values of i are $\gamma_{i,1}^g$ and $\gamma_{i,1}^c$, $c \in C$. When i registers, a user smart contract (USC) is invoked which records A_i and his corresponding information including global and expertise reputation values. This information will be updated at the end of each time slot.

2) Task Publication: In this phase, each requester $R_i \in \mathcal{R}$ posts a series of tasks $\mathcal{T}_{R_i}^l = \{\tau_1, \tau_2, \ldots, \tau_m\}$ to BRPC in T_l via transactions. Specifically, each task $\tau_j \in \mathcal{T}_{R_i}^l, j \in [1, m]$ contains the following information:

$$\tau_j = \{T_{id}, c_i, A_i, S_{deadline}, P_{deadline}, T_{deadline}, pk_i, \\ \varepsilon_j^1, \varepsilon_j^2, n_j, \mathcal{B}_j + \mathcal{D}_{R_i}^j, \mathsf{checkQ}, \mathsf{QEvaluate}\}, \tag{5}$$

where T_{id} denotes the task identifier and c_i is the category of current tasks posted by R_i. Moreover, $S_{deadline}$ is the data submission deadline. $P_{deadline}$ is the proving deadline before which the correctness of data quality evaluation should be verified. $T_{deadline}$ is the task deadline before which task τ_j should be completed (each task ends with reward allocation). pk_i is the public key of R_i. The global and expertise reputation thresholds are ε_j^1 and ε_j^2, respectively. The deposit $\mathcal{D}_{R_i}^j$ committed by R_i for task τ_j consists of two parts: the first part is used to prevent false-reporting attacks and the second part is used to reward workers (or verifiers) who participate in the correctness verification process (*i.e.*, as an incentive to motivate workers' participation in verification). Function checkQ: $\boldsymbol{\varepsilon} \times \boldsymbol{\gamma} \rightarrow bool$ checks if the worker is qualified to perform task τ_j based on his reputation, and $\mathsf{QEvaluate}: \boldsymbol{\omega} \times \boldsymbol{x} \rightarrow \boldsymbol{x}^*$ specifies the CATD algorithm. Note that, once a task is submitted by the requester, it is initiated with *Pending* and is waiting for miners' confirmation. The state switches to *Available* when the task is recorded on-chain.

3) Task Receiving and Data Submission: Any worker $U_k \in \mathcal{U}$ can find *available* tasks by querying the task state in TSC and has a chance to execute their interested tasks if they are qualified. For each task $\tau_j \in \mathcal{T}_{R_i}^l$, $\mathsf{checkQ}(\boldsymbol{\varepsilon}, \boldsymbol{\gamma}) \rightarrow bool$ determines if $\gamma_{k,l}^g \geq \varepsilon_j^1$ and $\gamma_{k,l}^{c_i} \geq \varepsilon_j^2$. Once a qualified worker U_k confirms to execute τ_j, a certain amount of deposit $\mathcal{D}_{U_k}^j$ will be locked, in which part deposit is used to tackle worker's misbehaviors (free-riding, Sybil attack, and DDoS) and another part is used as reward to motivate workers to act as verifiers. If the account balance of U_k is less than $\mathcal{D}_{U_k}^j$, he cannot execute the task. The state of τ_j switches into *Unavailable* when n_j workers confirm τ_j.

R_i queries the task state in TSC, if each of his posted task $\tau_j \in \mathcal{T}_{R_i}^l$ is *Unavailable*, R_i queries all participating workers, *i.e.*, $\mathcal{U}_i^l$. Next, R_i splits his private key sk_i into $|\mathcal{U}_i^l|$ shares $sk_i^1, sk_i^2, \ldots, sk_i^{|\mathcal{U}_i^l|}$. We can observe that $|\mathcal{U}_i^l| \leq \sum_{j=1}^m n_j$ since each worker may take two or more tasks. Subsequently, R_i encrypts $sk_i^1, sk_i^2, \ldots, sk_i^{|\mathcal{U}_i^l|}$ with workers' public keys $pk_1, pk_2, \ldots, pk_{|\mathcal{U}_i^l|}$, respectively, and sends the encrypted key shares $\left\{\mathsf{E}_{pk_j}(sk_i^j)\right\}_{j=1}^{|\mathcal{U}_i^l|}$ to the corresponding workers via a secure channel. After that, each worker decrypts with his private key and obtains the key share. Note that, this key share distribution ensures that every participating worker is able to join in later evaluation verification and partially decrypts the ciphertext with his assigned key share.

According to the task requirements, each worker $U_k \in \mathcal{U}_i^l$ collects sensory data x_j^k and sends $\mathsf{E}_{pk_i}(x_j^k)$ to the decentralized database while the corresponding hash is stored on the blockchain. Anyone can get $\mathsf{E}_{pk_i}(x_j^k)$, but only R_i can perform decryption $\mathsf{D}_{sk_i}(\mathsf{E}_{pk_i}(x_j^k))$. Note that if U_k successfully submits the encrypted data before $S_{deadline}$, the part deposit (for tackling worker's misbehaviors) of $\mathcal{D}_{U_k}^j$ will be refunded to U_k. Otherwise, it will be deducted and used to reward the verifiers later. The state of τ_j transits to *QEvaluating* when the hash of n_j ciphertexts are recorded on the blockchain.

4) Data Quality Evaluation and Reward Allocation: With all sensory data $\{x_j^k\}_{k=1,\tau_j\in\mathcal{T}_{U_k}^l}^{|\mathcal{U}_i^l|}$ submitted by each worker $U_k \in \mathcal{U}_i^l$ for task $\tau_j \in \mathcal{T}_{U_k}^l$, R_i performs the CATD algorithm and iteratively updates the weight of each worker and the estimated task truths until I_{max} is satisfied. After that, R_i derives the estimated truth x_j^* for each task $\tau_j \in \mathcal{T}_{U_k}^l$ and the estimated weight/reliability ω_k of each worker $U_k \in \mathcal{U}_i^l$ in slot T_l according to Eqs. (1) and (2). Moreover, ω_k is broadcasted to the network for worker-side verification.

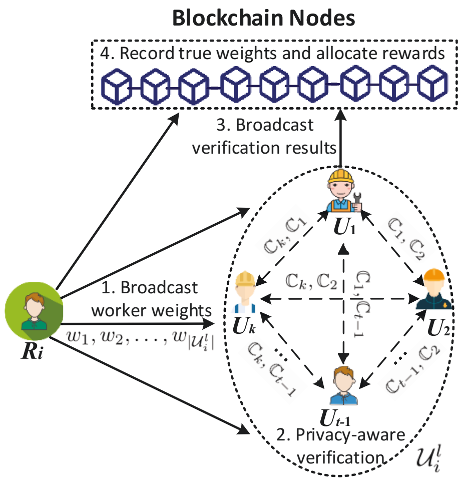
Blockchain Nodes
4. Record true weights and allocate rewards
3. Broadcast verification results
1. Broadcast worker weights
$w_1, w_2, \ldots, w_{|\mathcal{U}_i^l|}$
Ri
U1
Uk
U2
Ut-1
$\mathbb{C}_k, \mathbb{C}_1$
$\mathbb{C}_1, \mathbb{C}_2$
$\mathbb{C}_k, \mathbb{C}_2$
$\mathbb{C}_1, \mathbb{C}_{t-1}$
$\mathbb{C}_k, \mathbb{C}_{t-1}$
$\mathbb{C}_{t-1}, \mathbb{C}_2$
2. Privacy-aware verification
$\mathcal{U}_i^l$

Fig. 3. An example of workers' joint verification.

Fig. 3 illustrates an example of workers' joint verification of requester-side data quality evaluation. After obtaining the weights broadcasted (Step 1), each worker $U_k \in \mathcal{U}_i^l$ finds the encrypted sensory data from the decentralized database. Being aware of $\left\{\mathsf{E}_{pk_i}(x_j^k)\right\}_{k=1,\tau_j\in\mathcal{T}_{U_k}^l}^{|\mathcal{U}_i^l|}$, the workers can jointly run Algorithm 1 to perform privacy-aware verification (Step 2 in Fig. 3), in which each derived ciphertext $\mathsf{E}_{pk_i}(\widetilde{d}_j^k)$ (Line 12 in Algorithm 1) is partially decrypted using the key share. The corresponding result is broadcasted to the other participating workers. For example, as depicted in Fig. 3, U_1 broadcasts the partially decrypted ciphertext $\mathbb{C}_1$ to the other workers and meanwhile he also receives information from others (each worker can query the on-chain recorded requester-task and task-worker information and find other workers working for the same requester. The verification group is composed of these workers and each worker only accepts information from other workers in the same verification group).

Algorithm 1: Privacy-aware computation verification of worker's data quality

```
   Input: $\{[x_j^k]_{pk_i}\}_{k=1,\tau_j\in\mathcal{T}_{U_k}^l}^{|\mathcal{U}_i^l|}$, $\{\gamma_{k,l-1}^{c_i}\}_{k=1}^{|\mathcal{U}_i^l|}$, $I_{max}$, $\omega_k$,
          rounding factor $L$.
   Output: Verification results $f_k$.
1  // Worker side;
2  $\{\omega'_k\}_{k=1}^{|\mathcal{U}_i^l|} = \{\gamma_{k,l-1}^{c_i}\}_{k=1}^{|\mathcal{U}_i^l|}$ for each $U_k \in \mathcal{U}_i^l$;
3  for $I = 1$ to $I_{max}$ do
4      // Estimated truth update;
5      for $j = 1$ to $m$ do
6          $\mathsf{E}_{pk_i}(L \cdot \omega'_k x_j^k) = \mathsf{E}_{pk_i}(x_j^k)^{L\cdot\omega'_k}$ for each
             $U_k \in \mathcal{U}_j^*$;
7          $\mathsf{E}_{pk_i}(L \sum_{U_k\in\mathcal{U}_j^*} \omega'_k x_j^k) =$
             $\prod_{U_k\in\mathcal{U}_j^*} \mathsf{E}_{pk_i}(L \cdot \omega'_k x_j^k)$;
8          $\mathsf{E}_{pk_i}(\widetilde{x}_j^*) = \mathsf{E}_{pk_i}(L \cdot \sum_{U_k\in\mathcal{U}_j^*} \omega'_k x_j^k)^{\lfloor \frac{L}{\sum_{U_k\in\mathcal{U}_j^*} \omega'_k} \rfloor}$;
9      // Estimated weight update;
10     for $k = 1$ to $|\mathcal{U}_i^l|$ do
11         $\mathsf{E}_{pk_i}(\widetilde{d}_j^k) = \mathsf{E}_{pk_i}(x_j^k)^{L^2} \cdot \mathsf{E}_{pk_i}(\widetilde{x}_j^*)^{n-1}$ for each
             $\tau_j \in \mathcal{T}_{U_i}^l$;
12         $\mathbb{C}_k = \mathsf{D}_{sk_i^k}(\mathsf{E}_{pk_i}(\widetilde{d}_j^k))$;
13         Broadcast $\mathbb{C}_k$ to other workers in $\mathcal{U}_i^l$;
14         Once $U_k$ receives other $t-1$ partial decrypted
             ciphertexts $\mathbb{C}_1, \mathbb{C}_2, \ldots, \mathbb{C}_{t-1}$, he combines
             them with his own $\mathbb{C}_k$ and recovers $\widetilde{d}_j^k$ for
             each $\tau_j \in \mathcal{T}_{U_i}^l$;
15         $\omega'_k = \frac{y_k}{\sum_{\tau_j\in\mathcal{T}_{U_k}^l} (\widetilde{d}_j^k/L^2)^2}$;
16 for $k = 1$ to $|\mathcal{U}_i^l|$ do
17     if $\omega'_k == \omega_k$ then
18         $f_k = 1$;
19     else if $\omega'_k \neq \omega_k$ then
20         $f_k = 0$;
```

For Algorithm 1, specifically, each worker's weight is initialized with his latest expertise reputation regarding task category c_i (Line 2, set it to $\gamma_{k,1}^{c_i}$ if $l = 1$). Next, the estimated truths and weight updates are iteratively performed on encrypted data for I_{max} times. Concretely, the encrypted estimated truth for task τ_j is derived as $\mathsf{E}_{pk_i}(\widetilde{x}_j^*)$ where $\widetilde{x}_j^* = L \cdot x_j^*$ and L is a scaling factor to deal with fractional data (Line 8). In Section VI, we will show that the accuracy of our result is not compromised if a proper L is chosen. In the following weight update phase, the encrypted difference between $L \cdot x_j^k$ and $\widetilde{x}_j^*$ is first calculated, denoted by $\mathsf{E}_{pk_i}(\widetilde{d}_j^k)$ (Line 11). Since Paillier Cryptosystem does not have multiplicative homomorphic property, it is hard to derive $\mathsf{E}_{pk_i}((\widetilde{d}_j^k)^2)$ by a single worker based on $\mathsf{E}_{pk_i}(\widetilde{d}_j^k)$. Instead, we let each worker U_k decrypt $\mathsf{E}_{pk_i}(\widetilde{d}_j^k)$ using his partial private key (*i.e.*, pre-assigned key share) sk_i^k, and the partial decrypted ciphertext $\mathbb{C}_k$ is broadcasted to other workers in $\mathcal{U}_i^l$.

Leveraging $(t, |\mathcal{U}_i^l|)$-Threshold Paillier Cryptosystem, as long as U_k gets $t-1$ partial decrypted ciphertexts, he can recover $\widetilde{d}_j^k$ and compute each worker's weight ω'_k (Line 15), which will be utilized as input of the next iteration. After reaching I_{max}, U_k obtains the final weights and compares them with ω_k (broadcasted by R_i), if they are equivalent, the verification result f_k is set to 1, otherwise, it is set to 0. Before $P_{deadline}$, $U_k \in \mathcal{U}_i^l$ signs $f_j, j \in [1, |\mathcal{U}_i^l|]$ using sk_k and broadcasts the verification result (Step 3 in Fig. 3). Concretely, if $f_j = 1$, U_k broadcasts $(A_k, A_i, A_j, 1, Sig_{sk_k}(h(1||A_i||A_j)))$. Otherwise, $(A_k, A_i, A_j, 0, \omega'_j, Sig_{sk_k}(h(0||A_i||A_j||\omega'_j)))$ is broadcasted. Note that the later includes the correct weight of U_j (*i.e.*, ω'_j). All blockchain nodes can verify the integrity of the verification messages and miners can upload the correct weights to the blockchain. Although some workers may collude with each other and report incorrect verification results either to conceal R_i's misbehavior or slander honest requesters, such misbehaviors can be easily identified as we assume that the majority of the verification workers are honest. Solutions dealing with massive malicious workers are out of the scope of this paper and will be investigated in our future work.

When $P_{deadline}$ arrives, if all signatures are valid and there are no zeros with respect to R_i, each task budget $\mathcal{B}_j, j \in [1, m]$ is redeemed and the number of rewards assigned to each worker $U_k \in \mathcal{U}_j^*$ for each task $\tau_j \in \mathcal{T}_{U_i}^l$ is:

$$\xi_k^j = \begin{cases} \frac{\omega_k}{\sum_{U_k \in \mathcal{U}_j^*} \omega_k} \cdot \mathcal{B}_j & \tau_j \in \mathcal{T}_{U_k}^l, \\ 0 & \tau_j \notin \mathcal{T}_{U_k}^l. \end{cases} \tag{6}$$

From Eq. (6), we can observe that if U_k contributes data to task τ_j, he earns rewards proportionally to his data quality ω_k. Otherwise, nothing is rewarded. Therefore, the total number of rewards given by R_i to U_k is $\sum_{j=1}^m \xi_k^j$.

Moreover, the first part deposit of $\mathcal{D}_{R_i}^j$ is returned to R_i while another part is evenly distributed among the verifiers as rewards. In this case, the number of correct quality evaluation is $|\mathcal{U}_i^l|$ for R_i, which is regarded as a reputation evaluation factor. In contrast, if there exists a message containing zero regarding R_i, the rewards allocation is based on the evaluation results broadcasted by the most verifiers. Once R_i gives the wrong quality evaluation, $\mathcal{D}_{R_i}^j$ will not be refunded and will be used as verification rewards. Additionally, as a task requester, the reputation of R_i would be affected. Similarly, for verifiers, if they report incorrect verification results, they would get no rewards. Moreover, their reputation would degrade as a verifier. After reward allocation, the task is *completed*. In the above four phases, if the requester cancels the task or the task is expired, the task state will be *canceled*.

5) Reputation Evaluation/Update: We take the different roles of users into account and assign different weights to aggregate a user's global reputation. Moreover, we also examine the reliability of a user as a worker executing tasks with distinct categories and build an expertise reputation model. *W.o.l.g.*, in T_l, we assume that as a worker, U_k receives tasks posted by a set of requesters $\mathcal{R}' \subseteq \mathcal{R}$. Meanwhile, as a requester, U_k publishes a set of tasks assigned to η workers.

In data quality evaluation, let $w_{k,l}^{c_i}$ be the estimated weight of U_k who performs tasks for $R_i \in \mathcal{R}'$ in T_l, and the global weight of U_k as a worker is:

$$\gamma_k^w = \sum_{R_i \in \mathcal{R}'} \alpha_{c_i} \cdot w_{k,l}^{c_i}, \tag{7}$$

where α_{c_i} is the weight of task category c_i.

When U_k acts as a requester, we assume that U_k correctly evaluates and broadcasts the estimated weights of η_k workers. Hence, the reliability of U_k as a requester is $\gamma_k^r = \eta_k/\eta$. Similarly, as a verifier, if U_k joins in λ verifications, out of which λ_k verifications are correct, the reliability of U_k as a verifier is $\gamma_k^v = \lambda_k/\lambda$. Comprehensively, the global reputation value of U_k can be modeled using the following equation.

$$\gamma_{k,l}^g = (1-\beta)(\rho_1\gamma_k^w + \rho_2\gamma_k^r + \rho_3\gamma_k^v) + \beta\gamma_{k,l-1}^g, \tag{8}$$

where ρ_1, ρ_2 and ρ_3 denote the proportion of U_k's reliability as a worker, a requester, and a verifier, respectively. Moreover, $\rho_1+\rho_2+\rho_3 = 1$. β is the weight of historical global reputation.

As for U_k's expertise reputation regarding task category c, it is modeled as the user's weight in executing such type of tasks and can be updated as follows.

$$\gamma_{k,l}^c = (1-\beta)w_{k,l}^c + \beta w_{k,l-1}^c. \tag{9}$$

The aforementioned reputation evaluation and update will be conducted via smart contract for each time slot. Note that, the smart contract is executed by the blockchain nodes, and the correctness of execution results can be validated by each node on the public blockchain. Finally, the miners can compete for adding the reputation values to the blockchain (*i.e.,* consensus process). The correctness of related parameters of the reputation evaluation function in the smart contract is also guaranteed. For example, the reliability computation of a user acting as a worker can be verified by the participating workers as illustrated in Algorithm 1 and is further confirmed by the blockchain nodes with the broadcasted information. The number of correct evaluations and verifications as a requester and a verifier is also broadcasted to the network and verified accordingly. Hence, the correct reliability evaluation for different roles and the trusted execution of the smart contract ensure the trustworthiness of the final reputation values.

V. Security analysis

In this section, we will show that our proposed BRPC system can achieve the security goals stated in Section III-C.

Theorem 1. *BRPC ensures the privacy of requesters and workers throughout the entire crowdsourcing phase, under the assumption that fewer than t workers are corrupted out of $|\mathcal{U}_i^l|$ participating workers.*

Proof. In BRPC, the true identities of users are protected by the use of pseudonyms, *i.e.*, Ethereum account addresses. For data privacy, each sensory data x_j^k is encrypted with the requester's public key pk_i, and then stored in the decentralized database. Although everyone can retrieve $\mathsf{E}_{pk_i}(x_j^k)$ based on the on-chain metadata/hash, it is infeasible to infer the corresponding plaintext based on the semantic security of the public key encryption. Moreover, for $|\mathcal{U}_i^l|$ participating workers, corrupted workers cannot recover the plaintexts x_j^k and x_j^* as at least t shares are needed for successful recovery. □

Theorem 2. *BRPC achieves both system and user reliabilities. For a participating worker set $\mathcal{U}_i^l$, as long as t honest workers (i.e., verifiers) exist and there are fewer than t collusive workers, our protocol can ensure the correctness of data quality and user reputation evaluation without relying on a trusted third party.*

Proof. Due to the decentralized characteristics of blockchain, the single point of failure can be effectively prevented, which offers system reliability. For data quality evaluation of requesters, QEvaluate is included in each task and is open to all entities. By running QEvaluate, R_i derives the weight ω_k for each worker $U_k \in \mathcal{U}_i^l$ and obtains the task truths. ω_k is then broadcasted for worker-side verification. Based on Algorithm 1, it is observed that ω'_k will be derived without revealing each sensory data as long as t honest workers contribute their decrypted shares and there are fewer than t collusive workers in the remaining verifiers. Furthermore, the comparison between ω'_k and ω_k would validate the reliability of data quality evaluated by R_i. There may exist some malicious workers individually or collusively reporting incorrect verification results, but the final verification result is determined by the majority of honest verifiers. In other words, fewer than half of the verifiers cannot deviate the results even if they are all collusive. Hence the number of collusive verifiers n_{cv} should satisfy Eq. (10). Since when $t \geq \frac{|\mathcal{U}_i^l|}{2}$, n_{cv} is bound to be less than $\frac{|\mathcal{U}_i^l|}{2}$ even if $|\mathcal{U}_i^l|$ workers join the verification process, the correctness of data quality verification can be guaranteed with fewer than t collusive workers.

$$n_{cv} < \begin{cases} t & t < \frac{|\mathcal{U}_i^l|}{2}, \\ \frac{|\mathcal{U}_i^l|}{2} & t \geq \frac{|\mathcal{U}_i^l|}{2}. \end{cases} \tag{10}$$

Moreover, we use smart contracts for reputation evaluation, in which the reputation values are anchored to the blockchain. Hence, the reliability depends on the trusted execution of smart contracts (blockchain security and immutability). □

Theorem 3. *Financial fairness between the task requester and the worker as well as social fairness among the users are both realized, as long as each user is rational in participation.*

Proof. From the financial perspective, task requesters and workers need to deposit money on the blockchain before participation. On one hand, for a requester R_i posting task τ_j, he needs to deposit $\mathcal{B}_j$ which is allocated to the participating workers as long as R_i obtains the original sensory data. Additionally, he needs to make another deposit $\mathcal{D}_{R_i}^j$ consisting of two parts. The first part can only be redeemed when the quality evaluation of R_i is verified to be correct, or it will be allocated to the honest verifiers. Besides, such honest verifiers will also earn extra rewards from the remaining money of $\mathcal{D}_{R_i}^j$. On the other hand, for a worker U_k performing task τ_j, he obtains rewards proportional to the data quality as long as the contributed sensory data is submitted in time. Otherwise, U_k will get nothing and also lose the first part of his deposit $\mathcal{D}_{U_k}^j$. As such a deposit is much more expensive than the rewards and each worker has limited money, a rational worker will not take the risk to launch free-riding, Sybil, and DDoS attacks. If U_k honestly follows the designated protocol, the first part of $\mathcal{D}_{U_k}^j$ will be refunded. Similar to R_i, the second part also works as rewards for honest verifiers. In other words, task requesters and workers jointly use a partial deposit to stimulate workers to join in verification, which is fair in finance.

From the social perspective, we entitle the task requesters to evaluate data quality meanwhile ensuring that all participating workers can collaboratively verify the correctness in case of the requester's misbehaviors. The quality evaluation algorithm QEvaluate is public. Moreover, the reputation evaluation algorithm takes the three different social roles of a user into consideration, which is enforced by the smart contract. Obviously, users with good performance, such as submitting high-quality data, honestly reporting verification results, or evaluating data quality, will get a high reputation and are qualified to take more tasks. In contrast, malicious users reporting unreliable data/evaluation or verification results, mounting Sybil/DDos attacks will get a low reputation (or lose deposit) and may be filtered when requesting tasks. Therefore, BRPC achieves social fairness with the unified and immutable evaluation method and the correct evaluation results. □

VI. Experiments

A. Simulation Setup

We implemented BRPC over Ethereum and realized the functionalities of the key phases in crowdsourcing with smart contracts. Specifically, the USC and TSC were written in Solidity and were deployed to a local simulated network Ganache on a laptop with Intel Core i5-3210M CPU (2.50GHz) and 8G RAM. For user-side computations, we implemented the CATD algorithm using Java programming language.

Initially, 20 workers and 1 requester were registered with an initial reputation value 0.5. The requester had 10 tasks in default during each time slot. We used a synthetic dataset in our experiment. Follow the simulated data distribution in [29], all sensory data was generated from a normal distribution with a random mean $\mu \in [20, 40)$ (represents the ground truth) and variance $\sigma^2 \in [1, 2)$. The length of each sensory data (and worker weight) is set to 8 bytes (double type). We set the byte length of the verification result f_k to 1. For each Ethereum account address and signature in the broadcast information, we follow the Ethereum default setting, *i.e.*, 20-byte address and 65-byte signature, respectively. We require that each task is executed by at least one worker, and each worker performs at least one task. Under this premise, each worker randomly determines whether to contribute data to the tasks. We choose the same significance level $\alpha = 0.05$ as in CATD [29]. For cryptographic parameters, we set the public key to 512 bits to achieve sufficient security. It can be set to other values, depending on the security demand. Theoretically, a 1024-bit key can achieve 80-bit security levels [32]. The threshold t was set to half of the number of the participating workers.

We integrated the Paillier Threshold Encryption package[1] into CATD to implement worker-side verification.

Since gas usage is a crucial concern on Ethereum, we tested the gas usage and ether cost incurred under different number of posted tasks and submitted data to show the practicality of BRPC. In addition, we evaluated the accuracy of BRPC in truth discovery by taking the commonly used standard root of mean square error RMSE as the accuracy metric. On the one hand, we set the scaling factor L in the range from 10 to 10^5 to show its impact on the accuracy of CATD on unencrypted and encrypted data. In line with [29], we chose Median as the baseline for comparison on simulated data. On the other hand, we compared RMSE in different number of iterations and observed the convergence performance. Meanwhile, the corresponding verification cost, including worker-side computation and user-side communication cost, was also measured to evaluate its efficiency. From a high-level perspective, we compare our BRPC with some existing blockchain-based crowdsourcing schemes regarding specific functionalities and properties in Table I. Noticeably, no scheme can ensure computation verifiability against malicious users for fine-grained data evaluation and TD without sacrificing the data privacy. Due to the lack of functionalities and unrealized properties in prior literature, it is intractable to perform an empirical and quantitative comparison with BRPC in a fair way. Therefore, we only focus on the qualitative contrast.

B. Simulation Results

Gas Consumption. We deployed USC and TSC in Ganache, which consumes 431180 and 1530840 gas, respectively. The corresponding cost is 0.0086236ETH and 0.0306168ETH, leading to a total of \$6.13 transaction fee when referring to the average ETH price \$156.45 in Jan. 2020 (when we conducted the experiment). Such cost is not a big concern as the smart contracts were deployed only once.

Fig. 4 depicts the gas cost in the primary phases of BRPC versus the number of data records written into the blockchain. Our result shows a linear increase with more users registered, more tasks posted/accepted, and more data submitted. This is because gas cost is closely related to the size of data recorded on the blockchain and the type of computation operations. For the same phase with the same operations, the gas will be increased when more data records are written. However, we notice that task publishing requires more gas, about 10700000, *i.e.*, \$33.48 when compared to other phases for 50 written data records. The reason is that besides the task identifier, the task requester also needs to store other information on-chain for later task matching and reward allocation. Moreover, the budget and deposit transfer results in additional gas cost. For other phases, user registration and data submission have a similar number of written data, whereas accepting a task only needs to determine whether to add the requested workers into the task's worker list and charge the worker's deposit. Hence the former two phases present comparable gas cost, while task accepting is the most gas-efficient phase with only around 2500000 (\$7.8). The cost is acceptable as crowdsourcing tasks

[1]http://cs.utdallas.edu/dspl/cgi-bin/pailliertoolbox/.

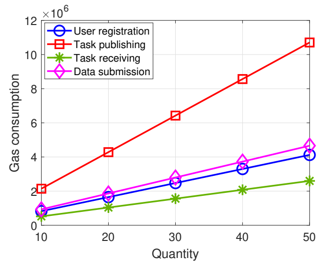

Fig. 4. Gas consumption in phases.

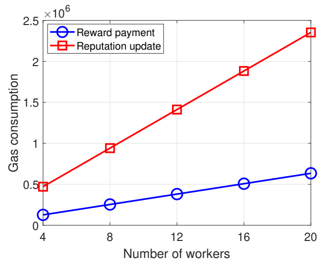

Fig. 5. Gas consumption in payment and update.

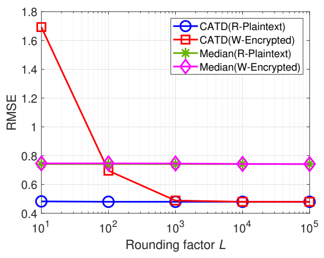

Fig. 6. Accuracy.

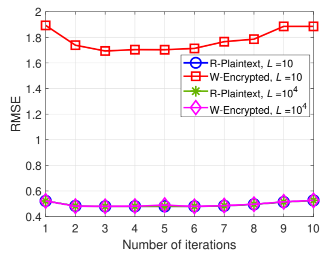

Fig. 7. Convergence.

generally come with enough reward budget for incentives. According to Amazon Mechanical Turk (AMT) policy, the requester needs to pay a 20% fee on the reward and bonus amount he pays workers and will be charged an additional 20% fee on the reward more if he has more than 10 tasks. Moreover, there is an additional fee for using the masters or premium qualifications. For more than 10 tasks with \$100 rewards, at least \$40 service fee is charged. In contrast, requesters in BRPC only need to pay the transaction fee independent of the reward amount, which saves the service cost and caters for practical large-scale crowdsourcing scenarios.

We then evaluate the gas cost generated for reward payment and user reputation update, respectively. As illustrated in Fig. 5, the gas cost scales linearly when the requester gives rewards to more workers (including rewarding verifiers and refunding deposit), and the reputation values of more workers are updated. This is in accord with that in Fig. 4 for similar reasons. In contrast to money transfer, it is more expensive to update the worker's reputation (\$7.3). Although the ETH price fluctuates a lot depending on the marketplace, we can see that the transaction fee is acceptable as the cost to ensure privacy, reliability, and fairness in the decentralized crowdsourcing.

Accuracy & Convergence. We report the impact of L on the accuracy of our encrypted CATD, which also affects the correctness of the verification result. As shown in Fig. 6, we observe that the difference in L does not affect the accuracy of CATD and Median at the requester (refer to R-Plaintext in the figure), in which CATD outperforms Median with a lower RMSE. On the encrypted data, CATD yields a slightly higher RMSE when L is small while Median presents a comparable RMSE with that in the plaintext. This is reasonable as the median of the encrypted sensory data is a one-time rounded number of the original median in the plaintext. In contrast, most fractional parts of sensory data and intermediate results

TABLE I
COMPARISON BETWEEN OUR BRPC AND EXISTING BLOCKCHAIN-BASED SCHEMES

Schemes	DP	Fine-grained DE	Comp. verifiability	TD method	Reputation model	Reputation UC	Dual fairness
[1]	✓	✗ (Integrity, H/L)	✗	Majority voting	Single role	$O(ml)$	✗
[10]	✓	✗ (Integrity)	✓	—	Single role	—	✗
[11], [15], [33]	✓	✗ (Integrity)	✗	—	—	—	✗
[13]	✗	✗ (Integrity, H/M/L)	✗	Cluster median	Single role	—	✗
[14]	✗	✗ (Integrity, [1,100])	✗	—	Two roles	$O(ml)$	✗
[17]	✓	✗ (Integrity)	✗	—	Single role	$O(ml)$	✗
[18]	✓	✓	✗	Weighted averaging	Single role	$O(ml)$	✗
[21]	✗	✓	✓	EM	—	—	✗
[23], [24]	✓	✗ (Integrity)	✓	Majority voting	—	—	✗
BRPC	✓	✓	✓	CATD	Three roles	$O(l)$	✓

1 DP and DE denote data privacy and data evaluation, respectively; Comp. denotes computation, EM and UC denote expectation maximization and update cost, respectively; H, M, and L denote three kinds of coarse-grained data quality, i.e., high, medium, and low quality.
2 ✓and ✗denote a realized and an unrealized property, respectively; — means that this property is unmentioned.
3 m denotes the number of tasks posted by a requester in a time slot; l denotes the number of time slots.

are removed for computations on integers in encrypted CATD (*i.e.*, multi-time rounding operations). However, when L is 1000 or higher, the worker-side encrypted CATD (refer to W-Encrypted) achieves nearly the same RMSE as that at the requester, which is still better than the Median. We can guarantee the accuracy of the worker-side verification as long as L is large enough. Hence, unless otherwise specified, we set L to 10^4 for Algorithm 1. For convergence performance, Fig. 7 presents the evolution of RMSE with the number of iterations in different CATD settings. As observed, the impact of L on the accuracy is consistent with the result in Fig. 6. Moreover, the algorithm can reach convergence with the least RMSE when the number of iterations is 3. Therefore, we can quickly derive the estimated truth and worker's weight in a few iterations without degrading the estimation accuracy.

User-side Verification Cost. We sample five sets of data with different number of workers and tasks, *i.e.*, $20\times 10, 20\times 20, 30\times 20, 40\times 20, 50\times 20$. Fig. 8 reports the worker-side verification time in three iterations. It is observed that the number of submitted data (112, 223, 309, 440, 528) in our random participation is more than half of the quantity of all-worker participation. Recall that each iteration consists of truth update and weight update, we denote the two key components: share decryption and share combination in weight update as SD and SC, respectively. Fig. 8 shows that the SD operation scales linearly as more data is involved and is much more expensive than others. This is reasonable as each worker needs to partially decrypt $\mathsf{E}_{pk_i}(\widetilde{d}_j^k)$, $\tau_j \in \mathcal{T}_{U_i}^l$, $U_k \in \mathcal{U}_i^l$, which is exactly the number of submitted data. Moreover, each partial decryption includes one expensive modular exponentiation operation. In contrast, the truth update cost scales with m, and SC only needs to perform a one-time share combination with one modular multiplication operation and one modular exponentiation operation. For the entire privacy-aware verification, the worker-side time cost is nearly 2600ms when 50 workers and 20 tasks are involved.

Furthermore, we plot the requester-side and worker-side communication cost in three iterations in Table II, which is measured by the data bytes sent by each entity under different numbers of workers and tasks in the verification phase. Recall that the requester only needs to broadcast the workers' weights to the network while each worker needs to broadcast the partially decrypted ciphertext for each $\widetilde{d}_j^k$ and the verification information. The worker-side communication cost is more expensive than that on the requester side. It entails 209307 bytes (≈204KB) for verification communication with 50 workers and 528 data submissions, which is insignificant in current network communication. Under a preset number of iterations, the associated communication complexity is $O(|\mathcal{U}_i^l|)$ and $O(|\mathcal{U}_i^l| \times |\mathcal{T}_{U_i}^l|)$, respectively. It is worth noting that the cost reported in Table II includes the bytes of the Ethereum address and signature information.

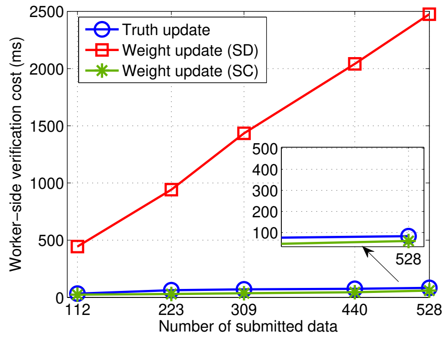

Fig. 8. Worker-side verification cost.

TABLE II
USER-SIDE COMMUNICATION COST IN JOINT VERIFICATION (BYTES)

Worker-task number (# of submissions)	20×10 (112)	20×20 (223)	30×20 (309)	40×20 (440)	50×20 (528)
Requester-side	245	245	325	405	485
Worker-side	45783	87255	122691	174255	209307

VII. RELATED WORK

A. Centralized Crowdsourcing Services

Existing crowdsourcing systems commonly rely on a centralized platform to provide the essential crowdsourcing services such as worker selection, task allocation, data evaluation, truth discovery [34], incentive provision [35], and reputation management. With privacy considerations, privacy-aware task allocation [36], [37], privacy-preserving and truthful incentive mechanisms [38], secure truth discovery protocols [39], and anonymous reputation assessment models [8] were separately investigated. However, they only work in a centralized setting with fully trusted or honest-but-curious server model.

B. Decentralized Crowdsourcing Services

Li *et al.* [1] first proposed a blockchain-based decentralized framework CrowdBC for general crowdsourcing systems. A framework FLUID was designed in [10] to provide transparent incentive mechanisms and enabled workers to share their profiles among different platforms. To mitigate the security threats from malicious participants, Gu *et al.* [11] proposed a blockchain-based participant management framework CrowdChain for fog-assisted crowdsensing. However, these aforementioned works only offer fundamental frameworks for blockchain-based crowdsourcing while not specifying some key components such as reliable data quality evaluation, accurate reputation assessment, and fair incentive provision.

For data quality and user reliability issues, An *et al.* [13] modeled a crowdsensing quality control method via blockchain. The data quality is controlled with consideration of matching degree and quality grading evaluation. WorkerRep [14] was developed to build trust on a crowdsourcing platform using blockchain. However, the sensory data is revealed to the evaluators. Edge computing was integrated with blockchain into crowdsourcing [18], [40], in which the requester was considered trusted for local reputation computation [18]. Although Zhang *et al.* [15] claimed that any worker can verify the correctness of evaluation and reward allocation without knowing the sensory data, their verification method via comparing two ciphertexts had flaws since two ciphertexts may be different even if they have the same plaintext. From the perspective of privacy and incentive, Wang *et. al* [21] designed a blockchain-based incentive mechanism which relied on the miners to verify the data quality. Aside from that, [25] focused on the location privacy in crowdsensing. To overcome data disclosure and identity breach, a private and anonymous decentralized crowdsourcing system ZebraLancer [23], [24] was proposed. Moreover, a hybrid blockchain crowdsourcing platform zkCrowd [33] was improved to relieve the performance bottleneck. Despite the customized privacy protection the fundamental functions of smart contracts and zero-knowledge proofs are still far away from successful and practical implementations.

VIII. Conclusion

In this paper, we present a blockchain-based reliable and privacy-aware system, named BRPC, for generic crowdsourcing. In contrast to prior work on crowdsourcing frameworks, BRPC further illustrates how the key functionalities are realized with curious and even malicious entities. We provide an efficient data quality evaluation method based on CATD and propose a privacy-aware computation verification protocol to detect incorrect evaluations of the malicious requesters. Moreover, we consider the different roles of a user and establish a multi-dimensional reputation model with a reliable computation and efficient update. Both social and financial fairness are achieved with our rewarding/punishment solution. Security analysis and experimental results over Ethereum validate the predefined goals and show its application potential.

References

[1] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J. Liu, Y. Xiang, and R. Deng, “CrowdBC: A blockchain-based decentralized framework for crowdsourcing,” *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 6, pp. 1251–1266, 2019.

[2] W. Feng, Z. Yan, H. Zhang, K. Zeng, Y. Xiao, and Y. T. Hou, “A survey on security, privacy, and trust in mobile crowdsourcing,” *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2971–2992, 2017.

[3] X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang, “Keep your promise: Mechanism design against free-riding and false-reporting in crowdsourcing,” *IEEE Internet Things J.*, vol. 2, no. 6, pp. 562–572, 2015.

[4] H. To, G. Ghinita, L. Fan, and C. Shahabi, “Differentially private location protection for worker datasets in spatial crowdsourcing,” *IEEE Trans. Mobile Comput.*, vol. 16, no. 4, pp. 934–949, 2016.

[5] A. Liu, W. Wang, S. Shang, Q. Li, and X. Zhang, “Efficient task assignment in spatial crowdsourcing with worker and task privacy protection,” *GeoInformatica*, vol. 22, no. 2, pp. 335–362, 2018.

[6] Y. Tong, Z. Zhou, Y. Zeng, L. Chen, and C. Shahabi, “Spatial crowdsourcing: a survey,” *The VLDB Journal*, vol. 29, no. 1, pp. 217–250, 2020.

[7] Z. Yan, W. Ding, V. Niemi, and A. V. Vasilakos, “Two schemes of privacy-preserving trust evaluation,” *Future Gener. Comput. Syst.*, vol. 62, pp. 175–189, 2016.

[8] X. O. Wang, W. Cheng, P. Mohapatra, and T. Abdelzaher, “Enabling reputation and trust in privacy-preserving mobile sensing,” *IEEE Trans. Mobile Comput.*, vol. 13, no. 12, pp. 2777–2790, 2014.

[9] D. Wu, S. Si, S. Wu, and R. Wang, “Dynamic trust relationships aware data privacy protection in mobile crowd-sensing,” *IEEE Internet Things J.*, vol. 5, no. 4, pp. 2958–2970, 2017.

[10] S. Han, Z. Xu, Y. Zeng, and L. Chen, “Fluid: A blockchain based framework for crowdsourcing,” in *Proc. ACM SIGMOD*. ACM, 2019, pp. 1921–1924.

[11] X. Gu, J. Peng, W. Yu, Y. Cheng, F. Jiang, X. Zhang, Z. Huang, and L. Cai, “Using blockchain to enhance the security of fog-assisted crowdsensing systems,” in *Proc. ISIE*. IEEE, 2019, pp. 1859–1864.

[12] C. Lin, D. He, S. Zeadally, N. Kumar, and K. R. Choo, “SecBCS: a secure and privacy-preserving blockchain-based crowdsourcing system,” *Sci. China Inf. Sci.*, vol. 63, no. 3, pp. 1–14, 2020.

[13] J. An, D. Liang, X. Gui, H. Yang, R. Gui, and X. He, “Crowdsensing quality control and grading evaluation based on a two-consensus blockchain,” *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4711–4718, 2018.

[14] G. K. Bhatia, P. Kumaraguru, A. Dubey, A. B. Buduru, and V. Kaulgud, “WorkerRep: Building trust on crowdsourcing platform using blockchain,” Ph.D. dissertation, IIIT-Delhi, 2018.

[15] J. Zhang, W. Cui, J. Ma, and C. Yang, “Blockchain-based secure and fair crowdsourcing scheme,” *International J. Distrib. Sens. Netw.*, vol. 15, no. 7, pp. 1–15, 2019.

[16] J. Zou, B. Ye, L. Qu, Y. Wang, M. A. Orgun, and L. Li, “A proof-of-trust consensus protocol for enhancing accountability in crowdsourcing services,” *IEEE Trans. Services Comput.*, vol. 12, no. 3, pp. 429–445, 2018.

[17] W. Feng and Z. Yan, “MCS-chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain,” *Future Gener. Comput. Syst.*, vol. 95, pp. 649–666, 2019.

[18] K. Zhao, S. Tang, B. Zhao, and Y. Wu, “Dynamic and privacy-preserving reputation management for blockchain-based mobile crowdsensing,” *IEEE Access*, vol. 7, pp. 74 694–74 710, 2019.

[19] Y. Tian, J. Yuan, and H. Song, “Secure and reliable decentralized truth discovery using blockchain,” in *Proc. IEEE CNS*. IEEE, 2019, pp. 1–8.

[20] C. Cai, Y. Zheng, A. Zhou, and C. Wang, “Building a secure knowledge marketplace over crowdsensed data streams,” *IEEE Trans. Dependable Secur. Comput.*, 2020, doi:10.1109/TDSC.2019.2958901.

[21] J. Wang, M. Li, Y. He, H. Li, K. Xiao, and C. Wang, “A blockchain based privacy-preserving incentive mechanism in crowdsensing applications,” *IEEE Access*, vol. 6, pp. 17 545–17 556, 2018.

[22] Y. He, H. Li, X. Cheng, Y. Liu, C. Yang, and L. Sun, "A blockchain based truthful incentive mechanism for distributed p2p applications," *IEEE Access*, vol. 6, pp. 27 324–27 335, 2018.
[23] Y. Lu, Q. Tang, and G. Wang, "Zebralancer: Private and anonymous crowdsourcing system atop open blockchain," in *Proc. IEEE ICDCS*. IEEE, 2018, pp. 853–865.
[24] ——, "ZebraLancer: Crowdsource knowledge atop open blockchain, privately and anonymously," *arXiv preprint arXiv:1803.01256*, pp. 1–17, 2018.
[25] M. Yang, T. Zhu, K. Liang, and R. H. Zhou, W.and Deng, "A blockchain-based location privacy-preserving crowdsensing system," *Future Gener. Comput. Syst.*, vol. 94, pp. 408–418, 2019.
[26] X. Xu, Q. Liu, X. Zhang, J. Zhang, L. Qi, and W. Dou, "A blockchain-powered crowdsourcing method with privacy preservation in mobile environment," *IEEE Trans. Computat. Social Syst.*, vol. 6, no. 6, pp. 1407–1419, 2019.
[27] Y. Wu, S. Tang, B. Zhao, and Z. Peng, "BPTM: Blockchain-based privacy-preserving task matching in crowdsourcing," *IEEE Access*, vol. 7, pp. 45 605–45 617, 2019.
[28] J. Huang, L. Kong, H. Dai, W. Ding, L. Cheng, G. Chen, X. Jin, and P. Zeng, "Blockchain based mobile crowd sensing in industrial systems," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6553–6562, 2020.
[29] Q. Li, Y. Li, J. Gao, L. Su, B. Zhao, M. Demirbas, W. Fan, and J. Han, "A confidence-aware approach for truth discovery on long-tail data," *VLDB J.*, vol. 8, no. 4, pp. 425–436, 2014.
[30] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. EUROCRYPT*. Springer, 1999, pp. 223–238.
[31] I. Damgård and M. Jurik, "A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system," in *Proc. PKC*. Springer, 2001, pp. 119–136.
[32] E. Barker, W. Burr, W. Polk, and M. Smid, *Recommendation for key management: Part 1: General*. National Institute of Standards and Technology, Technology Administration, 2006.
[33] S. Zhu, Z. Cai, H. Hu, Y. Li, and W. Li, "zkCrowd: a hybrid blockchain-based crowdsourcing platform," *IEEE Trans. Ind. Informat.*, vol. 16, no. 6, pp. 4196–4205, 2020.
[34] L. Jiang, X. Niu, J. Xu, D. Yang, and L. Xu, "Incentivizing the workers for truth discovery in crowdsourcing with copiers," in *Proc. ICDCS*. IEEE, 2019, pp. 1286–1295.
[35] H. Wang, S. Guo, J. Cao, and M. Guo, "MeLoDy: A long-term dynamic quality-aware incentive mechanism for crowdsourcing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 4, pp. 901–914, 2017.
[36] H. Wu, L. Wang, and G. Xue, "Privacy-aware task allocation and data aggregation in fog-assisted spatial crowdsourcing," *IEEE Trans. Netw. Sci. Eng.*, pp. 589–602, 2020.
[37] D. Yuan, Q. Li, G. Li, Q. Wang, and K. Ren, "Priradar: A privacy-preserving framework for spatial crowdsourcing," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 299–314, 2019.
[38] Z. Wang, J. Li, J. Hu, J. Ren, Z. Li, and Y. Li, "Towards privacy-preserving incentive for mobile crowdsensing under an untrusted platform," in *Proc. IEEE INFOCOM*. IEEE, 2019, pp. 2053–2061.
[39] Y. Zheng, H. Duan, and C. Wang, "Learning the truth privately and confidently: Encrypted confidence-aware truth discovery in mobile crowdsensing," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 10, pp. 2475–2489, 2018.
[40] J. Kang, R. Yu, X. Huang, M. Wu, S. Maharjan, S. Xie, and Y. Zhang, "Blockchain for secure and efficient data sharing in vehicular edge computing and networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4660–4670, 2018.

Haiqin Wu received her B.S. degree in Computer Science and Ph.D degree in Computer Application Technology from Jiangsu University in June 2014 and September 2019, respectively. She is currently a postdoctoral researcher in the Department of Computer Science, University of Copenhagen, Denmark. She was a visiting student in the School of Computing, Informatics, and Decision Systems Engineering at Arizona State University, US. Her research interests include data security and privacy protection, mobile crowdsensing, and blockchain application.

Boris Düdder received his Diploma degree (2008) and the Ph.D. degree (2014) in Computer Science at the Technical University of Dortmund, Dortmund, Germany. He is associate professor in software engineering in the Department of Computer Science at the University of Copenhagen, Copenhagen, Denmark, where he is head of the research group Formal Methods & Software Engineering. His research interests are in formal methods in software engineering and secure and reliable distributed systems. He has published over 30 technical papers at premium international journals and conferences. Dr. Düdder has served as TPC and PC member of many ACM conferences. His background is both academic and industry, e.g., Fraunhofer, Germany, and Microsoft, USA.

Liangmin Wang received his B.S. degree in Computational Mathematics in Jilin University, Changchun, China, and the PhD degree in Cryptology from Xidian University, Xi'an, China. He is a full professor in the School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China. He has been honored as a "Wan-Jiang Scholar" of Anhui Province since Nov. 2013. His research interests include data security & privacy, and blockchain. He has published over 60 technical papers at premium international journals and conferences, like IEEE/ACM TON, INFOCOM, TII, TITS, and Internet of Things Journal. Dr WANG has served as a TPC member of many IEEE conferences, such as IEEE ICC, IEEE HPCC, IEEE TrustCOM. Now he is an associate editor of Security and Communication Networks, a member of IEEE, ACM, and a senior member of Chinese Computer Federation.

Shipu Sun received the B.S. degree in electrical engineering from Zhengzhou University, Zhengzhou, China, in 2015. He is currently pursuing the M.S. degree in the School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, China. His current research interests include cryptography, network and data security, and blockchain.

Guoliang Xue is a professor of Computer Science and Engineering at Arizona State University. He received the Ph.D degree in Computer Science from the University of Minnesota in 1991. His research interests span the areas of Quality of Service provisioning, network security and privacy, crowdsourcing and network economics, RFID systems and Internet of Things, smart city and smart grids. He has published over 280 papers in these areas, many of which in top conferences such as INFOCOM, MOBICOM, NDSS and top journals such as IEEE/ACM Transactions on Networking, IEEE Journal on Selected Areas in Communications, and IEEE Transactions on Mobile Computing. He has received the IEEE Communications Society William R. Bennett Prize in 2019 (best paper award for IEEE/ACM TON and IEEE TNSM in the previous three years). He was a keynote speaker at IEEE LCN'2011 and ICNC'2014. He was a TPC Co-Chair of IEEE INFOCOM'2010 and a General Co-Chair of IEEE CNS'2014. He has served on the TPC of many conferences, including ACM CCS, ACM MOBIHOC, IEEE ICNP, and IEEE INFOCOM. He served on the editorial board of IEEE/ACM Transactions on Networking. He served as the Area Editor of IEEE Transactions on Wireless Communications, overseeing 13 editors in the Wireless Networking area. He is the Steering Committee Chair of IEEE INFOCOM.