A Blockchain-based Crowdsourced Task Assessment Framework using Smart Contract

Linta Islam[1], Syada Tasmia Alvi[2], Mafizur Rahman[3], Ayesha Aziz Prova[4],
Md. Nazmul Hossain[5], Jannatul Ferdous Sorna[6], Mohammed Nasir Uddin[7]
Dept. of CSE, Jagannath University, Dhaka, Bangladesh[1,2,5,7]
Dept. of CSE, East West University, Dhaka, Bangladesh[3,6]
Dept. of CSE, Central Women's University, Dhaka, Bangladesh[4]

***Abstract*—In today's world, crowdsourcing is a highly rising paradigm where mass people are engaged in solving a problem. Though this system has a lot of advantages, yet people are not interested in working on this platform. Thus, we survey people to find out the constraints of this platform and the main reason behind their unwillingness. 59% of people think that security and privacy is the major challenge of a crowdsourcing platform. Therefore, we propose a blockchain-based crowdsourced system which can provide security and privacy to the user's information. We also have used a smart contract to verify the task so that the users get the exact output that they have wanted. We implemented our system and compared the performance with the existing systems. Our proposed approach outperforms the current methods in terms of cost and properties.**

***Keywords*—Blockchain; crowdsourcing; task allocation; smart contract**

I. Introduction

Crowdsourcing is a marketing platform where a large task is distributed into a collection of small pieces and providing those pieces into a large group of workers [1]. It is one kind of sourcing model where individuals or organizations get services, goods, and ideas from a huge group of participants. Therefore, the workers get some incentives based on their respective work [2]. Crowdsourcing can be done by using smartphones, wearables, computers, etc. devices. Among them, mobile crowdsourcing is one of the powerful approach that can consolidate the wisdom of humans into mobile computations for solving the problem [3]. For collecting data ambient light, proximity, location, movement, noise, etc. sensors are needed and every smartphone has now those sensors like GPS, accelerometer, microphone, etc. Therefore, mobile crowdsourcing can be utilized for collecting unstructured data from heterogeneous sources, industrial applications as well as for subjective assessments with the help of mobile sensors [4].

Blockchain is known as a distributed database system that can store financial transactional records of the people in a linear set of blocks [5]. Each of the blocks is linked with the previous blocks. As it is a decentralized system, therefore there will be no central administrator or third party organization for necessary work. The participants can decide as there is no intermediary in the system [6]. Users have the capability of controlling their transactions. They can delete, rewrite, or change their information easily. As there will be a unique id for each user, hence there is no chance of data losing or hacking [7]. Blockchain has a crypto contract or a smart contract script that included unique addresses, executable functions, and variables. Therefore, there is a controlling over the digital currency transactions. It is one kind of computer code that runs on the blockchain system as an agreement among two people. Based on the agreement the transactions will happen in the smart contract. Thus, it will be added to the public database which can't be changed. Then the blockchain will process the transactions that happen in the smart contract.

As crowdsourcing is getting popular day by day, still the users have to face numerous problems like resource limitation, privacy and security, spatio-temporal issues, etc. [8]. The main problem is there is no confidentiality [9]. Hence, crowdsourcing with blockchain can be a useful system to resolve those problems fruitfully. As the smart contract will increase the sustainability of the information with a data-centric approach [10]. It can endure the people accountable not only for front end performance but also for back end sustainability. Hence, the user's privacy will be maintained and they can get all the facilities in a useful manner.

Our work aims to implement a crowdsourced task management model employing blockchain to provide security and privacy of the participants along with the task. We have used the smart contract for fairness of the task assessment. The participants can also submit rating points, which helps the coordinator to select the task contributors. As we are storing the task information in the blockchain network, it is difficult for malevolent users to modify any stored information.

The remainder of the paper is organized as follows. Section II overviews the related works. Section III illustrates the motivating scenario of this work. Section IV presents an overview of the system model and defines the problem. Section V describes the details of the proposed blockchain-based crowdsourcing framework. Sections VI and VII evaluates the approach by property analysis and shows the experiment results. Section VIII concludes the paper and highlights some future work.

II. Related Works

The insufficient amount of research has been done in the field of the blockchain-based crowdsourced system. Feng and Yan [11] presented a distributed blockchain-based system called MCS-Chain that uses consensus mechanism for block generation. This system also has less computational overhead, and it solves the centralization problems. CrowdBC [12] is a decentralized crowdsourcing framework which is developed using the blockchain network. This framework can provide a

user's privacy with a low service fee. CrowdSFL [13] used a re-encryption algorithm to preserve the privacy of the users. The authors also used blockchain-based crowdsourcing in this model. Lu et al. proposed a decentralized crowdsourcing system named ZebraLancer [14]. This system can also provide confidentiality and anonymity alike CrowdSFL. BPTM [15] is a blockchain-based task matching system that provides data confidentiality and anonymity. It also offers a secure and reliable task matching protocol using a smart contract. However, each of these models has several limitations. Some of them did not mention reputation values like other models, and others did not provide any security analysis. Thus, we tried to mitigate the problems of the existing works in our proposed model.

III. Motivation Scenario

Before implementing this system, we asked 362 students of Jagannath University and East-West University that whether they are comfortable to use online task participation system and whether they are facing any difficulties while using them. They also have been asked to note down the issues that need to be solved to provide an efficient task participation framework. In this survey, around 59% of people chose the security and privacy of their data as a significant issue. Less than 25% of people believe that the quality of the system and lack of knowledge to use the system as the latter issue. Several people consider that limited resource and location of the task participation is another issue for their unwillingness to participate in this task provisioning. A few numbers of people also believe that online task participation systems are less beneficial than onsite ones. The survey result is represented by a pie chart in Fig. 1.

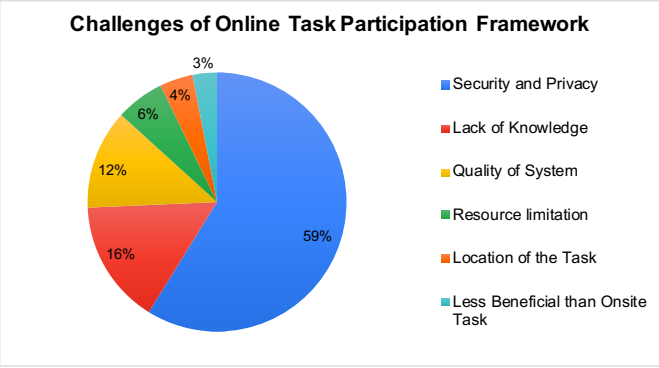

Fig. 1. Challenges of Online Task Participation Framework.

IV. System Overview

This section represents a novel formal model for blockchain based crowdsourcing. In this framework, the crowdsourcing tasks could be irregular and dynamic based on both location and time. We discuss the fundamental concepts of the proposed system and illustrates the workflow diagram briefly.

A. Components of the System Model

In this section, we describe the components of our proposed system. As shown in Fig. 2, the system is composed of four basic components. These are user block, crowdsourced marketplace, smart contract and blockchain. We will discuss each components thoroughly in this section.

Definition 1. ***User Block:*** *A mobile crowdsourcing system is consists of several types of participants. In our previous work, we have classfied the participants into three categories namely contributors, task requesters, and coordinators [16]. However, in this research work, we have included all the users in the user block of our system model. The users can reside in this block if they own any smart devices, including smartphone, smartwatch, notebooks. All of the users will have to take the authorization power to use this system. This procedure can be accomplished efficiently by registering themselves in the system. We denoted a user by* u *which is a tuple of* $\langle uid, loc_u, t_u, sts_u, urts_u, Q_u \rangle$ *where*

- uid *is a unique user ID*
- loc_u *is the latest recorded location of the user* u
- t_u *is the latest time at which the user* u *participated in any crowdsourced task*
- sts_u *is the current availability status of the user* u
- $urts_u$ *is the registration time of the user* u
- SKL_u *is a tuple* $\langle skl_1, skl_2, \ldots, skl_n \rangle$*, where each* skl_i *denotes a skill of the user* u *(e.g. video editing, photo tagging, content writing, website development).*
- Q_u *is a tuple* $\langle q_1, q_2, \ldots, q_n \rangle$*, where each* q_i *denotes a QoS property of the user* u *(e.g. bandwidth, reputation value, coverage distance, latency).*

In this user block, there can be two types of users according to their role in this proposed system, namely, task requester and contributor. In our previous paper [17], we have defined these two types of users briefly. The user who posts a task request to the crowdsourced marketplace is called task requester. Any user who will accomplish the task by fulfilling all the requirements is called contributor. Thus, in the crowdsourced marketplace, the task requester act as a buyer, and the contributor act as a seller. Their analytical definition is as follows:

Definition 2. ***Task Requester:*** *When a user* u *send a crowdsourced task request* crt *to the crowdsourced marketplace, then the marketplace consider that user as a task requester* $treq$ *which is a tuple of* $\langle treqid, uid_{treq}, nocrt, rp_{treq} \rangle$ *where*

- $treqid$ *is a unique task requester ID*
- uid_{treq} *is a unique user ID of the task requester* $treq$
- $nocrt_{treq}$ *is the total number of tasks requested by task requester* $treq$
- rp_{treq} *is the current rating points of the task requester* $treq$

The information mentioned above will be stored at the blockchain only for further references. This way of data storing will reduce the amount of redundant data in the system.

Definition 3. ***Contributor:*** *When a user* u *is eligible to perform a crowdsourced task request* crt *to the crowdsourced*

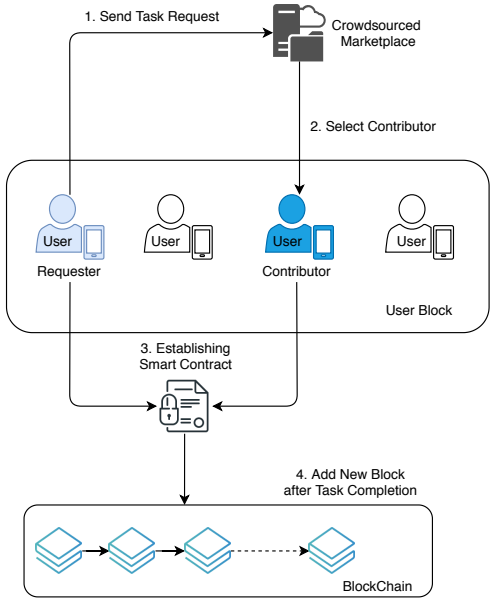

Fig. 2. System Model.

marketplace, then the marketplace consider that user as a contributor con which is a tuple of $\langle conid, uid_{con}, nocrt_{con}, tearn_{con}, rp_{con}\rangle$, *where*

- $conid$ *is a unique contributor ID*
- uid_{con} *is a unique user ID of the contributor* con
- $nocrt_{con}$ *is the total number of sub-tasks performed by contributor* con
- $tearn_{con}$ *is the total earning of the contributor* con
- rp_{con} *is the current rating points of the contributor* con

The aforementioned information will also be stored at the blockchain solely for additional references which will reduce the quantity of redundant data in the system.

Definition 4. ***Request:*** *When a task requester* $treq$ *wants to perform a task and requires more workers to achieve that, the task requester then demands one or more contributors by posting a task request to the crowdsourced marketplace. This request is called the crowdsourced task request* ctr *which is a tuple of* $\langle crtid, loc_{ctr}, t_{ctr}, nocon_{ctr}, cost_{ctr}, des_{ctr}, ty_{ctr}, SUBT_{ctr}, SUBDES_{ctr}, Q_{ctr}\rangle$ *where*

- $ctrid$ *is a unique crowdsourced task request ID*
- loc_{ctr} *is the location where the task* ctr *needed to be performed*
- t_{ctr} *is the last time within which the task* ctr *should be completed*
- $nocon_{ctr}$ *is the number of contributor needed to complete the task* ctr
- $cost_{ctr}$ *is the total cost of the task* ctr *that a task requester is ready to pay*
- ty_{ctr} *is the type of the task* ctr
- des_{ctr} *is the text description of the task* ctr
- $SUBT_{ctr}$ *is a tuple* $\langle subt_1, subt_2, \ldots, subt_n\rangle$, *where each* $subt_{ctr}$ *denotes the type for each subtask of each component which is constitutes of* ty_{ctr}. *Task requester can provide this information while requesting for a task.*
- $SUBDES_{ctr}$ *is a tuple* $\langle subdes_1, subdes_2, \ldots, subdes_n\rangle$, *where each* $subdes_{ctr}$ *denotes the text description for each subtask of each component of* ty_{ctr}. *Task requester can provide this information if he/she added the sub-task types while requesting for a task.*
- Q_{ctr} *is a tuple* $\langle q_1, q_2, \ldots, q_n\rangle$, *where each* q_{ctr} *denotes the minimum requirement for each QoS property of the selected contributor to do the task* ctr.

Definition 5. ***Crowdsourced Marketplace:*** *Crowdsourcing marketplace is a platform which brings the set of contributors* CON *and the set of task requesters* $TREQ$ *under the same network and connects them to each other through a task. There might be several tasks which are easier to do if it is distributed among other people. Crowdsourced marketplace receives such kind of task requests* CTR *and finds the suitable contributors who can accomplish the tasks. In this research, we have used an online cloud-based platform as our crowdsourced marketplace.*

Definition 6. ***Blockchain:*** *Blockchain is a distributed transaction ledger which can store a collection of blocks through establishing a chain. In our proposed model, a block stores information related to an accomplished task and adds it to the blockchain network. Each block* bc_n *is a tuple of* $\langle bcid_n, bcid_{n-1}, bct_n, ctr, uid_{treq}, uid_{CON}\rangle$ *where*

- $bcid_n$ *is the unique block ID. Here, block ID is the hash value of the block.*
- $bcid_{n-1}$ *is the previous block's ID. Here, the previous block's ID is the hash value of the previous block.*
- bct_n *is the time when block* bc_n *is created.*
- ctr *is the crowdsourced task request.*
- uid_{treq} *is the unique user ID of the task requester who posted the crowdsourced task request.*
- uid_{CON} *is the unique user ID of the set of contributors that participated in the crowdsourced task.*

Definition 7. ***Smart Contract:*** *A smart contract is a contract which contains previously defined terms and conditions written by the blockchain network. In our system, the smart contract* sc *is a tuple of* $\langle scid_{ctr}, ctr, uid_{treq}, uid_{CON}, scp, scac_{sc}\rangle$ *where*

- $scid_{ctr}$ *is the unique smart contract id for the task* ctr
- ctr *is the crowdsourced task request*
- uid_{treq} *is the unique user ID of the task requester who posted the crowdsourced task request*
- uid_{CON} *is the unique user ID of the set of contributors that participated in the crowdsourced task*

- *scp is the protocol or rules that is used to create the smart contract*
- *$scac_{sc}$ is the account balance of the smart contract sc.*

B. Work Flow of the System Model

The workflow of our proposed system is presented in Fig. 3. This system performs the following actions are illustrated subsequently:

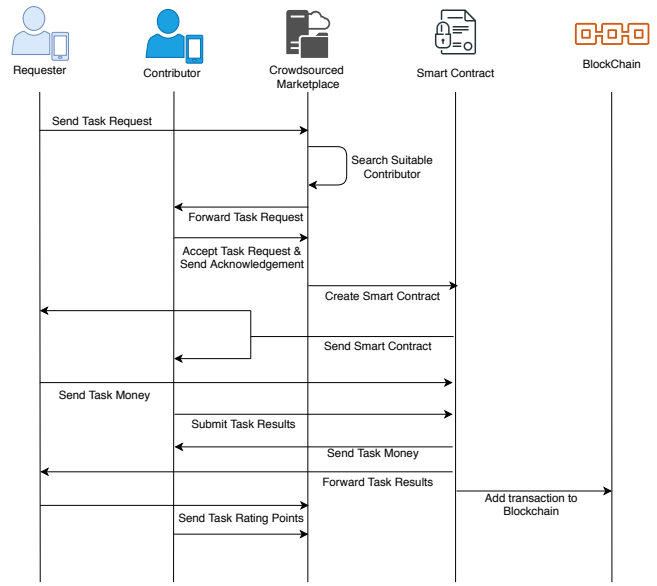

Fig. 3. Work Flow Diagram of Blockchain-based Mobile Crowdsourcing Framework.

Step 1: The task requester $treq$ posts a task request ctr in the crowdsourced marketplace with his intended budget and minimum requirements of the contributors to do that specific task.

Step 2: The crowdsourced marketplace explores the set of contributors CON and retrieves eligible contributors according to the task request ctr. Later, the request is forwarded to the selected contributors only.

Step 3: When a contributor con agrees to do the task, then he/she conveys an acknowledgment message to the marketplace.

Step 4: The crowdsource marketplace triggers the smart contract upon receiving the acknowledgment from the contributor con. The smart contract will be designed based on the task request ctr.

Step 5: The smart contract is sent to the task requester $treq$ and the contributor con. After receiving the smart contract, the contributor con starts working on the task ctr, and the task requester $treq$ transfers the task money.

Step 6: After finishing the task ctr, the contributor con forwards the output to the smart contract. If the output satisfies the task requirements, the smart contract transfers the task money to the contributor con and forward the output to the task requester $treq$.

Step 7: The task requester $treq$ and the contributor con will give a rating point to each other and submit the rating points to the crowdsourced marketplace. Eventually, the crowdsource marketplace will add the information related to the task ctr, task requester $treq$ and contributor con to the blockchain by creating a block bc as a transaction for further references.

V. Proposed Model

In this section, we propose a novel blockchain-based crowdsourcing framework using smart contract. Our crowdsourced model is segmented into seven steps. We explain each step in each subsection.

A. User Registration to the System

The crowdsourced marketplace mainly permits the authorized user to operate the system. To post a task or to receive a task request, a user must have a smart device through which he/she can register himself/herself to the system. After completion of the registration process, the crowdsourced marketplace will provide a unique user identity (User ID) to the user. However, there might be several malicious users who can exploit the system by creating fake user identities. This kind of intrusion is called the Sybil attack [18].

The most common way to prevent this attack is to take the registration fee from the user while registering into the system. Nevertheless, this will increase the cost of the user, and they might be unwilling to pay additionally for the registration [19]. Thus, the user has to submit any identification card (e.g. Student ID, Office ID, National ID, Passport, Driving Licence) to prove his/her identity. In this way, we can prevent sybil attack [20].

Fig. 4 illustrates the process of user registration in the system. Initially, the user will submit his/her profile information including his/her identification card and smart device information to the crowdsourced marketplace. The crowdsourced marketplace will forward the information to the identity verification centre for user authentication. If the user is legitimate, then the crowdsourced marketplace will send a unique user ID to the user, and it will add the user in the particular position in the user block so that the user can gain a remarkable advantage of the system. The users are stored in a sorted form according to their smart device information in the user block. This sorting will help the marketplace to choose contributors from the user block.

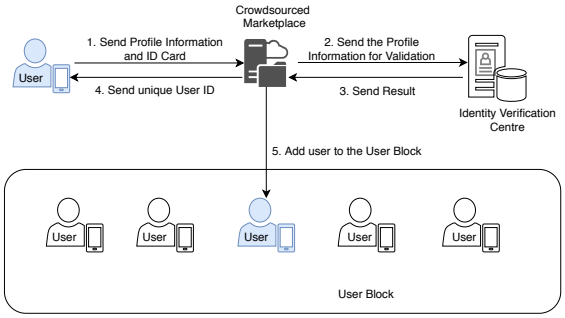

Fig. 4. User Registration Process.

B. Task Request Placement

The task requesters can place a task request at the crowdsourced marketplace irrespective of time and place. However, to post a task, he/she must specify the task request precisely to such an extent that the marketplace finds suitable contributors for that task without any issue. In the previous section, we have formally defined a task request. The requester must provide the following information while posting a task request.

- **Task Location:** The task requester should clearly define the area where the task should be carried out. The task location can be an area, a city or evan a country. If the task requester keeps this field blank, then the task can be performed at any places.
- **Task Finishing Time:** One of the essential criteria of a task request is the finishing time of the task. The task requester must provide a deadline within which the contributor should complete the task.
- **Number of Contributors:** There are various tasks where we need only one person to complete it. For example, to write an article for a blog needs only one person. Additionally, there are several tasks where we need many people to finish it. In particular, if we want to collect data from several places of Dhaka city, we need more people. The task requesters should specify the number of contributors to perform the task.
- **Budget:** The task requesters also necessitate estimating the cost needed to execute any task. Moreover, he/she have to pay to use the services of the marketplace, smart contract and the blockchain network. The service charge of the marketplace, contract and the network is constant. Thus, the task requester has to make a budget considering all these costs. The budget of the task can be defined using equation 1:

$$cost_{ctr} = cost_{con} + cost_{mkt} + cost_{sc} + cost_{bc} \quad (1)$$

  Here, $cost_{ctr}$ is the budget of the task which is the total cost to execute the task. The budget includes four types of cost:
  - $cost_{con}$ is the cost of the contributor. This amount will be transferred to the contributor after the successful task completion.
  - $cost_{mkt}$ is the cost of the crowdsourced marketplace. The crowdsourced marketplace will charge a few amount to maintain the task placement, task distribution and more.
  - $cost_{sc}$ is the cost of smart contract. To maintain the feasibility of the smart contract, a fixed amount will be cut from the task budget.
  - $cost_{bc}$ is the cost of blockchain. A certain amount will be fixed for the maintenance of the blockchain network.
- **Task Type:** Different types of task can be implemented using crowdsourcing. The example includes photo tagging, video editing, content writing, website or application development. The most popular task, according to the paper [21], is content creation. This task type selection will be helpful for the marketplace to find a suitable contributor who has expertise in this field.
- **Task Description:** The task requester also should describe the task according to the context. For example, a task requester needs an article writer for his/her blog. This information is not sufficient to understand the task. Thus, the requester must explain the task clearly by setting the article types, word limits, font style and size and more. By reading the task description, the contributors will understand the task thoroughly.
- **Sub-Task Type:** Occasionally, a complex crowdsourced task is consists of several simple subtasks. For example, a task requester requests for a website for his/her organisation. This website development task is consists of several subtasks such as user interface designing, database designing, connecting interface with the database, testing the system and so on [22]. The task requester may choose the types of subtask for a better product.
- **Sub-Task Description:** The task requester also may provide a text description for the betterment of their product. If a task requester asks for a website for his/her company, he/she might address the specifications for several sub-tasks. For example, the user interface should contain a plethora of blue colour; the database should be built using MySQL and so on. This description will help the contributor to study and implement the small details provided by the task requester.
- **Minimum Requirement of the Contributor:** The task requester can also outline the requirements of the contributor that he/she wants to perform the task. For example, he/she can prefer a contributor who has a mobile device with at least 4 GB of RAM, have a rating point more than four and completed at least ten tasks in the platform. Thus, the marketplace promptly finds a contributor as the requester has already minimized the search list.

Algorithm 1 illustrates the process of task request placement in the crowdsource marketplace. The above-explained elements are the input of this algorithm. The output is a successful task request which will be posted in the marketplace with the unique ID $ctrid$. In the beginning, we investigate whether the task requester is a posting for the first time or not. If he/she is a new task requester, then the system will provide them with a unique task requester ID $treqid$ using createTaskRequester() function. This function creates a new task requester's ID. The task requester $treq$ is assigned a threshold value θ_{rp} as rating point. (Line 1 $\sim$ 3). Next, the algorithm will check whether the rating point of the $treq$ is greater than or equal to the threshold value θ_{rp} (Line 4). Otherwise, $treq$ will be ineligible to post a task in the marketplace (Line 15 $\sim$ 16). Both of the threshold values θ_{rp} and θ_{cost} will be decided by the marketplace based on the task type ty. Then, we will compare the budget of the task. If the budget of the task $cost_{ctr}$ is higher than the total cost of the task, then it will proceed in the next step. Contrarily, it will return an insufficient budget error message. In the sixth line, we calculate the number of available contributors

Algorithm 1: Task Request Placement

Input: Input for Task Request, In = { $treqid$, rp_{treq} loc_{ctr}, t_{ctr}, $nocon_{ctr}$, $cost_{ctr}$, des_{ctr}, ty_{ctr}, Q_{ctr}}
Output: $ctrid$

1 **if** *newTaskRequester()* **then**
2 $treqid$ = createTaskRequester()
3 $rp_{treq} = \theta_{rp}$
4 **if** $In.rp_{treq} >= \theta_{rp}$ **then**
5 **if** $In.cost_{ctr} > (\theta_{cost} + cost_{mkt} + cost_{sc} + cost_{bc})$ **then**
6 $availCon$ = $checkAvailableContibutors()$;
7 **if** $In.nocon <= availCon$ **then**
8 Create task request ctr along with a unique ID $ctrid$
9 Update CTR
10 $nocrt++$
11 **else**
12 **return** "No Available Contributors" message
13 **else**
14 **return** "Insufficient Task Budget" message
15 **else**
16 **return** "Ineligible for Task Placement" message
17 **return** "Successful Task Placement" message with $ctrid$

who are interested in participating in the crowdsourcing task. We use checkAvailableContibutors() functions and store the result in $availCon$. Then, we compute if the number of contributors required to perform the task is less than $availCon$. If affirmative, then a task request ID $ctrid$ is created, and the task is posted to the marketplace. The set of all the task CTR is updated, and the number of tasks posted by the task requester $nocrt$ is incremented (Line 7 $\sim$ 10). Contrarily, the task request is refused due to the lack of contributors (Line 11 $\sim$ 12).

C. Task Request Arrival and Distribution

Suppose there is $noctr$ number of task requests posted by $notreq$ number of task requesters. To distribute the task request among the contributors, we first divide the time period TP to i slots as TP = {t_1, t_2, t_3, ..., t_i}. In each slot, the task requests have been posted randomly. Thus, we use the Poisson process [23] with the arrival rate λ as the task types are heterogeneous. If the crowdsourced marketplace starts searching suitable contributors for a task immediately after the task posting, it will increase the cost as well as decrease the utility of the marketplace. Therefore, the crowdsourced marketplace distribute the tasks among the contributors using the Algorithm 2.

The marketplace will push all the incoming task requests crt in the task buffer $tbfr_{ty}$ based on task types. The buffer $tbfr_{ty}$ will be used as a input of the Algorithm 2 and the $taskReleaseFlag$ is the output of this algorithm. The $taskReleaseFlag$ is initialized to false for all $tbfr_{ty}$. By applying Algorithm 2, the marketplace will wait for λ amount of time to distribute the task (Line 8 $\sim$ 9). After the completion of waiting time wt_{tbfr}, the marketplace starts searching for the contributor to assign all the task requests (Line 6 $\sim$ 8). If $tbfr_{ty}$ is full before the completion of waiting period, the contributor starts releasing the task for distribution (Line 3 $\sim$ 5).

Algorithm 2: Task Request Distribution

Input: {$crtid$, λ, wt_{tbfr}, $tbfr_{ty}$}
Output: $taskReleaseFlag_{tbfr}$

1 **foreach** $crtid$ *in* $tbfr_{ty}$ **do**
2 $taskReleaseFlag_{tbfr}$ = false
3 **if** $tbfr_{ty}$ *is full* **then**
4 $taskReleaseFlag_{tbfr}$ = true
5 **return** $taskReleaseFlag_{tbfr}$
6 **else if** wt_{tbfr} ¿= λ **then**
7 $taskReleaseFlag_{tbfr}$ = true
8 **return** $taskReleaseFlag_{tbfr}$
9 **else**
10 waitForRelease()

D. Contributor Selection

When the $taskReleaseFlag$ is true, the system proceeds to the next step and initiates Algorithm 3. In our previous work [17], we used the reverse-auction method to select the contributor. This method was used to find whether the users are interested in performing any task. In this proposed model, we used a flag to find the availability status of each user. The availability status sts of a user will be true by default which means he/she is willing to perform a task. If a user is already assigned any task, his/her sts will be false. A user can also change his/her sts from true to false manually for a definite amount of time, according to his/her choice if he/she is not interested in performing any task. The function $findAvailableUser()$ is used to determine the list of interested users $availableuser$.

To become a contributor, users have to pass through several requirements. At first, we verify whether their current location is in the task location or not (Line 5). Then, we examine whether the task type is similar to their skills (Line 6). Finally, we review each QoS parameters (Line 8 $\sim$ 10). If the user passes all of these requirements, then the algorithm chooses him as a contributor. This process is continued until we find all the contributors needed to perform the task (Line 11 $\sim$ 14). Ultimately, the algorithm returns the list of contributors to the marketplace (Line 15 $\sim$ 16). If the number of recognised contributors is less than the required number of contributors, then we call for a reverse auction which was proposed in [17] and include all the contributors in the list (Line 17 $\sim$ 21). This method will help the marketplace to collect all the eligible contributors to perform the task.

E. Smart Contract Implementation

The smart contract sc is an agreement between the task requester $treq$ and the contributor con. The contract runs automatically according to predefined rules and protocols. The crowdsourced marketplace will initiate the smart contract for both task requester and contributor. The Algorithm 4 depicts the specific actions of the smart contract.

Firstly, the crowdsourced market place will accept the task money $taskMoney$ from the task requester $treqid$ based on

Algorithm 3: Contributor Selection

Input: $\{crt, tbfr, U\}$
Output: $conlist$

```
1  count = 0
2  need = 0
3  foreach crtid in tbfr_ty do
4      availableuser = findAvailableUser(U)
5      foreach user u in availableuser do
6          if loc_u in loc_ctr then
7              if ty_crt in SKL_u then
8                  if each q_u ¿= q_ctr then
9                      conlist_i = u
10                     count++
11                     if count ¿= nocon_ctr then
12                         break
13                     else
14                         continue
15     if count ¿= nocon_ctr then
16         return conlist
17     else
18         need = nocon_ctr - count
19         conAuction = reverseAuction(crt, need, θ_t)
20         append conAuction in conlist
21         return conlist
```

Algorithm 4: Smart Contract Implementation

Input: $\{crt, treqid, con\}$
Output: $taskResult$

```
1  taskMoney = ReceiveMoney(treqid, cost_ctr)
2  conMoney = taskMoney - (cost_mkt + cost_sc + cost_bc)
3  SignContract(treqid, con, ctrid, conMoney)
4  taskResult = ReceiveTaskResult(con)
5  if ResultValidation(taskResult) then
6      if currentTime < t_ctr then
7          TransferMoney(con, conMoney)
8          return taskResult
9      else
10         TransferMoney(treqid, taskMoney)
11         return "Task Incomplete" message
12 else
13     TransferMoney(treqid, taskMoney)
14     return "Task Invalid" message
```

the task request ctr (Line 1). The cost of the marketplace $cost_{mkt}$, smart contract $cost_{sc}$ and blockchain $cost_{bc}$ will be deducted before the contract creation (Line 2). Then, the marketplace will create a smart contract for the budget of $conMoney$ based on the predefined protocols. The task requester $treqid$ and the contributor con will sign the contract for the task completion using $SignContract$ function (Line 3). After receiving the task result $taskResult$ from the contributor, the contract will verify the result. If the result is valid and submitted on due time, then the smart contract will transfer the money to the contributor (Line 4 $\sim$ 8). Otherwise, the money will be transferred to the task requester (Line 9 $\sim$ 14).

F. Rating Point Computation

After receiving the task result, the task requester bestows a rating point to the contributors based on their satisfaction level. On the other hand, the requester also gets a rating point from the contributors. These rating points will define the reputation of the users. There are several ways to compute reputation scores including Mean-based reputation [24], Bayesian reputation [25], Fuzzy reputation [26] and so on. We modify the following equation to calculate the reputation score of the users which has been proposed in [27].

$$rp = \sum_{i=1}^{n} \sum_{j=1}^{qos} tr_i \times tw_i \tag{2}$$

We calculate the reputation score rp in equation 2 using weighted average for each rating points. Here, tr_i is the task rating for each task, tw_j is the task weight for each QoS (e.g. bandwidth, reputation value, coverage distance, latency).

G. Addition of Transaction Block

After computing the rating points, the crowdsourced marketplace will add the task details in the blockchain. We will not use any miner competition to add any transaction to the blockchain. The miner competition executes very time-consuming calculations. This miner competition also wastes a lot of resources where the mobile devices have resource constraints [28]. Thus, the marketplace will choose a miner using the miner selection algorithm proposed in [29]. The transaction block is illustrated in the Fig. 5.

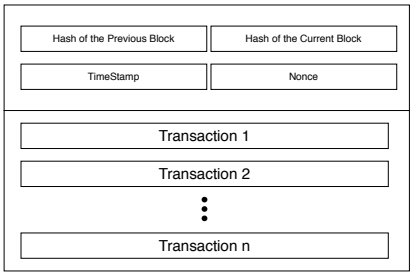

Fig. 5. Structure of Transaction Block.

Each transaction block in the blockchain is consists of two parts: block header and block body. The block header contains the hash value of the current and previous block, timestamp and nonce. Each block is encrypted using a hash algorithm, and the hash is stored in the block header. The block header also contains the hash of the previous block for security purpose. The timestamp is applied to retain the time for each transaction. A nonce is a number which is employed for hashing in the blockchain. The block body comprises of a list of transactions. Each transaction is consists of task requester's information, contributor's information, and task request's information.

VI. Security Analysis

A blockchain-based model should address several security properties. Our proposed framework also provides these properties as it is established based on blockchain. In this section, we discuss several properties that our system offers.

A. Concusses in the Blockchain

With decentralized frameworks, particularly in blockchain-based, a concussion problem can arise. This problem happens when various miner compete to make a block around the same time [30]. Here each miner chooses the same block's hash as a parent hash and creates a block. These types of blocks are not accepted into the blockchain network. In the case of bitcoin, these types of blocks are called orphan block [31] and in the case of ethereum, we call it uncle block [32]. As shown in Fig. 6, when a miner creates a block, it will be linked to create a chain that neither corruptible nor changeable.

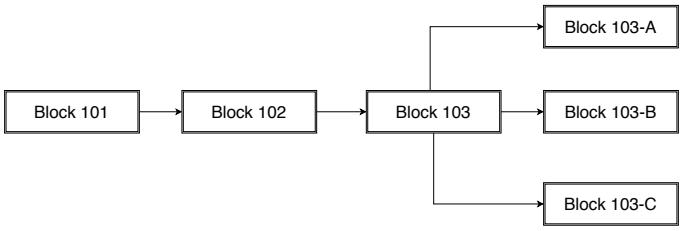

Fig. 6. Concusses in Blockchain.

In the proposed system, there is no concusses problem as all the task-related transaction maintained by the smart contract. Thus, each block's information will be sent to the miner after a specific time interval, as shown in Fig. 7 and, there is no chance for missing any block.

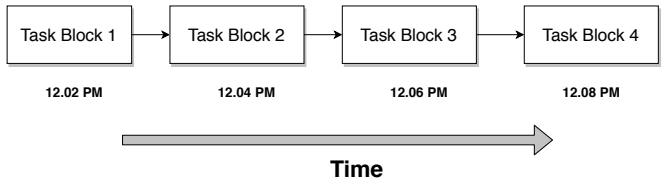

Fig. 7. Solution of Concusses.

B. Anonymity

The identity information of contributors and requesters are protected by anonymity which is essential for crowdsourcing platforms. The anonymity of contributor indicates the inter-connectivity between the retrival and the privacy of workers [15]. Similarly, anonymity of requesters can be defined as the submission unlinkability between task submission and identity information. In the proposed system uid_{treq} is the unique user ID of the task requester and who posted the crowdsourced task request and uid_{CON} is the unique user ID of the set of contributors that participated in the crowdsourced task. From these ID it is not possible to find the relation between task requester, task request and contributors.

C. Efficiency

The computing cost should be minimal enough to guarantee the equivalent services can be provided by the smart contract. The cost of the associated operation must not cross the gas limit [15]. The efficiency of the proposed system is shown in Section VII.

D. Integrity

Blockchain technology implements a Merkle tree to guarantee the integrity of the data [33]. In ethereum, the principle of Merkle tree was introduced to allow for a lightweight, and adequate verifiable proof that assures a transaction is added into a block. The hashes of the child nodes in this data structure are merged into the header of a parent node. This strategy for connecting headers of children's nodes and linking them to the header of the parent node goes on until the last node, right above the root node. This process implies that the root node holds information of all nodes. The Merkle tree includes a hash of all transactions in a block. Thus, if a node wishes to check if a transaction has been modified or not, nodes need to create the Markle Block using entire block transactions [34]. This tree makes validating or invalidating a task effortless.

E. Security

The proposed method ensures protection by preventing unauthorised access and misuse of task-related information. Due to the immutable attribute of Blockchain, it is difficult to alter the documented information on Blockchain. If someone changes a transaction, all block information from that block to the new block would have to be re-mined. A unique hash value is contained by each block using a hash function and the hash of the previous block. If one tries to modify a block's data, the different hash value will result out from this modification. New hash value will conflict with the next block's value. For this reason, re-mined will be needed for the next block. For all the blocks in the chain, the same process of re-mining is required. When the miner is involved in re-mining old blocks, new blocks would be attached to the chain, making it incredibly impossible to exploit a single block. This method requires a massive computing capacity, which is practically impossible in reality [34]. Blockchain, shown in Fig. 8 assures us the information is tamper-proof, and it is quite impossible to manipulate data in our proposed system.

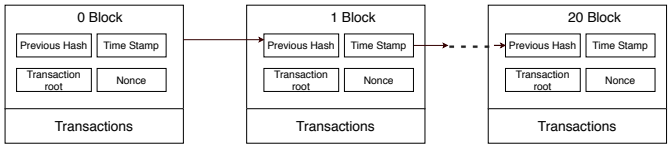

Fig. 8. Security in Blockchain.

F. Privacy

For widespread deployment and acceptance of crowdsourcing, privacy considerations must involve both task requester and contributors [35]. As task requester and contributor are registered into the system, and they use the unique identifier into the blockchain, this provides anonymity and privacy of requesters and contributors.

G. Fairness

Fairness in crowdsourcing refers to a fair sharing in labour results and rewards between the task requesters and the contributors who have given the right solutions [36]. In the proposed system, the task management process is done by smart contracts where the two party's agreement makes the

contracts. Smart contracts are responsible for checking the task results after submitted by the contributors. If the results match the requirements, then the reward money will be sent to the contributors to ensure fairness.

H. Decentralization

The blockchain is a decentralized distributed ledger system built into a network of multiple interconnected nodes. All the network nodes get a distributed ledger, which includes the transaction history known in the blockchain [37]. Compared to conventional systems that depend on crowdsourcing networks, the proposed scheme guarantees that transaction details of requesters and contributors are recorded in the blockchain.

I. Comparative Analysis of Property

In Table I, we have presented the comparison between existing systems and our proposed system based on several security properties which are discussed in this section. From this table, we can observe that the concuss solutions, integrity, efficiency are missing in [38], [15], [13], [39]. Moreover, [38] do not provide anonymity and privacy. In our proposed system, we have solved the concuss problem and provided all the properties.

TABLE I. COMPARISON BETWEEN EXISTING SYSTEMS AND PROPOSED SYSTEM

Properties	BPTM [15]	CrowdSFL [13]	WorkerREp [39]	Ours
Concuss	✓	✓	✓	×
Anonymity	✓	✓	✓	✓
Integrity	×	×	×	✓
Privacy	✓	✓	✓	✓
Security	✓	✓	✓	✓
Decentralization	✓	✓	✓	✓
Efficiency	×	×	×	✓

VII. EXPERIMENT RESULTS

The name of one of the most popular blockchain communities is Ethereum [40]. Ethereum facilitates smart contract functionality in such a way that any Ethereum wallet owner can implement their smart contract on the Ethereum network to benefit their specific business rules. To deploy a blockchain-based crowdsourcing framework in Ethereum, we need to set up the environment first.

A. Environment and Tools

For the implementation of our system prototype, Truffle, Ganache and Metamask are used.

1) Truffle: Truffle is primarily able to compile solidity-written smart contracts, executing migration on different contracts, and producing ABI (Application Binary Interface) [41].

2) Ganache: Ganache is a local RPC blockchain server which is embedded into Truffle. Ten initial accounts are provided by it to pre-fund with 100 Ether including a 12 words seed sentence to restore them.

3) MetaMask: MetaMask is a browser plug-in that can be used by a user on Chrome, Firefox, Opera and Brave. It adds an API to the browser that makes the read-write requests from standard websites in the Ethereum blockchain, which is a JavaScript library built by the Ethereum core team named web3.js. It enables users to transact Ethereum via traditional websites, communicate to an Ethereum node locally or remotely, via an HTTP or IPC connection[15].

B. Cost Analysis of the System

The cryptocurrency in the Ethereum chain is known as Ether (ETH). Computing cost is payable in ETH within the blockchain and EVM. The execution fee is measured in gas terminology. The Ethereum transaction specifies the data signed by the party initiating the exchange and includes a message sent from the client to some other blockchain client. Contracts can also transfer information to all other contracts when function calls have been formulated for each contract. There is a $gasPrice$ sector in a transaction, reflecting the fee that the sender is expected to pay for gas. A notification or another transaction causes the execution of a contract. Every command is then executed on every network node. For every executed operation there is a specified cost, expressed in several gas units and each transaction has a maximum ether cost that is then equal to $gasLimit \times gasPrice$ [42].

TABLE II. COMPARISON OF SMART CONTRACT DEPLOYMENT

Contract	Gas Used	Actual Cost (Ether)	USD
BPTM [15]	973093	0.0194619	6.77274
CrowdSFL[13]	455416	0.0091083	3.16969
WorkerRep[39]	247863	0.0049573	1.72514
Proposed System	156290	0.0031258	1.08778

In Table II, we show the gas costs and the corresponding prices in $ for the deployment of the agreement contract between existing and proposed systems. At the time of carrying out the experiments, September 2020, the ether exchange rate was 1 ETH = 348.28$, and the median gasPrice was approximately 0.0000002 ETH (20 Gwei). The comparison between different blockchain-based system is also illustrated in Fig. 9.

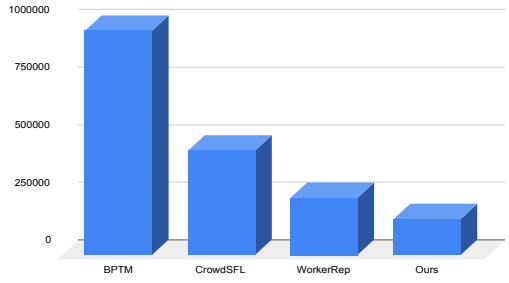

Fig. 9. Comparison Analysis.

VIII. Conclusions and Future Works

Privacy of the user's data is still a significant issue nowadays. As it is difficult to breach the confidentiality of data stored in a blockchain, we proposed a blockchain-based crowdsourcing model in this paper. The coordinator distributes the tasks in this model, so the fairness of the system prevails. Additionally, the tasks are assessed by the smart contract; therefore, it is possible to prevent any failure in the output. Through adopting blockchain, this model can provide anonymity, integrity, security and efficiency. We have analysed our model with a few existing models. Our model performs adequately well, and it consumes less gas and ether than other models. In future, we will add an incentive mechanism to our proposed model; hence, the users will be more interested in using this platform.

References

[1] J. Howe, "The rise of crowdsourcing," *Wired magazine*, vol. 14, no. 6, pp. 1–4, 2006.

[2] X. Zhang, Z. Yang, Z. Zhou, H. Cai, L. Chen, and X. Li, "Free market of crowdsourcing: Incentive mechanism design for mobile sensing," *IEEE transactions on parallel and distributed systems*, vol. 25, no. 12, pp. 3190–3200, 2014.

[3] J. Phuttharak and S. W. Loke, "A review of mobile crowdsourcing architectures and challenges: Toward crowd-empowered internet-of-things," *IEEE Access*, vol. 7, pp. 304–324, 2018.

[4] V. Pilloni, "How data will transform industrial processes: Crowdsensing, crowdsourcing and big data as pillars of industry 4.0," *Future Internet*, vol. 10, no. 3, p. 24, 2018.

[5] S. Nakamoto and A. Bitcoin, "A peer-to-peer electronic cash system," *Bitcoin.–URL: https://bitcoin. org/bitcoin. pdf*, 2008.

[6] J. Golosova and A. Romanovs, "The advantages and disadvantages of the blockchain technology," in *2018 IEEE 6th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE)*. IEEE, 2018, pp. 1–6.

[7] M. Crosby, P. Pattanayak, S. Verma, V. Kalyanaraman *et al.*, "Blockchain technology: Beyond bitcoin," *Applied Innovation*, vol. 2, no. 6-10, p. 71, 2016.

[8] L. Islam, S. T. Alvi, M. N. Uddin, and M. Rahman, "Obstacles of mobile crowdsourcing: A survey," in *2019 IEEE Pune Section International Conference (PuneCon)*. IEEE, 2019, pp. 1–4.

[9] H. Simula, "The rise and fall of crowdsourcing?" in *2013 46th Hawaii International Conference on System Sciences*. IEEE, 2013, pp. 2783–2791.

[10] R. Hull, "Blockchain: Distributed event-based processing in a data-centric world," in *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, 2017, pp. 2–4.

[11] W. Feng and Z. Yan, "Mcs-chain: Decentralized and trustworthy mobile crowdsourcing based on blockchain," *Future Generation Computer Systems*, vol. 95, pp. 649–666, 2019.

[12] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. H. Deng, "Crowdbc: A blockchain-based decentralized framework for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 6, pp. 1251–1266, 2018.

[13] Z. Li, J. Liu, J. Hao, and H. Wang, "Crowdsfl: A secure crowd computing framework based on blockchain and federated learning," *Electronics*, vol. 9, p. 773, 05 2020.

[14] Y. Lu, Q. Tang, and G. Wang, "Zebralancer: Crowdsource knowledge atop open blockchain, privately and anonymously," *arXiv preprint arXiv:1803.01256*, 2018.

[15] 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, 01 2019.

[16] L. Islam, M. A. Islam, and M. N. Uddin, "Three-layered incentive framework for mobile crowdsourcing," in *2018 10th International Conference on Electrical and Computer Engineering (ICECE)*. IEEE, 2018, pp. 153–156.

[17] ——, "A quality-aware reverse auction-based incentive model for mobile crowdsourcing," in *2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*. IEEE, 2019, pp. 1–5.

[18] J. R. Douceur, "The sybil attack," in *International workshop on peer-to-peer systems*. Springer, 2002, pp. 251–260.

[19] M. Hossain, "Crowdsourcing: Activities, incentives and users' motivations to participate," in *2012 International Conference on Innovation Management and Technology Research*. IEEE, 2012, pp. 501–506.

[20] S. Kim, Y. Kwon, and S. Cho, "A survey of scalability solutions on blockchain," in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2018, pp. 1204–1207.

[21] D. E. Difallah, M. Catasta, G. Demartini, P. G. Ipeirotis, and P. Cudré-Mauroux, "The dynamics of micro-task crowdsourcing: The case of amazon mturk," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 238–247.

[22] R. Yulius, F. Neta, M. F. A. Nasrullah, and M. F. A. Rambe, "Implementation of task-centered design in a web-based catalogue," in *2nd International Media Conference 2019 (IMC 2019)*. Atlantis Press, 2020, pp. 381–386.

[23] T. Xie and X. Qin, "Stochastic scheduling with availability constraints in heterogeneous clusters," in *2006 IEEE International Conference on Cluster Computing*. IEEE, 2006, pp. 1–10.

[24] S. Wang, Z. Zheng, Q. Sun, H. Zou, and F. Yang, "Evaluating feedback ratings for measuring reputation of web services," in *2011 IEEE International Conference on Services Computing*. IEEE, 2011, pp. 192–199.

[25] H. T. Nguyen, W. Zhao, and J. Yang, "A trust and reputation model based on bayesian network for web services," in *2010 IEEE International Conference on Web Services*. IEEE, 2010, pp. 251–258.

[26] S. Liu, H. Yu, C. Miao, and A. C. Kot, "A fuzzy logic based reputation model against unfair ratings," in *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*, 2013, pp. 821–828.

[27] A. Abdel-Hafez, Y. Xu, and A. Jøsang, "A normal-distribution based rating aggregation method for generating product reputations," in *Web Intelligence*, vol. 13-1. IOS Press, 2015, pp. 43–51.

[28] A. Dorri, S. S. Kanhere, and R. Jurdak, "Blockchain in internet of things: challenges and solutions," *arXiv preprint arXiv:1608.05187*, 2016.

[29] M. A. Uddin, A. Stranieri, I. Gondal, and V. Balasubramanian, "Continuous patient monitoring with a patient centric agent: A block architecture," *IEEE Access*, vol. 6, pp. 32 700–32 726, 2018.

[30] A. B. Ayed, "A conceptual secure blockchain based electronic voting system," *International Journal of Network Security & Its Applications*, vol. 9, pp. 01–09, 2017.

[31] P. Tasatanattakool and C. Techapanupreeda, "Blockchain: Challenges and applications," in *2018 International Conference on Information Networking (ICOIN)*. IEEE, 2018, pp. 473–475.

[32] S.-Y. Chang, Y. Park, S. Wuthier, and C.-W. Chen, "Uncle-block attack: Blockchain mining threat beyond block withholding for rational and uncooperative miners," in *International Conference on Applied Cryptography and Network Security*. Springer, 2019, pp. 241–258.

[33] B. Rogers, S. Chhabra, M. Prvulovic, and Y. Solihin, "Using address independent seed encryption and bonsai merkle trees to make secure processors os- and performance-friendly," in *40th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO 2007)*, 2007, pp. 183–196.

[34] R. Bosri, A. R. Uzzal, A. A. Omar, A. S. M. T. Hasan, and M. Z. A. Bhuiyan, "Towards a privacy-preserving voting system through blockchain technologies," in *2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, 2019, pp. 602–608.

[35] J. Phuttharak and S. Loke, "A review of mobile crowdsourcing architectures and challenges: Towards crowd-empowered internet-of-things," *IEEE Access*, vol. PP, pp. 1–1, 12 2018.

[36] J. Zhang, W. Cui, J. Ma, and C. Yang, "Blockchain-based secure and fair crowdsourcing scheme," *International Journal of Distributed Sensor Networks*, vol. 15, p. 155014771986489, 07 2019.

[37] S. Gao, D. Zheng, R. Guo, C. Jing, and C. Hu, "An anti-quantum e-voting protocol in blockchain with audit function," *IEEE Access*, vol. PP, pp. 1–1, 08 2019.

[38] J. Fan, G. Li, B. C. Ooi, K.-l. Tan, and J. Feng, "icrowd: An adaptive crowdsourcing framework," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, 2015, pp. 1015–1030.

[39] G. K. Bhatia, S. Gupta, A. Dubey, and P. Kumaraguru, "Workerrep: Immutable reputation system for crowdsourcing platform based on blockchain," *arXiv preprint arXiv:2006.14782*, 2020.

[40] G. Wood *et al.*, "Ethereum: A secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, no. 2014, pp. 1–32, 2014.

[41] Y. Rosasooria, A. K. Mahamad, S. Saon, M. A. M. Isa, S. Yamaguchi, and M. A. Ahmadon, "E-voting on blockchain using solidity language," in *2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE)*, 2020, pp. 1–6.

[42] C. Braghin, S. Cimato, S. R. Cominesi, E. Damiani, and L. Mauri, "Towards blockchain-based e-voting systems," in *International Conference on Business Information Systems*. Springer, 2019, pp. 274–286.