Hindawi
Security and Communication Networks
Volume 2022, Article ID 8034677, 15 pages
https://doi.org/10.1155/2022/8034677

Research Article

A Trusted and Privacy-Preserved Dispersed Computing Scheme for the Internet of Mobile Things

Yan Zhao,[1] **Ning Hu**,[1] **Yue Zhao**,[2] **and Yuqiang Zhang**[1]

[1]*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China*
[2]*Science and Technology on Communication Security Laboratory, Chengdu, China*

Correspondence should be addressed to Ning Hu; huning@gzhu.edu.cn

Received 3 February 2022; Revised 22 June 2022; Accepted 4 August 2022; Published 16 September 2022

Academic Editor: Ilsun You

Existing computing paradigms of the Internet of Mobile Things (IoMT) and social networks cannot effectively serve users due to their limited computing ability, dynamic mobile networks, weak connectivity, and high energy consumption and investment costs. Dispersed computing (DCOMP) is a promising way to solve the above issues. However, DCOMP is emergent, and few studies apply DCOMP to IoMT and social networks. Moreover, security problems also arise when introducing DCOMP to IoMT and social networks. In this paper, we propose a novel reference architecture to realize DCOMP in IoMT or social networks. We also propose two models—a trusted application discovery and acquisition model and a security domain-based computing offloading model—to enhance the security of the proposed architecture. The key idea of the first model is to use blockchain to construct a trusted application storage and acquisition system. This system guarantees that the tasks offloaded to dispersed devices are trusted. In the second model, we design two algorithms to offload tasks to appropriate dispersed devices to protect users' privacy as much as possible. The experimental results prove the effectiveness of the proposed scheme.

1. Introduction

With the rapid development of the Internet of Things (IoT) and social networks, a new paradigm combining social networks with the IoT has appeared and received increasing attention [1]. The Internet of Mobile Things (IoMT), which is a subset of IoT, focuses on connecting mobile devices, such as smartphones, vehicles, and wearable devices, to the Internet [2]. These mobile things with computing and network functionality automatically interact with other objectives and independently establish social relationships [3]. More than 50 billion mobile devices are forecasted to connect to the Internet by 2025 [4]. It can be expected that IoMT-powered social networks with unprecedented scale and intelligence will appear soon.

Although the IoMT has given people more convenient and intelligent lives, it has also introduced new threats in terms of privacy and trust. Due to the limited performance of IoMT devices, it is difficult to deploy complex security services on them, and they are very vulnerable to cyberattacks [5]. Therefore, we need to ensure that the service provided to the user by the IoMT device is trustworthy, which means that the device will provide the service as the user expects, without unintended or malicious behavior. On the other hand, the personal information collected by these devices has the risk of being stolen and illegally accessed in the process of transmission, storage, calculation and sharing, posing a threat to people's privacy [6]. Many research efforts have attempted to address the above challenges. For the trust problem of the IoMT, the main solutions are decision-based methods and attribute-based methods [7]. Decision-based approaches typically have a decision-making entity that trusts devices through trust policies or rules. Attribute-based methods usually build a trust computing model to quantify the trustworthiness of a device or service. For the privacy protection of mobile IoT, the current solutions mainly include lightweight encryption [8], homomorphic encryption [9], blockchain, and federated learning [10]. These schemes protect the privacy of user data in terms of communication, storage, sharing, and computing.

Although the above solutions have achieved good results in terms of trust and privacy protection, they make the computation inefficient. Currently, the computing paradigm of IoMT mainly includes two categories: mobile cloud computing (MCC) and mobile edge computing (MEC) [11]. However, the immense traffic generated by massive IoMT devices will burden the core network, and the long transmission path causes a high transmission delay [12, 13]. MEC offloads IoMT applications to the edge server located at the mobile base station, alleviating network congestion, and poor real-time performance [14]. However, due to the limited computing power of mobile edge servers, it is difficult to effectively provide services for multiple users with heavy computing demands [15]. Moreover, the dynamicity of mobile or social networks and weak connectivity of mobile devices hinder the ability of MEC to provide continuous and reliable services to users. In addition, network operators need to deploy a large number of edge servers in MEC systems, which also leads to substantial investment costs and energy consumption [16].

We argue that the "end-to-end" computing paradigm should be broken and that a more efficient computing system can be employed to completely solve the above problems. Dispersed computing is a new and disruptive computing paradigm that has gained widespread attention [17, 18]. This paradigm breaks the traditional "end-to-end" computing paradigm and integrates network and computing through "forward and computing hop by hop," which can significantly reduce core network load and effectively improve network and application performance. In this paper, we propose a trusted and privacy-preserving dispersed computing scheme for IoMT and social networks. A three-layer architecture of dispersed computing, which realizes DCOMP in IoMT and social network scenarios, is introduced. Moreover, two models are proposed to overcome the trust and privacy problem in the proposed architecture. First, we use blockchain and a content-based, addressing file system to construct a trusted application discovery and acquisition system, which gives the dispersed device the ability to verify whether the offloaded task is trusted. Second, we design two algorithms to offload tasks with different privacy risks to an appropriate dispersed platform with different security levels to protect users' privacy as much as possible. The main contributions of this work are presented as follows:

(i) A novel dispersed computing architecture, which includes an application layer, control layer, and computing layer, is proposed. The proposed architecture can be used to build a secure IoMT and social network computing system and provides a reference architecture design of a dispersed computing paradigm suitable for an IoMT or social network.

(ii) A trusted application discovery and acquisition model that is based on the consortium blockchains and the content-based addressing file system is proposed. Both the on-chain metadata storage mechanism and off-chain content storage mechanism are utilized in the proposed model to build a system that enables dispersed devices to discover and acquire applications in a trusted way.

(iii) A security domain-based computing offloading scheme is proposed. For different users, the proposed method groups dispersed devices into different domains. By scheduling IoMT tasks with different security risks to appreciate security domain devices, the privacy of user data is guaranteed.

This paper is an extension and revision of the conference paper [19]. Compared with previous work, this version (1) provides more related works; (2) provides more background for dispersed computing; (3) proposes a reference dispersed computing architecture for the IoMT; (4) designs a trusted application discovery and acquisition model that enables trusted IoMT application discovery, acquisition, and offloading; and (5) introduces more details about the security domain-based computing offloading model.

The remainder of the paper is organized as follows: An overview of related work is presented in Section 2. The background and motivation of our work are described in Section 3. A Secure Dispersed Computing Scheme for the Internet of Mobile Things is elaborated in Section 4. The experimental results of the proposed scheme and algorithm are discussed in Section 5. In Section 6, the conclusion and future work are provided.

2. Related Works

2.1. Architecture of the IoMT. With the cloud/edge computing paradigm, data collected from mobile devices can be processed and analyzed at cloud or edge platforms, which offer more intelligent services to users. Nastic et al. [20] propose a unified cloud and edge data analytics platform, which applies the conception of serverless computing to edge and cloud computing. This platform integrates cloud, edge/fog devices, and IoT devices as an infrastructure resource pool. The data collected from underlying devices can be processed and analyzed on edge nodes, cloud nodes, or both based on the scheduling and placement mechanisms to reduce the network latency. Greco et al. [21] proposed a 4-layer architecture for real-time data analysis of wearable sensors, which consists of a sensing layer, preprocessing layer, cluster processing layer, and persistency layer. The data collected by the sensing layer are converted to RDF streams and sent to the cluster processing layer to perform data analysis tasks. The processed data are stored in the persistency layer.

To further reduce the transmission delay of edge computing systems, some edge computing architectures based on 5G are proposed. Nunna et al. [22] propose a collaboration platform with mobile edge computing (MEC) and 5G to effectively support use cases with low-latency demand. In the schematic view of the MEC-based collaboration system, MEC servers connect to 5G base stations and provide an API to various applications by a middleware. Moreover, two

application cases of road accident scenarios and remote robotic telesurgery are also given to explain how the proposed collaboration platform can be used in the real world. Tran et al. [23] propose a real-time, context-aware cooperation framework of MESs in 5G networks, which are composed of MEC servers and mobile devices. This framework implements the MES servers at 5G base stations and integrates heterogeneous resources to form a resource pool.

However, the above IoMT architectures adopt an end-to-end computing paradigm that separates the computing and networking processes, and the network only responds to data transmission without data processing. With this computing and networking paradigm, it is difficult to jointly optimize computation offloading and network flow routing and scheduling to deeply improve the performance of the IoMT and social applications.

2.2. Trust Mechanism of IoMT. The trust mechanism has a crucial role in the IoMT. Whether users, nodes, data, and services are trusted is important for the security of many IoMT applications, and many studies have addressed this issue [24–26]. To select trusted users who can contribute high-quality data in the mobile crowd-sensing scenario, Truong et al. [27] propose an experience-reputation-based trust computation method to evaluate the trust relationships among users of mobile things. By establishing virtual interactions among user service requests and data contributors and by performing the computation of the quality of data, first, the experience relationships are established and calculated. Second, based on these experience relationships, the reputation of the user is calculated. Last, the trust value is a linear function based on the above experience value and reputation value. The limited computing resources of the IoT make it difficult to deploy trust evaluation to IoT devices, and placing the trust evaluation process on the cloud has the problem of obtaining fine-grained information of IoT nodes due to the long distance between the cloud and the IoT device. To address this challenge, Wang et al. [28] propose an edge-based trust evaluation scheme, which offloads the trust evaluation task to mobile edge devices. The authors model the trust between two IoT nodes as three types of trust chains and provide a corresponding calculation method and algorithm.

However, regardless of whether the trust evaluation process is placed on cloud or edge nodes, the above trust solutions are centralized trust systems that depend on a pretrusted three-party. For this problem, blockchain-based trust solutions have received increasing attention. Kouicem et al. [29] design an IoT trust management protocol using blockchain technology, which provides mobility support and enables a trust system without a pretrusted, three-party entity. In this decentralized trust management system, mobile devices that share trust data receive IoMT services through blockchain. Yang et al. [30] propose decentralized trust management in vehicular networks based on blockchain and edge computing. The roadside unit is used to maintain a blockchain system to collect, store, and manage the trust value of a vehicle's message.

2.3. Privacy Preservation of IoMT. Using data aggregation and encryption algorithms for privacy preservation is a common method. Li et al. [6] propose a privacy-preserving data aggregation scheme for IoT applications under mobile edge computing environments. The proposed scheme aggregates encrypted data of terminal devices by the edge node. These aggregated data are transmitted to the cloud platform and decrypted by the cloud private key. The evaluation shows that this scheme can reduce approximately half of the communication overhead compared with the traditional method while protecting user data privacy. Usman et al. [31] propose P2DCA, a privacy-preserving-based data collection and analysis framework for IoMT applications, which uses data aggregation and a local differential privacy service to protect the location privacy of IoT devices. In the proposed framework, data of IoT devices are collected and aggregated by a cluster head used to manage a set of IoT terminals. The cluster head sends the aggregated data to the cloud platform for analysis by a Counter-Propagation Artificial Neural Network that is responsible for extracting meaningful information from aggregated data. Applying privacy-preserving mechanisms to IoMT applications may degrade the quality of service (QoS).

Anonymous communication technology is a promising way to protect user privacy in IoMT or social applications. However, due to the poor computing ability of IoT devices, it is difficult to implement anonymous mechanisms that rely on heavy cryptographic processes. To close this gap, Hiller et al. [32] designed a tailored onion routing that offloads the expensive cryptographic processing to a router or web server. Privacy-aware computing offloading is another research hotpot in the IoMT field. Location information of IoMT devices may be exposed when the IoMT interacts with the mobile edge server to which the task is offloaded. For this problem, He et al. [33] propose a privacy-aware offloading method based on a deep postdecision state learning technique that enables IoT devices to learn a rational offloading strategy at a faster speed.

3. Background and Motivation

3.1. Background. Recently, a new computing paradigm, dispersed computing (DCOMP), has attracted considerable attention [17, 18, 34–37]. DCOMP provides a promising way to solve the challenges of the existing computing paradigms of the IoMT and social networks [38]. The dispersed computing paradigm was proposed by the Defense Advanced Research Projects Agency (DARPA) in 2016 [39]. The basic idea of DCOMP is to realize the code and data on-demand movement, which enable the secure and collective execution of tasks on geographically dispersed, ubiquitous, and heterogeneous computing platforms [40, 41]. DCOMP is aimed at breaking the traditional "end to end" design principle of the network and adopting a "forwarding and computing hop by hop" manner, which renders the network not only a data transmission channel but also a data processing platform. DCOMP promises to significantly improve application and network performance and to minimize the task failure risk

when the network condition degrades. Moreover, DCOMP reduces energy consumption and investment costs [16].

Figure 1 shows the IoMT dispersed computing paradigm [18, 40]. The dispersed computing system of the IoMT consists of three parts, namely, the IoMT service provider, dispersed task-aware computing, and programmable nodes and protocol stacks. The workflow of the IoMT is described as follows: (1) The IoMT device sends a service request to the dispersed task-aware computing module; (2) The dispersed task-aware computing module sends a request to the service discovery module of the IoMT service provider based on the task identifier; (3) The service discovery module retrieves the service storage database and sends the service to the dispersed task-aware computing module; (4) The dispersed task-aware computing module calculates the task offloading plan, offloads multiple tasks to the appropriate DCP and sends the flow table to the DCP; (5) The IoMT device sends data to the first DCP node, which are then forwarded and processed hop by hop.

Dispersed computing and edge computing are similar as both emphasize pushing computing closer to users and provide computing services to nearby users by offloading computing. However, dispersed computing and edge computing are fundamentally different. Edge computing still uses the network as a channel for data transmission, and data computing and processing are performed on computing terminals (such as edge servers) at the edge of the network. In the dispersed computing paradigm, the network is not only a data transmission channel but also a data processing platform. Dispersed computing tasks will be computationally offloaded along the data transmission path, and data will be processed and calculated hop-by-hop during the hop-by-hop transmission of the network. Similarly, federated learning is fundamentally different from dispersed computing. Federated learning is essentially a distributed edge system, where large amounts of data are computed and processed on users' local or edge servers. Federated learning is still a traditional "end-to-end" computing paradigm, and the network is only utilized as a data transmission channel. As a result, a large number of network devices in the network remain idle for a long time. Therefore, compared with federated learning, the system based on dispersed computing can effectively utilize the computing resources in the network and improve the utilization rate of the computing resources of the system.

3.2. Motivation and Objectives. However, DCOMP is still in its infancy, and reference architecture to guide people to implement the DCOMP paradigm in the IoMT and social network scenarios is lacking. Moreover, applying DCOMP to the IoMT or social networks will introduce new security challenges.

First, untrusted applications may be offloaded to other users' devices through DCOMP. In the DCOMP environment, users can share the idle computing resources of their devices with other users. We refer to these devices as dispersed computing points (DCPs). Although the DCOMP system provides users with low-latency and high-reliability services, it introduces security risks to the DCP. For example, a malicious user can offload malicious code to a nearby DCP through the DCOMP system. These malicious codes perform malicious operations on the DCP and threaten the devices. Moreover, DCOMP can be employed to build a botnet. For example, malicious users offload specific code to numerous DCP devices. This code establishes massive half-TCP connections to a target host to launch a DDoS attack. Therefore, DCOMP needs a mechanism that guarantees that applications or tasks offloaded to the DCP are trusted to protect the security of the DCP.

Second, malicious DCP may steal the privacy information of normal users through DCOMP. In the DCOMP environment, normal users offload applications to nearby DCP nodes for execution. Therefore, user data need to be stored and processed on multiple untrusted DCP devices. Storing unencrypted user data on these devices will result in the disclosure of user privacy. Even if the data have been encrypted to ensure the confidentiality of transmission and storage, the data need to be decrypted during processing, which means that the encrypted data are processed on untrusted devices, and there is still a problem of user privacy leakage.

According to the above analysis, it is necessary to design a method for trusted application discovery and code acquisition in the IoMT dispersed computing paradigm. During application or task offloading, the integrity of the code must also be guaranteed. In addition, the privacy of user data should be preserved when user data are processed on the DCP. To the best of our knowledge, current research on dispersed computing focuses on architecture design and task scheduling [17, 18, 34–37], [42], and few studies explore security solutions in the IoMT dispersed computing scenario. Therefore, in this paper, we propose a secure dispersed computing scheme for the Internet of Mobile Things, which has the following design objectives:

(i) To enable the trusted discovery and acquisition of an application and its code.

(ii) To preserve user data privacy as much as possible when user data are processed on dispersed devices.

4. Overview of Proposed Scheme

In this section, we provide details about the proposed DCOMP scheme for the Internet of Mobile Things and social networks. First, a reference DCOMP architecture is provided. Second, we propose two models for the above design objectives. In the first model, we design a trusted application discovery and acquisition scheme based on blockchain and distributed file system technologies, which adopt an on-chain storage system of application metadata and an off-chain storage system of application content. In the second model, we propose a security domain-based computing offloading method that protects user data privacy by offloading tasks with different security risks to appropriate dispersed computing devices.

4.1. Architecture. The proposed architecture, including the application layer, control layer, and computing layer, is shown in Figure 2. The key modules and components consist

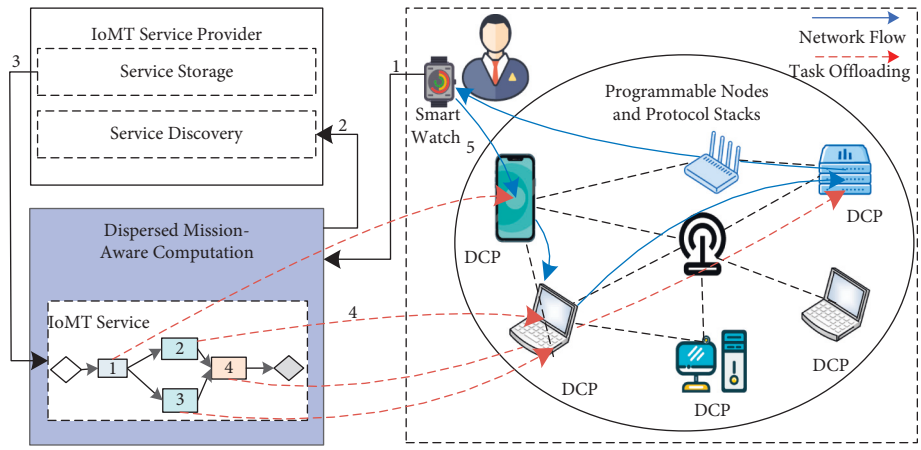

Figure 1: IoMT dispersed computing paradigm.

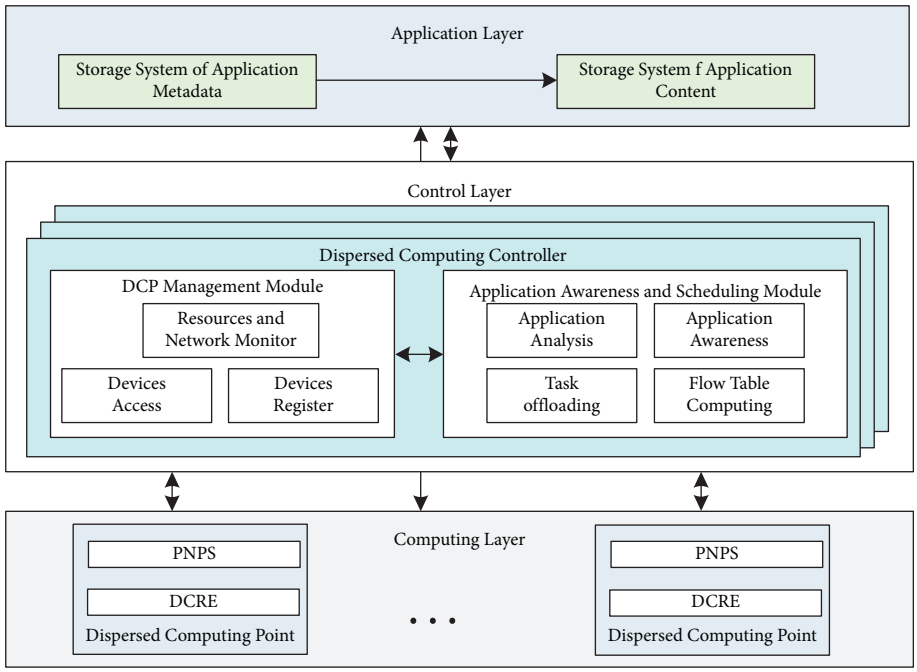

Figure 2: Architecture of the proposed scheme.

of the dispersed computing nodes, dispersed computing operating environment, programmable network protocol stack, dispersed computing controller, application metadata storage system, and application content storage system.

4.1.1. Dispersed Computing Point (DCP). Dispersed computing points include all kinds of dispersed computing devices and platforms, and the computation task of the IoMT or social applications are offloaded to the DCP for execution. A DCC can be any device with computing capabilities, such as mobile phones, tablet computers, in-vehicle computing devices, smart switches, Wi-Fi devices, and edge server devices.

4.1.2. Dispersed Computing Runtime Environment (DCRE). The dispersed computing runtime environment (DCRE) is a lightweight software system that is used to support the execution of computing tasks. Any DCP needs to install the DCRE and is responsible for device access, device status monitoring, task trust verification, and task content acquisition and execution. In addition, the DCRE is also responsible for accepting task offloading requests and deploys and executes the task.

4.1.3. Programmable Network Protocol Stack (PNPS). The programmable network protocol stack includes the logic of the programmable network control plane and programmable network data plane.

4.1.4. Dispersed Computing Controller (DCC). The dispersed computing controller (DCC) is responsible for DCP management, application awareness, computing scheduling, and the DCOMP network control plane. Each DCC manages multiple DCP devices around it. The key modules of the DCC include the DCP management module and dispersed task perception and scheduling module.

(1) DCP Management Module. The DCP management module is responsible for dispersed device access, registration, and resource monitoring. The device access submodule is responsible for the access of dispersed devices by a fixed port. The device registration module records information such as the IP address, management port, device identification, resource type information, mobility information, and digital certificates of DCP devices. The resource monitoring and network monitoring modules are responsible for monitoring the status and available resource information of the managed DCP devices.

(2) Task Awareness and Scheduling Modules. The task awareness and scheduling module offloads the application requested by the user to the appropriate DCP device in its management domain according to the user's service request and distributes the network flow table to the corresponding DCP.

4.1.5. Storage System of Application Metadata. The application metadata storage system is used to implement a trusted application discovery mechanism and provides an open interface through which DCC and DCP devices can obtain metadata about applications and tasks in a trusted way.

4.1.6. Storage System of Application Content. The application content storage system is used to implement a trusted application acquisition mechanism and to provide an open interface through which the DCP device can obtain the task content in a trusted way.

Figure 3 shows the collaborative process of each component of the proposed architecture. The process is detailed as follows:

(1) The user initiates a service request to the DCC, requesting the DCC to offload the IoMT application to nearby DCP devices that are available.

(2) After the DCC receives the user's service request, it extracts the application identifier from the request and then retrieves the application metadata from the application metadata storage system.

(3) The DCC analyses the metadata of the application and obtains the application's task list, task dependencies, resources and QoS requirements of each task, and other information.

(4) The DCC calculates the offloading location of each task of the application according to the global resource view of the DCP under its management.

(5) The DCC calculates the data flow forwarding rules between the tasks of the application according to the global network view of the DCP devices under its management and schedules the network flow between two tasks through the flow table.

(6) The DCC sends a task offloading request to the corresponding DCP.

(7) After the DCP receives the task offloading request, it extracts the application ID and task ID from the request and verifies whether the task is trustworthy through the application metadata storage system.

(8) After the trustworthiness of the task is verified, the DCP requests the application content storage system for the necessary code, data and supporting environment to execute the task.

(9) DCP locally deploys tasks.

(10) The DCP sends a response message to the DCC to notify the task that offloading has been completed.

(11) When all DCCs receive the offloading completion responses of all tasks, the DCC sends a response message to the user to notify the user that the application offloading has been completed and to inform the application about the offloading location of each task.

(12) The user initiates a service request to the first DCP devices.

(13) The programmable network protocol stack of the DCC directs user data to corresponding tasks for processing according to the application identification and task identification in the service request.

(14) The user data are forwarded and computed hop-by-hop among multiple DCP devices. The DCP sends the processing results and response messages to the user after the data are processed according to the application logic.

4.2. Trusted Application Discovery and Acquisition Model. In the proposed IoMT DCOMP architecture, DCP devices contribute their available resources to provide computing services for nearby users. However, in actual application scenarios, one situation that must be considered is that malicious users may utilize the DCOMP mechanism to offload malicious code to the DCP device. DCP devices may be personal devices of other users or organizations, and running untrusted codes on these devices may cause serious security problems. To solve this problem, this section proposes a trusted application discovery and acquisition model that provides a trust mechanism, so that the DCC and DCP can discover, acquire, and verify applications and tasks in a trusted manner.

4.2.1. Basic Ideal. The premise of this model is to use blockchain to build a trust mechanism for the discovery and acquisition of IoMT applications. Blockchain has the characteristics of trustworthiness and verifiability and has been successfully utilized for many systems [5, 43–50].

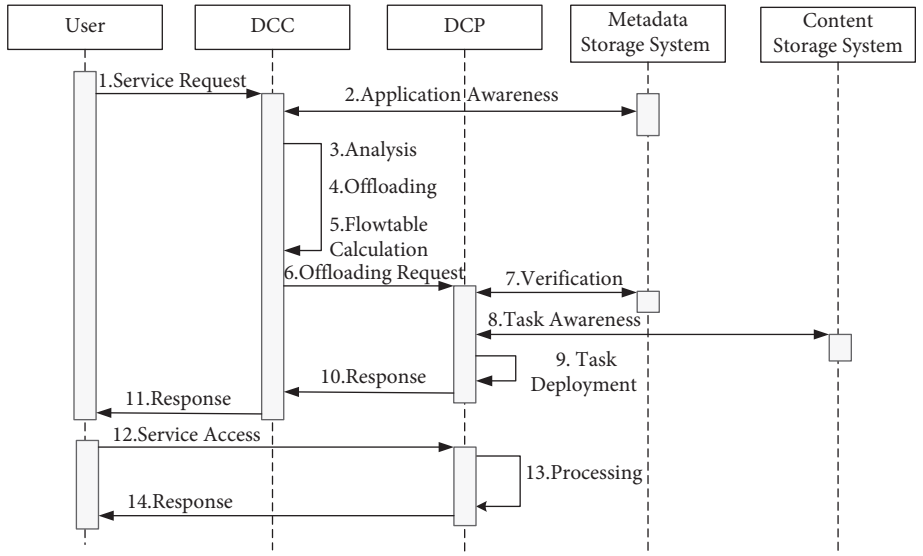

Figure 3: Cooperation among the components of the proposed architecture.

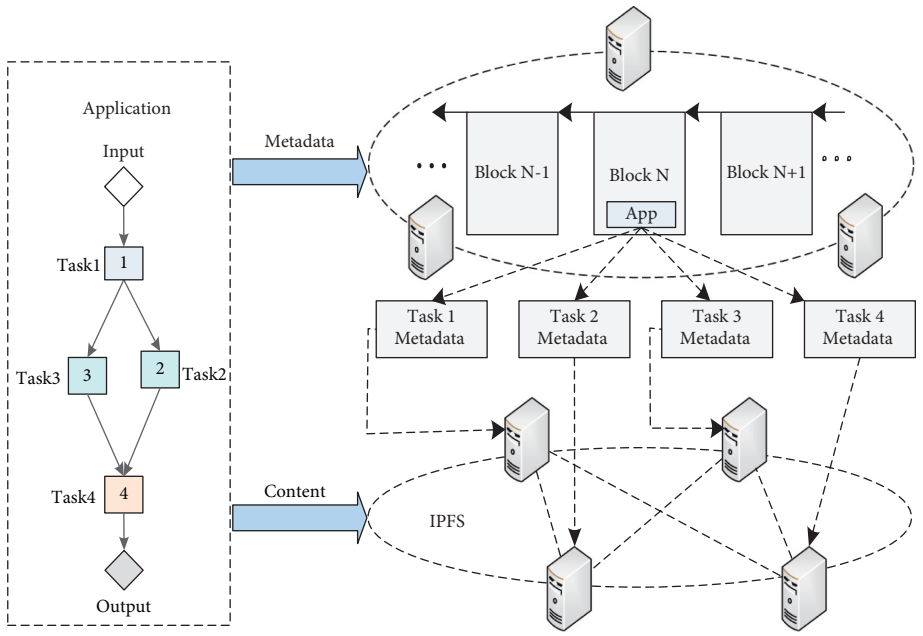

Figure 4: Trusted application discovery and acquisition model.

The proposed trusted application discovery and acquisition model is shown in Figure 4. The model consists of three components, namely, the IoMT application, consortium blockchain, and IPFS. The IoMT application is based on a microservice architecture and can be represented by a directed acyclic graph (DAG), in which the nodes represent the tasks of the application and the connections represent the dependencies between two tasks. This model uses a consortium blockchain to store metadata for mobile IoT applications. These IoMT application providers jointly endorse the trust of applications on the chain. Any application that wants to be published on the chain must pass the security verification for more than half of the consortium entities. After security verification, a security certificate is generated for the application, which contains the application ID, task ID, and digital signature of the verification node. When an application provider submits a metadata publication request to the blockchain, the metadata include at least half of the consortium node security certificates, and a transaction that encapsulates the application metadata will be received by other nodes in the consortium chain.

The proposed model uses a distributed file system to store application content. The distributed file system uses the IPFS [51]. Service providers publish service metadata to the blockchain and the task content of the application to the IPFS. The IPFS cluster is maintained by the service

providers. The application metadata include the hash value of each task content of the application, which is used to index the task content in the IPFS. The integrity and authenticity of the task content are guaranteed by the application metadata stored in the consortium chain.

4.2.2. Storage of Application Metadata. Due to the limited block capacity, it is unrealistic to store the service code in the blockchain. Therefore, only application metadata are stored in the consortium chain. The DCC only needs the metadata information of the application to calculate the offloading plan of the task.

The content of the application metadata is shown in Figure 5. The AID field identifies an application through a globally unique identification algorithm. The version field indicates the version of an application. The certificate list is a set of security certificates issued by consortium nodes. The list should include at least half of the security certificates issued by the consortium nodes. The signature file is a digital signature provided by the application provider, which can be used for authenticity verification of an application. The task dependency information describes the number of tasks included in the application and the dependencies among tasks. There is also a list of task metadata, which include the task ID, task security level, resource requirements, and content hash fields. The task ID uniquely identifies the task of an application. When the DCC performs task offloading, it must determine whether the location of the task offloading point meets the security level according to the task security level. The resource requirements field describes the computing, storage, and network resources required to perform the task. The content hash field stores the hash value of the task content, which is also the storage address of the task content in the IPFS.

Before the application provider releases application metadata to the consortium chain, it needs to obtain the security certificates of at least half of the nodes in the consortium chain. Afterward, the application provider generates the metadata of the application and sends a transaction request to any node in the consortium chain, and the node verifies the certificate list and digital signature in the application metadata through a smart contract. Next, this node verifies whether the hash value is a valid IPFS file address. If the verification is passed, the node generates a transaction and publishes it to the blockchain and reaches a consensus with other nodes through a consensus algorithm, such as PBFT [48].

4.2.3. Storage of Application Metadata. The IPFS is used in our model to store application content. In the IPFS file system, each file will generate a hash value based on the content, and in the IPFS, each file is addressed based on its content hash value. In our model, the application metadata contain the content hash value of each task of the application. DCP requests task content from the IPFS based on the task content hash value in the application metadata.

An IoMT application content is composed of multiple tasks. The complete content of a task may include multiple

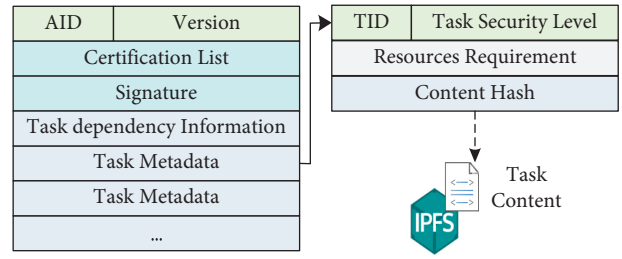

Figure 5: Trusted application discovery and acquisition model.

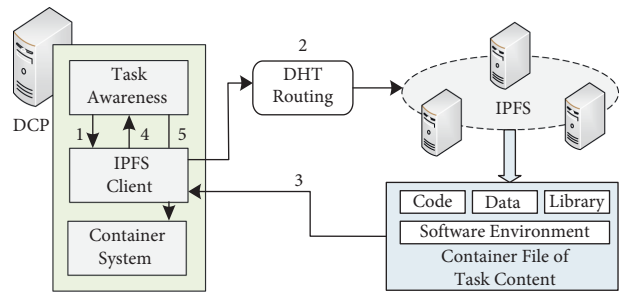

Figure 6: Task awareness operation.

files, such as executable code, necessary data, libraries, and software environment. All content of a task is packaged into a container and stored in the IPFS. The application provider should submit the container file of each task of its application to the IPFS and then generate the metadata of the application and publish it to the consortium blockchain. After DCP obtains the container of a task from IPFS, it can directly and locally deploy and allow the task without installing any software environment that supports the operation of the task. Figure 6 shows the process of DCP obtaining task content from the IPFS.

4.2.4. Operation of Application Discovery and Acquisition Model. The proposed trusted application discovery and acquisition model provides a trusted way for DCC and DCP to discover and acquire application content. The process is shown in Figure 7. First, the application awareness module of the DCC requests application metadata from a node in the consortium chain. After obtaining the application metadata, the DCC calculates the uninstallation plan of the application according to the application metadata. Next, the DCC sends a task uninstallation request to the corresponding DCP. After the DCP receives the task offloading request, it requests verification of the task from a node in the consortium chain. The consortium chain node then initiates a visa transaction and calls the smart contract responsible for the verification request in the transaction. The smart contract will verify whether the task exists in the blockchain. The task is included in the application metadata, and the integrity of the task metadata is verified. After the smart contract is verified, the consortium chain node will return the response to the DCP verification result. If the verification result is true, the DCP requests the content of the task to the IPFS

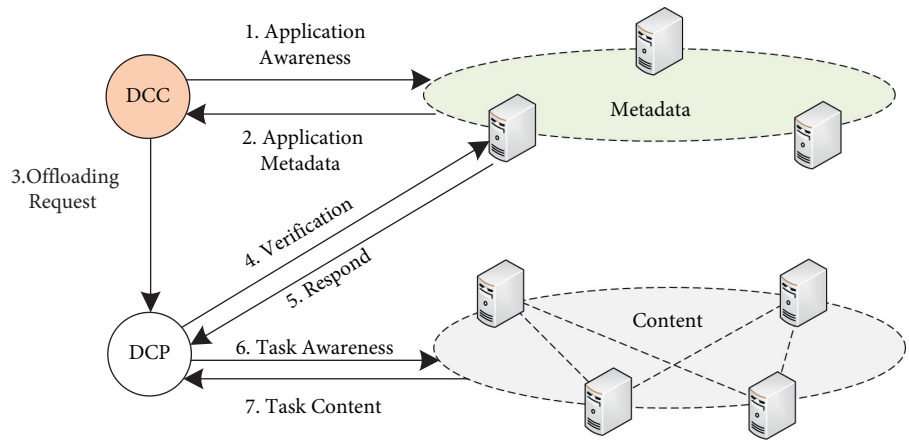

Figure 7: Operation of application discovery and acquisition.

based on the hash of the task content in the task metadata and locally deploys the task, as shown in Figure 7.

4.3. Security Domain-Based Computing Offloading Model. The trusted application discovery and acquisition model provides security protection for the DCP, enabling malicious users to offload untrusted code onto the DCP through the dispersed computing system. However, another situation is that malicious users use their devices as DCP devices. Even if the user data are encrypted, the data still need to be decrypted when it is processed on the malicious person's DCP, which will cause the user's privacy to be leaked. To solve this problem, this section proposes a security domain-based application uninstall program to protect the privacy of user data as much as possible.

4.3.1. Basic Idea. The proposed model is based on the following two observations: For an IoMT service, different tasks have different security risks. On the other hand, for a user, nearby devices have different security priorities. Based on the above observations, we propose a computing offloading method based on security domains. The premise of the proposed method is that the DCC divides the DCP in its management domain into private domain devices, organizational domain devices, and public domain devices according to users. By scheduling tasks with different security risks to appropriate domain devices, the privacy of user data is improved.

The proposed model is shown in Figure 8. For a user and DCC that provide services for the user, the DCP devices under DCC management are divided into private domains, organizational domains, and public domains. The private domain is composed of the user's private devices, such as cell phones, mobile computers, and smart home equipment. The organization domain is composed of the devices of the organization to which the user belongs, such as the devices of the user's home, company, school, and other organizations. The public domain is composed of various public devices, such as the edge server of the network operators and smart city devices. In addition, users can share idle computing resources of their devices with nearby users. At this time, these computing resources also belong to the public domain.

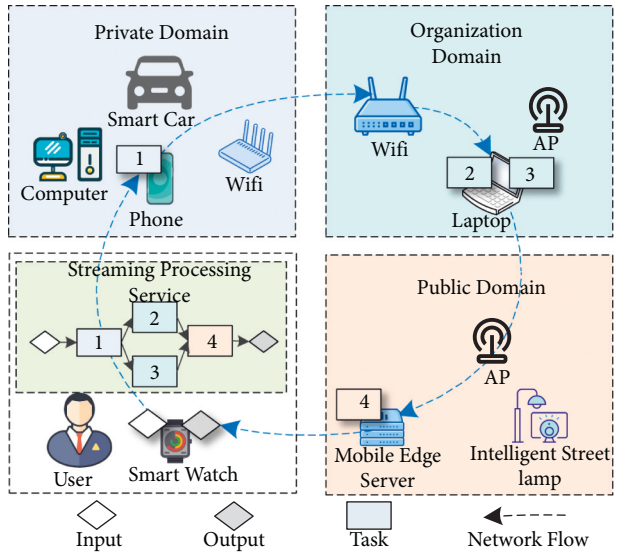

Figure 8: Security domain-based task offloading model.

When a user wants to offload the IoMT or social application to nearby devices through the DCOMP system, first, the user sends a service request to a DCC, and the DCC obtains the metadata information of the application through the trusted application discovery and acquisition mechanism described in Section 4.2. Second, the DCC calculates the security domain information of the user, that is, the list of devices contained in each domain of the user. After the security domain calculation is completed, the DCC will schedule different tasks of the application to the appropriate domain devices for execution according to the security level of each task stored in the task metadata. We will describe the process of user security domain division and computing offloading in the following subsections.

4.3.2. Security Domain Division. The geographically adjacent DCP devices are managed by a DCC. We convert the collection of the DCC and the DCP managed by it to a management domain and use $GD_j = DCC_j \cup \{DCP_{j,i} | 1 \le i \le N\}, 1 \le j \le M$ to represent the management

Input: $UInfo_u, GD_j, \{SG_{j,i}\}$
Output: $SDP_{j,u}$
(1) $PriD_{j,u}, OrgD_{j,u}, PubD_{j,u} = []$
(2) **For** $DCP_{j,i}$ **in** GD_j
(3) **if** $UIDS_{j,i} == UInfo_u$ and
(4) $Equation1 holds$ **Then**
(5) $AppendDCP_{j,i} toPriD_{j,u}$
(6) end
(7) **else if** $OIDS_{j,i} == OInfo_u$ and
(8) $1 holds$ **Then**
(9) $AppendDCP_{j,i} toOrgD_{j,u}$
(10) end
(11) else
(12) $Append\ DCP_{j,i}\ to\ PubD_{j,u}$
(13) end
(14) end
(15) $SDP_{j,u} = PriD_{j,u} \cup OrgD_{j,u} \cup PubD_{j,u}$
(16) **return** $SDP_{j,u}$

Algorithm 1: User security domain division.

domain j. Simultaneously, U and O represent user collection and organization collection, respectively. For user u, $UInfo_u = \{UID_u, OID_u, PubK_u, PubK_o\}$ is used to identify, where UID_u identifies the ID of the user, $OrgID$ represents the ID of the organization to which the user belongs, $PubK_u$ represents the user public key, and $PubK_o$ represents the public key of the user organization. For $DCP_{j,i}$, its security group is represented by $SG_{j,i} = \{UIDS_{j,i}, OIDS_{j,i}, PubG_{j,i}\}$. $UIDS_{j,i} = \{(UID_u, Sig_u) | u \in U\}$ is the user ID and array signature list of its owner, and $OIDs_{j,i} = \{(OID_u, Sig_o) | o \in O\}$ is the ID list of the organization of the user. $PubG_{j,i}$ is a Boolean value that indicates whether the device is a public device.

A noninjective and nonsurjective function dividing: $(j, u) \longrightarrow SDP_{j,u}$ is used to represent the security domain division problem of user u in the management domain j. $SDP_{j,u} = \{\mathrm{PriD}_{j,u}, OrgD_{j,u}, PubD_{j,u}\}$ represents the security domain of user u in management domain j. $\mathrm{PriD}_{j,u} = \{DCP_{j,i} | 1 \le i \le N, 1 \le j \le M\}$ represents the private domain of user u in management domain j. $OrgD_{j,u} = \{DCP_{j,i} | 1 \le i \le N, 1 \le j \le M\}$ represents the organizational domain of user u in management domain j. $PubD_{j,u} = \{DCP_{j,i} | 1 \le i \le N, 1 \le j \le M, PubG_{j,i} = True\}$ represents the public domain of user u in management domain j. The solution of the function *dividing* needs to satisfy the constraints of formulas (1) and (2), where $Sig_u \in UIDS_{j,i}$, $\{UID_u, OID_u, PubK_u, PubK_o\} \subset UInfo_u$; $PubK^e$ and $PubK^N$ are the public exponent and modulus from the public key, Pad is the padding function, and $Hash$ is the hashing function.

$$Sig_u^{PubK_u^e} = Pad(Hash(UID_u)) \left(\mathrm{mod}\ PubK_u^N\right), \quad (1)$$

$$Sig_o^{PubK_o^e} = Pad(Hash(OID_u)) \left(\mathrm{mod}\ PubK_o^N\right). \quad (2)$$

The solution process of the function dividing is shown in Algorithm 1.

4.3.3. Computing Offloading. By scheduling tasks with different security risks to appropriate devices, the security of data movement can be improved. For user u, the IoT service that it offloaded is denoted as $S_u = (Task_u, SP_u)$. $Task = \{task_{u,k} | 1 \le k \le K\}$ is a collection of tasks. $SP_u = \{SP_{u,k} | 1 \le k \le K, SP_{u,k} \in h, m, l\}$ represents the security risk of the task, where h, m, l represents the high risk, medium risk, and low security risk in task $task_{u,k}$, respectively. We use $resType = \{cpu, memory, netband\}$ to represent the resource type. This resource model is extensible, and $resType$ can contain any type of resource. The resource capacity currently available for $DCP_{j,i}$ is denoted as $DR_{j,i} = \{DCP_{j,i}^{type} | type \in resType\}$. The resource cost of task $task_{u,k}$ is modeled as $TR_{u,k} = \{tr_{u,k}^{type} | ype \in resType\}$.

A noninjective and nonsurjective function offloading: $(j, u) \longrightarrow \{DCP_{j,i}^{u,k} | 1 \le i \le N, 1 \le j \le M, \quad 1 \le k \le K\}$ is used to represent the offloading problem of service S_u in management domain j. $DCP_{j,i}^{u,k}$ means unloading task $task_{u,k}$ to $DCP_{j,i}$. The solution of the function dividing needs to satisfy the constraints of formulas (3) and (4).

$$DCP_{j,i}^{u,k} \in \begin{cases} \mathrm{PriD}_{j,u}, & if\ SP_{u,k} = h, \\ \mathrm{PriD}_{j,u} \cup OrgD_{j,u}, & if\ SP_{u,k} = m, \\ GD_j, & if\ SP_{u,k} = l, \end{cases} \quad (3)$$

$$\sum_{m \in dcpTasks_i} TR_{u,k} < DR_{u,k}, \forall i \in V. \quad (4)$$

The solution process of the computing offloading is shown in Algorithm 2.

Input: $SDP_{j,u}, S_u$
Output: $\{DCP_{j,i}^{type}\}$
(1) result = []
(2) **For** $task_{u,k}$ **in** S_u
(3) **if** $SP_{u,k} == h$ **then**
(4) **For** $DCP_{j,i}$**in**$PriD_{j,u}$
(5) **If** *formula4holds then*
(6) $AppendDCP_{j,i}^{u,k}$*toresult*
(7) **break**
(8) end
(9) **else if** $SP_{u,k} == m$ **then**
(10) **For** $DCP_{j,i}$**in**$PriD_{j,u} \cup OrgD_{j,u}$
(11) **If** *formula4holds then*
(12) $AppendDCP_{j,i}^{u,k}$*toresult*
(13) **break**
(14) end
(15) else
(16) **For** $DCP_{j,i}$**in** GD_j
(17) **If** *formula4holds then*
(18) $AppendDCP_{j,i}^{u,k}$*toresult*
(19) **break**
(20) end
(21) end
(22) end
(23) **return** $SDP_{j,u}$

Algorithm 2: User service offloading.

Table 1: Experimental server.

Platform	Type	CPU	Cores	Memory (GB)	Storage (T)
Think system ST558	Server	Intel X-Gene 4210	20	64	12

5. Evaluation

5.1. Experiment Setup. We design two experiments to verify the effectiveness of the proposed scheme.

In the first experiment, we evaluate the throughput of the consortium blockchain with various client numbers. In the second experiment, we implement the proposed algorithms with Python and evaluate the time consumption of the algorithms.

The performance parameter of our experiment server is shown in Table 1.

We implement the consortium blockchain by Hyperledger Fabric, Docker, and VMware Workstation. The communication of containers is realized by assigning the static IP of the virtual machine (VM) hosting container. Using the extra_hosts function in the Docker-Compose file can assign the IP of the VM. The configuration of the Docker cluster is shown in Table 2.

5.2. Evaluation of Services Discovery. We use *Hyperledger Caliper* to simulate the DCC to send service discovery requests to a blockchain node. In this experiment, the number of requests per second gradually increases, and the various block sizes include 128 kB, 256 kB, 512 kB, 10 24 kB, and 2048 kB. The results of this experiment are shown in Figure 9. When the block size is 512 kB and the number of requests per second increases from 10 to 120, the throughput also gradually increases from 9.9 TPS to 124.4 TPS. When the number of requests per second is greater than 120, the throughput is stable between 120.9 and 130.1. When the block size is 128 kB, the throughput is basically similar to other block sizes in the interval of 10 to 100 requests per second. However, when the number of requests per second exceeds 120, the throughput of the 128 kB block size is in the interval of 104.5 and 110. When the block size is 256 kB, 1024 kB, and 2048 kB, the throughput of the system is not much different. When the requests per second is greater than 120, the system throughput for these three block sizes falls between 117.6 TPS and 123.4 TPS. Therefore, when the

Table 2: Docker cluster configuration.

Hostname	IP	Port
Orderer0.ciat.com	192.168.255.141	7050
Orderer1.ciat.com	192.168.255.142	7050
Orderer2.ciat.com	192.168.255.143	7050
peer0.org1.ciat.com	192.168.255.141	7051
peer1.org1.ciat.com	192.168.255.142	7051
peer0.org2.ciat.com	192.168.255.143	7051
peer1.org2.ciat.com	192.168.255.144	7051

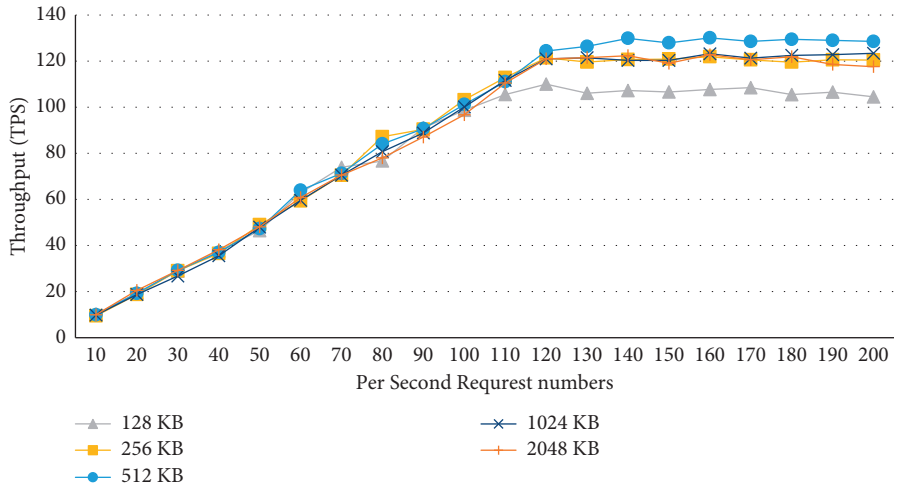

Figure 9: Throughput with various block sizes.

client request rate is below a certain range, the throughput increases as the request rate increases. When the request rate reaches a certain point, the throughput reaches the maximum and remains stable. In our experimental context, the number of points is 120, and the maximum throughput is approximately 120 TPS. Simultaneously, different block sizes impact the throughput. This experiment shows that block size affects throughput and that larger or smaller block sizes can have a negative effect on throughput. Storing the complete service in the blockchain requires a large block size, causing a severe drop in system throughput. In the proposed scheme, the blockchain only stores the metadata and hash value of the service, and its size will not exceed 512 KB. Therefore, this experiment verifies the effectiveness of the proposed method.

In addition, we evaluate the impact of the number of DCCs on throughput. DCCs are nodes in the blockchain, and we measured the throughput of blockchain networks containing 1 to 50 DCCs. The results are shown in Figure 10. When the number of DCCs is less than 10, the throughput increases as the number of clients increases. When the number of clients is 10, the throughput reaches the maximum value of 185.4 TPS. Next, the system throughput decreases as the number of clients increases. When the number of DCCs reaches 30, the system throughput drops to 175.8 TPS. When the number of DCCs reaches 50, the system throughput drops to 157.6. This experiment shows that the number of nodes has an impact on throughput and that a larger or smaller number of nodes will have a negative effect on throughput. The proposed method uses DCCs to manage many IoMT devices, so it can effectively reduce the number of nodes in the blockchain. Therefore, this experiment verifies the effectiveness of the proposed method.

Due to the hardware limitation of our experiment, the throughput does not reach a high level in the test. In practical applications, the throughput can be improved by improving the hardware configuration. In addition, the system throughput can be further improved by employing multiple channels and designing more efficient consensus algorithms.

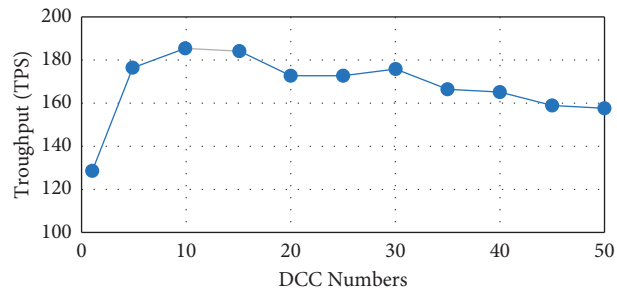

Figure 10: Throughput with the number of clients.

5.3. Evaluation of Computing Offloading. IoMT devices will move across multiple DCC management domains. When an IoMT device requests a DCC to serve, the DCC should quickly calculate the security domain and task offloading schedule. This means that the proposed security domain division and task offloading algorithm need to complete the calculation in a short time. We implement the proposed algorithm in Python and evaluate its time consumption.

The time consumption of the proposed security domain division algorithm is shown in Figure 11(a). When the number of DCC devices is 50, completing the division takes 29 milliseconds. When the number of DCC devices increases to 300, the division takes 164 milliseconds. When the number of DCC devices reaches 500, the division takes 290 milliseconds. The experimental results show that the time consumption of the proposed security domain division algorithm linearly increases with increasing DCC and has good real-time performance and expansibility.

The time consumption of the proposed task offloading algorithm is shown in Figure 11(b). In this experiment, we set the number of DCCs to 500 and measure the time consumption of the algorithms with different task numbers. When the number of tasks is 1000, the algorithm takes approximately 7 milliseconds to determine an offloading plan. When the number of tasks reached 5000, 114 milliseconds were needed. When the number of tasks is 10000, calculating the offloading scheme takes 382 milliseconds.

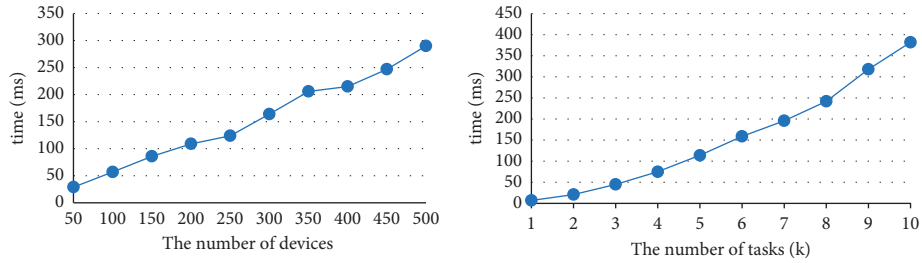

Figure 11: Time-consumption of proposed algorithms. (a) Security domain division. (b) Task offloading.

The results show that the proposed task offloading algorithm can calculate a task offloading scheme within a short time.

5.4. Discussion of the Efficiency of the Dispersed Computing Paradigm. Compared with cloud computing and edge computing, dispersed computing has better resource utilization and a smaller network load because dispersed computing utilizes the computing resources of network devices on the transmission path. In dispersed computing, not only computing terminals (cloud servers and edge servers) in cloud computing and edge computing but also personal computing devices (mobile phones, smart cars, and personal computers), and even network devices (programmable routers and switches) are responsible for computing. Network devices, similar to the in-network computing concept [52], use programmable switches and routers to perform computation and processing of data. Undoubtedly, a dispersed computing paradigm can fully utilize the distributed computing resources in the network and provide resource utilization efficiency.

This subsection discusses a method for measuring and evaluating the utilization of decentralized computing resources. First, the experimental environment should include a variety of computing terminals and programmable network devices. Programmable network devices can not only store and forward network packets but also perform logical calculations. This type of a virtual network device can be implemented on standard computing platforms using NFV technology. Next, a stream computing task is deployed, which consists of multiple modules, and the output of each module is the output of the other module. In this way, each module of the stream computing task can be deployed on devices in the network. After the transmission path of the data is determined, the modules of the streaming computing task are deployed on the devices of the transmission path. The stream computing task is deployed in a cloud server and an edge server. The resource utilization of cloud servers, edge servers, and devices should be monitored and compared where computing task modules are deployed.

6. Conclusions

In this paper, we propose a novel architecture that realizes the dispersed computing paradigm in the IoMT and social network scenarios. The necessary modules and components of the proposed architecture are provided and described in detail. Moreover, we propose a trusted application discovery and acquisition model to protect dispersed devices from untrusted applications using a blockchain-based application storage, discovery, and acquisition system. In addition, a security domain-based computing offloading model is also proposed to protect user privacy when user data are processed among multiple dispersed devices. We design a series of experiments to evaluate the proposed scheme. In future work, we will study more computationally efficient task scheduling algorithms and consensus algorithms to further improve the practicality of the proposed approach.

We also plan to extend our offloading model with better mobility-aware ability.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in National Natural Science Foundation of China (Grant no. 788 61976064), the National Key Research and Development Program of China (2021YFB2012402), and The Major Key Project of PCL (Grant nos. PCL2022AS21, PCL2021A02, and PCL2021A09).

References

[1] M. Malekshahi Rad, A. M. Rahmani, A. Sahafi, and N. Nasih Qader, "Social internet of things: vision, challenges, and trends," *Human-centric Computing and Information Sciences*, vol. 10, no. 1, p. 52, 2020.

[2] L. E. Talavera, M. Endler, I. Vasconcelos, R. Vasconcelos, M. Cunha, and F. J. Da Silva, "The mobile hub concept: enabling applications for the internet of mobile things," in *Proceedings of the 2015 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, St. Louis, MO, USA, June 2015.

[3] R. Ms, S. Pattar, R. Buyya, V. Kr, S. S. Iyengar, and L. M. Patnaik, "Social Internet of Things (SIoT): foundations,

thrust areas, systematic review and future directions," *Computer Communications*, vol. 139, pp. 32–57, 2019.

[4] A. Bhullar, A. Mancilla, A. Nijilar, and Teixeira, *The Future of mobile Computing in 2025*, 2014.

[5] X. Jia, "IRBA: an identity-based cross-domain authentication scheme for the internet of things," *Electron*, vol. 9, no. 4, p. 3, 2020.

[6] X. Li, S. Liu, F. Wu, S. Kumari, and J. J. P. C. Rodrigues, "Privacy preserving data aggregation scheme for mobile edge computing assisted IoT applications," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4755–4763, 2019.

[7] V. Sharma, I. You, K. Andersson, F. Palmieri, M. H. Rehmani, and J. Lim, "Security, privacy and trust for smart mobile-Internet of Things (M-IoT): a survey," *IEEE Access*, vol. 8, pp. 167123–167163, 2020.

[8] X. Jiang, X. Ge, J. Yu, F. Kong, X. Cheng, and R. Hao, "An efficient symmetric searchable encryption scheme for cloud storage," *J. Internet Serv. Inf. Secur.*vol. 2, no. May, pp. 1–18, 2017.

[9] R. Lu, K. Heung, A. H. Lashkari, and A. A. Ghorbani, "A lightweight privacy-preserving data aggregation scheme for fog computing-enhanced IoT," *IEEE Access*, vol. 5, pp. 3302–3312, 2017.

[10] Y. Zhao, J. Zhao, L. Jiang et al., "Privacy-preserving blockchain-based federated learning for IoT devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[11] H. Elazhary, "Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: disambiguation and research directions," *Journal of Network and Computer Applications*, vol. 128, no. October 2018, pp. 105–140, 2019.

[12] J. Pan and J. McElhannon, "Future edge cloud and edge computing for internet of things applications," *IEEE Internet of Things Journal*, vol. 5, no. 1, 2018.

[13] Z. Li and E. Peng, "Software-defined optimal computation task scheduling in vehicular edge networking," *Sensors*, vol. 21, no. 3, p. 955, 2021.

[14] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[15] C. Puliafito, E. Mingozzi, and G. Anastasi, "Fog computing for the internet of mobile things: issues and challenges," in *Proceedings of the 2017 IEEE International Conference on Smart Computing (SMARTCOMP)*, Hong Kong, China, May 2017.

[16] N. Hu, Z. Tian, X. Du, N. Guizani, and Z. Zhu, "Deep-Green: a dispersed energy-efficiency computing paradigm for green industrial IoT," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 750–764, 2021.

[17] A. Knezevic, "DEMO: CIRCE - a runtime scheduler for DAG-based dispersed computing," in *Proceedings of the 2017 2nd ACM/IEEE Symp. Edge Comput. SEC 2017*, pp. 1-2, New York, NY, USA, October 2017.

[18] H. Yang, G. Li, G. Sun et al., "Dispersed computing for tactical edge in future wars: vision, architecture, and challenges," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8899186, 31 pages, 2021.

[19] N. Zhao, J. Zou, and Y. Zhang, *A Secure Dispersed Comuting Scheme for Internet of Mobile Things*, Springer, Singapore.

[20] S. Nastic, T. Rausch, O. Scekic, S. Dustdar, M. Gusev, and B. Koteska, "A serverless real-time data analytics platform for edge computing," *IEEE Internet Comput*, vol. 21, no. 4, pp. 64–71, 2017.

[21] L. Greco, P. Ritrovato, and F. Xhafa, "An edge-stream computing infrastructure for real-time analysis of wearable sensors data," *Future Generation Computer Systems*, vol. 93, pp. 515–528, 2019.

[22] S. Nunna, A. Kousaridas, M. Ibrahim et al., "Enabling real-time context-aware collaboration through 5G and mobile edge computing," in *Proceedings of the 2015 12th International Conference on Information Technology - New Generations 2015*, pp. 601–605, Las Vegas, Nevada, USA, April 2015.

[23] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: new paradigms, scenarios, and challenges," *IEEE Communications Magazine*, vol. 55, no. 4, pp. 54–61, 2017.

[24] A. Sharma, E. S. Pilli, A. P. Mazumdar, and P. Gera, "Towards trustworthy internet of things: a survey on trust management applications and schemes," *Computer Communications*, vol. 160, pp. 475–493, 2020.

[25] J. Guo, I. R. Chen, and J. J. P. Tsai, "A survey of trust computation models for service management in internet of things systems," *Computer Communications*, vol. 97, pp. 1–14, 2017.

[26] I. Ud Din, M. Guizani, B. S. Kim, S. Hassan, and M. Khurram Khan, "Trust management techniques for the internet of things: a survey," *IEEE Access*, vol. 7, pp. 29763–29787, 2019.

[27] N. B. Truong, G. M. Lee, T. W. Um, and M. MacKay, "Trust evaluation mechanism for user recruitment in mobile crowd-sensing in the internet of things," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 10, pp. 2705–2719, 2019.

[28] T. Wang, P. Wang, S. Cai et al., "Mobile edge-enabled trust evaluation for the Internet of Things," *Information Fusion*, vol. 75, pp. 90–100, 2021.

[29] D. E. Kouicem, Y. Imine, A. Bouabdallah, and H. Lakhlef, "A decentralized blockchain-based trust management protocol for the internet of things," *IEEE Transactions on Dependable and Secure Computing*, vol. 2020, Article ID 3003232, 2020.

[30] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, 2019.

[31] M. Usman, M. A. Jan, X. He, and J. Chen, "P2DCA: a privacy-preserving-based data collection and analysis framework for IoMT applications," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1222–1230, 2019.

[32] J. Hiller, J. Pennekamp, M. Dahlmanns, M. Henze, A. Panchenko, and K. Wehrle, "Tailoring onion routing to the internet of things: security and privacy in untrusted environments," in *Proceedings of the 2019 IEEE 27th International Conference on Network Protocols (ICNP)*, Chicago, IL, USA, October 2019.

[33] X. He, R. Jin, and H. Dai, "Deep PDS-learning for privacy-aware offloading in MEC-enabled IoT," *IEEE Internet of Things Journal*, vol. 6, no. 3, 2019.

[34] C. S. Yang, R. Pedarsani, and A. S. Avestimehr, "Communication-aware scheduling of serial tasks for dispersed computing," *IEEE/ACM Transactions on Networking*, vol. 27, no. 4, pp. 1330–1343, 2019.

[35] D. Hu and B. Krishnamachari, "Throughput optimized scheduler for dispersed computing systems," in *Proceedings of the 2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud) 2019*, pp. 76–84, Newark, CA, USA, April 2019.

[36] P. Rahimzadeh, J. Lee, Y. Im et al., "Sparcle: stream processing applications over dispersed computing networks," in

Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), vol. 2020, pp. 1067–1078, Singapore, November 2020.

[37] H. Wu, J. Zhang, Z. Cai et al., "Resolving multi-task competition for constrained resources in dispersed computing: a bilateral matching game," *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 16972–16983, 2021.

[38] M. García-Valls, A. Dubey, and V. Botti, "Introducing the new paradigm of social dispersed computing: applications, technologies and challenges," *Journal of Systems Architecture*, vol. 91, pp. 83–102, 2018.

[39] *Dispersed Computing*, https://www.darpa.mil/program/dispersed-computing.

[40] M. R. Schurgot, M. Wang, A. E. Conway, L. G. Greenwald, and P. D. Lebling, "A dispersed computing architecture for resource-centric computation and communication," *IEEE Communications Magazine*, vol. 57, no. 7, pp. 13–19, 2019.

[41] A. E. Conway, M. Wang, E. Ljuca, and P. D. Lebling, "A dynamic transport overlay system for mission-oriented dispersed computing over IoBT," *MILCOM 2019 - 2019 IEEE Military Communications Conference (MILCOM)*, vol. 2019, Article ID 9020772, 820 pages, 2019.

[42] C. S. Yang, A. S. Avestimehr, and R. Pedarsani, "Communication-aware scheduling of serial tasks for dispersed computing," in *Proceedings of the 2018 IEEE International Symposium on Information Theory (ISIT)*, vol. 2018, pp. 1226–1230, Vail, Colorado, June 2018.

[43] C. S. Shih, W. Y. Hsieh, and C. L. Kao, "Traceability for vehicular network real-time messaging based on blockchain technology," *J. Wirel. Mob. Networks, Ubiquitous Comput. Dependable Appl*, vol. 10, no. 4, 2019.

[44] C. N. Ribalta, M. Lombard-Platet, C. Salinesi, and P. Lafourcade, "Blockchain mirage or silver bullet? A requirements-driven comparative analysis of business and developers' perceptions in the accountancy domain," *J. Wirel. Mob. Networks, Ubiquitous Comput. Dependable Appl*, vol. 12, no. 1, 2021.

[45] M. Alizadeh, K. Andersson, and O. Schelen, "A survey of secure internet of things in relation to blockchain," *J. Internet Serv. Inf. Secur*, vol. 10, no. 3, 2020.

[46] L. König, S. Unger, P. Kieseberg, and S. Tjoa, "The risks of the blockchain a review on current vulnerabilities and attacks," *J. Internet Serv. Inf. Secur*, vol. 10, no. 3, 2020.

[47] N. Hu, Z. Tian, Y. Sun et al., "Building agile and resilient UAV networks based on SDN and blockchain," *IEEE Network*, vol. 35, no. 1, pp. 57–63, 2021.

[48] N. Hu, S. Yin, S. Su, X. Jia, Q. Xiang, and H. Liu, "Blockzone: a decentralized and trustworthy data plane for DNS," *Computers, Materials & Continua*, vol. 65, no. 2, pp. 1531–1557, 2020.

[49] N. Hu, Y. Teng, Y. Zhao, S. Yin, and Y. Zhao, "IDV: internet domain name verification based on blockchain," *Computer Modeling in Engineering and Sciences*, vol. 129, no. 1, pp. 299–322, 2021.

[50] X. Jia, N. Hu, S. Yin, Y. Zhao, C. Zhang, and X. Cheng, "A2Chain: a blockchain-based decentralized authentication scheme for 5G-enabled IoT," *Mobile Information Systems*, vol. 2020, Article ID 8889192, 19 pages, 2020.

[51] I. Baumgart and S. Mies, "IPFS - content addressed, versioned, P2P file system (DRAFT 3)," in *Proceedings of the Int. Conf. Parallel Distrib. Syst. - ICPADS*, Seoul, Korea, December 2007.

[52] N. Hu, Z. Tian, X. Du, and M. Guizani, "An energy-efficient in-network computing paradigm for 6G," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 4, pp. 1722–1733, 2021.