






A Secure Dispersed Computing Scheme for Internet of Mobile Things

Yan Zhao , Ning Hu  , Jincui Zou , and Yuqiang Zhang 

Cyberspace Institute of Advanced Technology, Guangzhou University,
Guangzhou 510006, China
{2111906107,2112006304,2112006277}@e.gzhu.edu.cn, huning@gzhu.edu.cn

Abstract. With the rapid development of the Internet of Mobile Things (IoMT) and countless mobile devices connected to the Internet, problems of the IoMT computing paradigm, Mobile Cloud Computing (MCC) and Mobile Edge Computing (MEC), have been aggravated in terms of network congestion, high energy consumption, and huge investment cost. Dispersed computing utilize geographically distribute computing resources and the “code and data” strategy movement, reducing energy and investment costs and improving the performance of IoMT applications. However, security problems raise as the “code and data” movement among multiple devices. We analyze the security threats in IoMT dispersed computing and propose a secure dispersed computing scheme for IoMT. The proposed scheme improves the availability, integrity, confidentiality, and privacy of IoMT dispersed computing by a decentralized service discovery and distribution model and a security domain-based task offloading scheme. The experiment results prove the effectiveness of the proposed scheme.

Keywords: Dispersed computing · Blockchain · Privacy

1 Introduction

The Internet of Mobile Things (IoMT), a subset of the Internet of Things, focuses on connecting mobile devices, such as smartphones, vehicles, and wearable devices, to the Internet [21]. The interconnection and collaboration of IoMT devices by the Internet will create a smarter world. According to a forecast, more than 50 billion mobile devices will connect to the Internet by 2025 [3].

Effective computing paradigms are needed to serve such massive IoMT devices. Nowadays, the computing paradigm of IoMT mainly includes Mobile Cloud Computing (MCC) and Mobile Edge Computing (MEC) [5]. MCC uses the cloud as a data storage and computing platform to provide global interconnection of mobile devices. However, The massive traffic of MCC not only makes the core network congested but also affects the real-time performance of mobile applications. MEC offload IoMT applications to the edge server, alleviating the network congestion and poor real-time performance. However, a large number

of mobile edge servers need to be deployed to support a large number of mobile devices, which leads to huge investment costs and energy consumption [9, 10].

IoMT dispersed computing is a promising way to solve the above problems. The basic idea of the dispersed computing paradigm is to realize the “code and data” on-demand movement by designing new programmable protocols and algorithms, enabling the securely and collectively execution of tasks on geographically dispersed, ubiquitous, and heterogeneous computing platforms [4, 6]. These dispersed platforms include network elements, intelligent terminals, programmable sensors, and edge servers. In an IoMT dispersed computing scenario, users can use “local” or “nearby” available computing resources to carry out the application, significantly improving application and network performance while reducing energy consumption and investment costs [10]. However, the security of IoMT dispersed computing has not been studied in depth. The security of the “code and data” movement needs to be guaranteed.

In this paper, we discuss the security issues of IoMT dispersed computing. In response to these security issues, A secure dispersed computing scheme for the internet of mobile things is proposed. The proposed method realizes a decentralized service publication, discovery, and distribution mechanism through the consortium blockchain, which improves the usability, integrity, and authenticity of code movement. In addition, a computational offloading method based on security domain division is proposed, which improves the security of data movement by offloading tasks with different security risks to different security domains. The main contributions of this article are as follows:

- We analyze the security issues involved in the movement of “code-data” in the IoMT dispersed computing.
- We propose a decentralized service publication, discovery and distribution scheme. The proposed method applies the consortium blockchain and IPFS distributed file system to IoMT dispersed computing, ensuring the availability, integrity and authenticity of the code publication, discovery and distribution process.
- We propose a security domain-based computing offloading scheme. For different users, the proposed method group dispersed devices into different domains. By scheduling IoMT tasks with different security risks to appreciate security domain devices, the confidentiality and privacy of data transmission, storage and processing are guaranteed.

The rest of the paper is organized as follows. The overview of related work is presented in Sect. 2. The motivation and objectives of our work are described in Sect. 3. A Secure Dispersed Computing Scheme for Internet of Mobile Things is elaborated in Sect. 4. The results of experiment of the proposed scheme and algorithm are discussed in Sect. 5. Finally, in Sect. 6, the conclusion of this work are presented.

2 Related Works

Although the security of internet of mobile things has been studied widely [17, 18, 22, 23], there is little research on the security of distributed computing, especially in the scenario of mobile Internet of things.

The research of dispersed computing paradigm is mainly focused on architecture and task scheduling. Schurgot et al. [20] describes the architecture of dispersed computing. The architecture consists of application layer, dispersed computing layer and physical layer. The application layer consists of applications and dispersed computing API for submitting tasks. The dispersed computing layer consists of dispersed computing task-aware computation, programmable nodes and network protocol stacks. Dispersed computing task-aware computing algorithms share transfer task details, data flow details, performance boundaries, and may even share job task graphs under the middleware “stack”. Programmable nodes and protocol stacks provide NCP lists, overlay network abstractions, flow performance details. The physical layer is various underlying networks.

Dispersed computing task scheduling mainly studies the task unloading problem of stream processing application modeled by Directed Acyclic Graph (DAG). Different from edge computing offloading, task offloading in the dispersed computing paradigm needs to jointly optimize computing offloading and network scheduling. Yang et al. [25] study the problem of chained task scheduling in dispersed computing networks. They propose a novel virtual queuing network encoding the state of the network and a Max-Weight type scheduling policy for the virtual queuing network. Rahimzadeh et al. [19] propose a scheduling system framework, SPARCLE, for stream processing applications in dispersed computing networks. The proposed method can complete the task assignment and resource allocation of a stream processing application in polynomial time. H. Wu et al. [7] proposed a dispersed computing offloading framework, formal the offloading problem into a multi-objective optimization problem, and design a bilateral matching algorithm to obtain the optimal task offloading strategy.

3 Motivation and Objectives

Figure 1 shows the IoMT dispersed computing paradigm [20, 26]. The IoMT dispersed computing includes the IoMT service provider, dispersed task-aware computation, and programmable nodes and protocol stacks. Its workflow is: (1) The IoMT device sends a service request which contains the service identifier to the dispersed task-aware computing module; (2) The dispersed task-aware computing module sends a request to the service discovery module; (3) The discovery module returns the service to the dispersed task-aware computing module; (4) The dispersed task-aware computing module calculates the task offloading plan, and offloads tasks of the services and flow tables to the appropriate DCP; (5) The IoMT device sends data to the first DCP.

Although dispersed computing improves the responsiveness, reliability, real-time performance, and availability of IoMT applications through the “code and data” strategy movement and “forwarding and computing hop-by-hop”, there are the following security issues of IoMT dispersed computing.

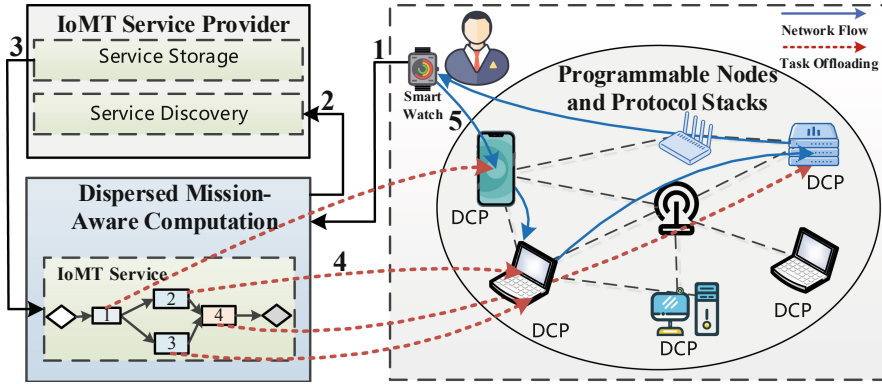


Fig. 1. IoMT dispersed computing paradigm

1. **Availability of Service Discovery**

In the IoMT scenario, the code of the application or service is usually not stored in the resource-constrained IoT device but the cloud or the hosting server of service providers. Moreover, service providers will provide a centralized service discovery mechanism. A simple way is to use this centralized mechanism directly by the dispersed task-aware computing module. However, the inherent availability limitations of the centralized system, such as single points of failure, network bandwidth and service capacity bottlenecks, contradict the high availability promised by dispersed computing.

2. **Integrity and Authenticity of Services Distribution**

During the code distribution, the service code may be maliciously tampered with or replaced. Running malicious code threatens to not only user data but also nodes themselves. In addition, the order of task execution may also be maliciously tampered with, causing code to be executed in an unexpected location and resulting in unavailability of services or leakage of user privacy.

3. **Confidentiality of Data Transmission**

Due to the limited computing resources and battery life of some IoMT devices (such as smart bracelets, smart sensors, wireless headsets, etc.), it is hard to deploy heavy encryption algorithms. Data is easily sniffed maliciously during the transmission process between dispersed computing devices.

4. **Privacy of Data Storage and Processing**

User data needs to be stored and processed on some uncontrolled devices. Storing unencrypted data on these devices will lead to a privacy leak. Even data are encrypted, it also needs to be decrypted before or during processing, which means there is still a problem of user privacy leakage if data needs to be processed on uncontrolled devices.

According to the above analysis, there are many security problems in IoMT dispersed computing. In the “code and data” movement, the availability, integrity, and authenticity of code publication, discovery, and distribution, as well as the confidentiality and privacy of data transmission, storage, and processing need

to be guaranteed. To the best of our knowledge, Existing researches on dispersed computing mainly focuses on architecture design and task scheduling [7, 15, 19, 24–26], and there are little researches on security solutions in the IoMT dispersed computing scenario. Therefore, in this article, we propose a secure dispersed computing scheme for the Internet of Mobile Things, which has the following design objectives:

- Improve the availability, integrity and authenticity of service publication, discovery and distribution in IoMT dispersed computing.
- Improve the confidentiality and privacy of data transmission, storage and processing in IoMT dispersed computing.

4 Proposed Scheme

For the above two objectives, we propose a secure dispersed computing paradigm for the Internet of Mobile Things in this section. The proposed scheme includes a decentralized service publication and discovery scheme and a security domain-based task offloading model. The first scheme improves the availability of service publication and discovery by employing the consortium blockchain. Meanwhile, it also provides integrity and authenticity verification capabilities. The second scheme groups dispersed nodes into the private, organization, and public domains. By scheduling tasks with different security priorities to appropriate domain nodes, the confidentiality, integrity, and privacy of data transmission, storage, and processing are guaranteed as much as possible. We will describe the detail of the proposed two scheme in Sects. 4.1 and 4.2.

4.1 A Decentralized Service Publication, Discovery and Distribution Scheme

A decentralized service publication, discovery, and distribution scheme for IoMT dispersed computing is described in this section. Blockchain has been used successfully for many systems [1, 8, 11–14, 16]. We argue that blockchain is a promising technology to realizes a high-availability service discovery and high-trust code distribution of IoMT dispersed computing. The basic idea of the proposed scheme is to use the blockchain to store the metadata of IoMT services and to use the distributed file system to store service codes and models. IoMT service providers jointly manage and maintain the consortium blockchain. The distributed file system uses IPFS, which is a media protocol based on blockchain and uses distributed storage and content addressing technology [2]. IPFS address files according to their content hash value. Service providers publish service metadata to the blockchain and publish service codes to IPFS. Users initiate requests to the blockchain to obtain service metadata and requests to IPFS to obtain service codes. The proposed method improves the availability of service publication, discovery, and distribution by a decentralized design.

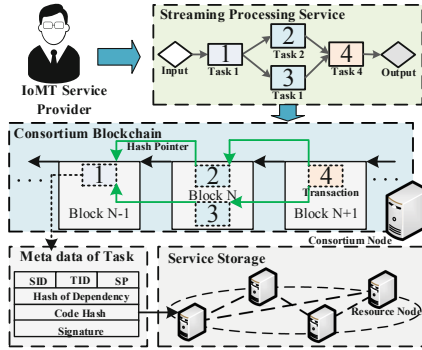


Fig. 2. A decentralized service publication and storage model

Service Publication. Figure 2 shows the proposed service publication model. The IoMT services can be modeled as a Directed Acyclic Graph (DAG) in which the nodes represent the tasks and the edge represent the dependencies [15, 26]. This model indicates the number and the order of the tasks. Meanwhile, it also makes the storage and verification of IoMT service stored in the blockchain and IPFS simple.

Due to the limitation of block capacity, the consortium blockchain only stores the service metadata. The metadata includes service identification (SID), task identification (TID), task security priority (SP), task-dependency hash, code hash, and digital signature of the service provider. SID is globally unique by a Globally Unique Identifier (GUID) algorithm. All service providers in the consortium blockchain use the same GUID algorithm. TID is the index of the topological sorting of tasks in a service. The metadata of each task is encapsulated as a transaction and published to the consortium blockchain. The transaction is published in the order of the topological sorting. The service provider first initiates task transactions that have no forward dependencies. After all consortium nodes reach a consensus on these tasks transactions, the service provider will continue to initiate subsequent task transactions. Each task transaction points to the transaction of the task it depends on through the hash pointer.

The service code is stored in IPFS, and the IPFS network is composed of resource servers of various service providers. IPFS uses the hash value of the file as the index of the file. The hash value of each task code is stored in the metadata. The user can quickly read and obtain the service code through the service metadata.

Service Discovery and Distribution. As shown in Fig. 3, the service publication and distribution system include consortium blockchain, IPFS, control layer, and computing layer. The consortium blockchain is used to store the metadata of services and the IPFS is responsible for storing service data and code. The computing layer consists of dispersed computing devices called DCP. The control layer includes a multiple dispersed computing controller (DCC) and a DCC that

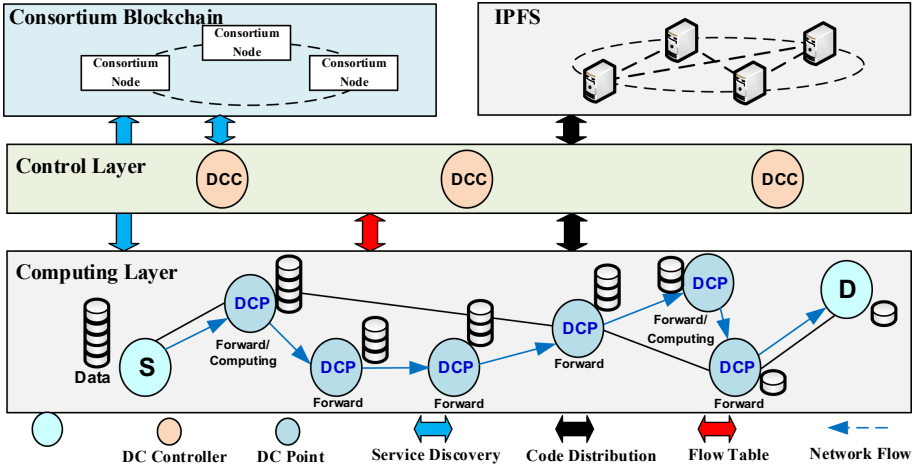


Fig. 3. A decentralized service discovery and distribution model

manages a set of DCP. The user device first initiates a request to a DCC. The DCC receives the request and obtains the SID. Then, DCC initiates a service discovery request to a consortium blockchain node. After receiving the request, the consortium node gets the service ID from the request, queries all transactions whose SID is equal to this ID, and verifies the hash pointer of each transaction. After obtaining the metadata of all tasks, the consortium node will package this metadata into a service metadata and return it to DCC. Then, DCC parses the service metadata, obtains the hash value of all task codes, and then sends a request to IPFS to obtain the task code. Finally, DCC offloads the service to a suitable DCP for execution according to the scheduling strategy.

Availability, Integrity and Authenticity. First, the proposed method is based on the consortium blockchain and IPFS, and adopts a completely decentralized design, so that the system does not have a single point of failure. This method greatly improves the availability of service publication and discovery. At the same time, based on the IPFS decentralized storage system, service codes can be stored on multiple resource servers. Even if some consortium nodes or resource nodes fail, the service publication and discovery functions can still operate normally. Secondly, the metadata of the service is stored in the block of the consortium blockchain as transactions of tasks. In addition, the integrity of the dependencies of tasks in the service is protected by hash pointers between transactions. Therefore, any attempt to tamper with service metadata must simultaneously modify the entire blockchain. The task code is stored in IPFS, and the storage address is the code hash in the task metadata. The non-tampering feature of the blockchain can ensure the integrity of the service metadata and service code. Finally, the authenticity is guaranteed by the digital signature in the task data metadata. After DCC receives the service metadata returned by

the consortium node, it uses the public key of the service provider to check the digital signature fields of all task metadata.

4.2 A Security Domain Division Based Computing Offloading Scheme

Offloading data or computing to the arbitrary DCP has the risk of privacy leakage. This section proposes a security domain-based computing offloading method. As shown in Fig. 4, for a user, DCC divides the DCP in its management domain into private domains, organizational domains, and public domains. By scheduling tasks with different security priorities to appropriate security domains, the confidentiality and privacy of data movement can be guaranteed.

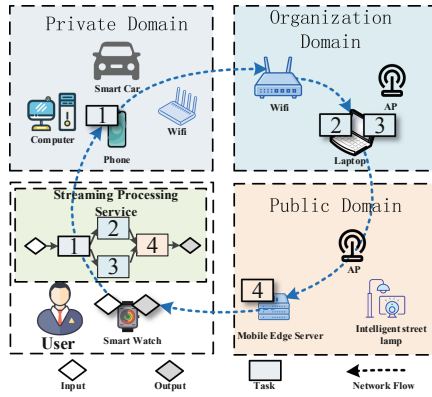


Fig. 4. Security domain division based task offloading model

Security Domain Division. The geographically adjacent DCP are managed by a DCC. A management domain include a DCC and certain DC. We use $MD_j = DCC_j \cup \{DCP_{j,i} \mid 1 \leq i \leq N\}$, $1 \leq j \leq M$ to represent the management domain j . U and O are used to represent user and organization set respectively. For a user u , using $UInfo_u = \{UID_u, OrgID_u, PubK_u, PubK_o\}$ to represent, where UID_u is the user ID, $OrgID$ is the user organization ID, $PubK_u$ represents the user public key, and $PubK_o$ represents the user organization public. For $DCP_{j,i}$, its security group is represented by $SG_{j,i} = \{UIDS_{j,i}, OIDS_{j,i}, PubG_{j,i}\}$. $UIDS_{j,i} = \{(UID_u, Sig_u) \mid u \in U\}$ is the user ID and array signature list of its owner, and $OIDS_{j,i} = \{(OID_u, Sig_o) \mid o \in O\}$ 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 non-injective and non-surjective function $dividing : (j, u) \rightarrow SDP_{j,u}$ is used to represent the security domain division problem of the user u in the management domain j . The $SDP_{j,u} = \{PriD_{j,u}, OrgD_{j,u}, PubD_{j,u}\}$ represents the

security domain of the user u in the management domain j . The $PriD_{j,u} = \{DCP_{j,i} \mid 1 \leq i \leq N, 1 \leq j \leq M\}$ represents the private domain of the user u in the management domain j . The $OrgD_{j,u} = \{DCP_{j,i} \mid 1 \leq i \leq N, 1 \leq j \leq M\}$ represents the organizational domain of the user u in the management domain j . And the $PubD_{j,u} = \{DCP_{j,i} \mid 1 \leq i \leq N, 1 \leq j \leq M\}$, $PubG_{j,i} = True\}$ represents the public domain of the user u in the 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.

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

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

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

Algorithm 1: Security Domain Division

Input: $UInfo_u, MD_j, SG_{j,i}$
Output: $SDP_{j,u}$

- 1 $PriD_{j,u}, OrgD_{j,u}, PubD_{j,u} = []$;
- 2 **for** $DCP_{j,i}$ **in** MD_j **do**
- 3 **if** $UIDS_{j,i} == UInfo_u$ **and the equation (1) holds then**
- 4 | Append $DCP_{j,i}$ to $PriD_{j,u}$;
- 5 **end**
- 6 **else if** $OIDS_{j,i} == OInfo_u$ **and the equation (1) holds then**
- 7 | Append $DCP_{j,i}$ to $OrgD_{j,u}$;
- 8 **end**
- 9 **else**
- 10 | Append $DCP_{j,i}$ to $PubD_{j,u}$;
- 11 **end**
- 12 **end**
- 13 $SDP_{j,u} = PriD_{j,u} \cup OrgD_{j,u} \cup PubD_{j,u}$;
- 14 **return** $SDP_{j,u}$

Task Offloading. By scheduling tasks with different security risks to appropriate DCC, the security of data movement can be improved. For a user u , the IoMT service needed to offload is denoted as $S_u = (Task_u, SP_u)$. $Task = \{task_{u,k} \mid 1 \leq k \leq K\}$ is a set of tasks. $SP_u = \{SP_{u,k} \mid 1 \leq k \leq K, SP_{u,k} \in h, m, l\}$ represents the task security risk, where h, m, l represents the high, medium, and low security risks of task $task_{u,k}$, respectively. We use $resType = (cpu, memory, netband)$ to represent the resource type. The resource capacity currently available for $DR_{j,i} =$

$\{DCP_{j,i}^{type} | type \in resType\}$. The resource cost of task $task_{u,k}$ is modeled as $TR_{u,k} = \{tr_{u,k}^{type} | type \in resType\}$.

A non-injective and non-surjective function $offloading : (j, u) \rightarrow \{DCP_{j,i}^{u,k} | 1 \leq i \leq N, 1 \leq j \leq M, 1 \leq k \leq K\}$ is used to represent the offloading problem of service S_u in management domain j . $DCP_{j,i}^{u,k}$ means to offload task $task_{u,k}$ to $DCP_{j,i}$. The solution of the function dividing needs to satisfy the constraints of formulas (3) and (4).

$$DCP_{j,i}^{u,k} \in \begin{cases} PriD_{j,u}, if SP_{u,k} = h \\ PriD_{j,u} \cup OrgD_{j,u}, if SP_{u,k} = m \\ MD_j, if SP_{u,k} = l \end{cases} \quad (3)$$

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

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

Confidentiality, Integrity and Privacy. The proposed method offloads tasks with different security risks to devices in different security domains near the user. Tasks with high security risks are offloaded to the user's private device, which ensures the confidentiality and privacy of data storage and data processing.

5 Evaluation

we implement proposed algorithms with Python and evaluate the time consumption of the algorithms. The performance parameter of our experiment server is shown in Table 1.

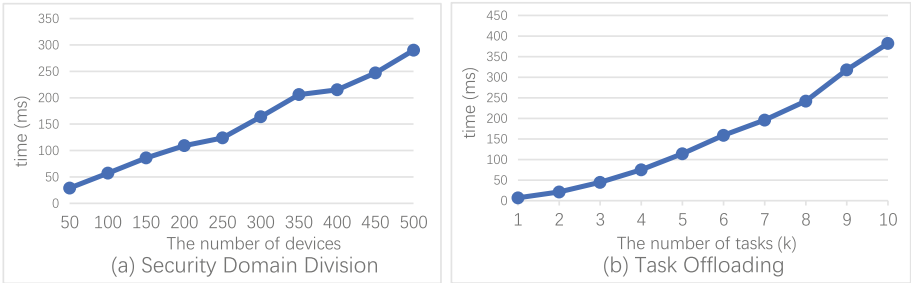


Fig. 5. The time-consuming of the proposed security domain division and computing offloading algorithm

IoMT devices will move across multiple DCC management domains. When an IoMT device requests a DCC to serve, the DCC should quickly calculate

Algorithm 2: Task Offloading

```

Input:  $SDP_{j,u}, S_u$ 
Output:  $\{DCP_{j,i}^{type}\}$ 
1 result = [ ] ;
2 for  $task_{u,k}$  in  $S_u$  do
3   if  $SP_{u,k} == h$  then
4     for  $DCP_{j,i}$  in  $PriD_{j,u}$  do
5       if the formula (4) holds then
6         Append  $DCP_{j,i}^{u,k}$  to result;
7         break;
8       end
9     end
10  end
11  else if  $SP_{u,k} == m$  then
12    for  $DCP_{j,i}$  in  $OrgD_{j,u} \cup OrgD_{j,u}$  do
13      if the formula (4) holds then
14        Append  $DCP_{j,i}^{u,k}$  to result;
15        break;
16      end
17    end
18  end
19  else
20    for  $DCP_{j,i}$  in  $PubD_{j,u} \cup OrgD_{j,u}$  do
21      if the formula (4) holds then
22        Append  $DCP_{j,i}^{u,k}$  to result;
23        break;
24      end
25    end
26  end
27 end
28 return result

```

the security domain and task offloading schedule. It 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-consuming.

The time-consuming of the proposed security domain division algorithm is shown in Fig. 5 (a). When the number of DCC devices is 50, completing the

Table 1. Configuration of the edge server

Platform	CPU	Cores	Memory	Storage
ST558	X-Gen 4210	20	64 GB	12T

division takes 29 ms. When the number of DCC devices increases to 300, the division takes 164 ms. When the number of DCC devices reaches 500, it takes 290 ms. The experimental results show that the time consuming of the proposed security domain division algorithm increases linearly with the increase of DCC, and has good real-time performance and expansibility.

The time-consuming of the proposed task offloading algorithm is shown in Fig. 5 (b). In this experiment, we set the number of DCC to 500 and measure the time consumption of the algorithms under different tasks numbers. When the number of tasks is 1000, the algorithm takes about 7 ms to determine an offloading plan. When the number of tasks reaches 5000, it took 114 ms. When the number of tasks is 10000, calculating the offloading scheme takes 382 ms. The results show that the proposed task offloading algorithm can calculate a task offloading scheme in a short time.

6 Conclusion

In this paper, we discuss the security problems in IoMT dispersed computing paradigm and propose a secure dispersed computing scheme for the Internet of Mobile Things. The proposed scheme realizes a decentralized service publication, discovery, and distribution mechanism by employing consortium blockchain, which improves the availability and integrity of service discovery and distribution of dispersed computing systems. Moreover, we also propose a security domain division-based task offloading scheme. By offloading tasks with different security risks to devices in the different security domains, it can protect the confidentiality and privacy of user data as much as possible. Finally, we design a series of experiments to evaluate the proposed scheme. The experiment results prove the effectiveness of our scheme.

Acknowledgments. This work was supported in National Natural Science Foundation of China (Grant No. 61976064), National Defence Science and Technology Key Laboratory Fund (61421190306), Guangzhou Science and Technology Plan Project (202102010471), and Guangdong Province Science and Technology Planning Project (2020A1414010370).

References

1. Alizadeh, M., Andersson, K., Schelén, O.: A survey of secure internet of things in relation to blockchain. *J. Internet Serv. Inf. Secur. (JISIS)* **10**(3), 47–75 (2020)
2. Baumgart, I., Mies, S.: IPFS - content addressed, versioned, P2P file system (DRAFT 3). In: *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS* (2007)
3. Bhullar, J., Mancilla, A., Nijilar, A., Teixeira: the future of mobile computing in 2025. Technical report (2014)
4. Dispersed Computing: DARPA-BAA-16-41 1, 1–44 (2016)
5. Elazhary, H.: Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: disambiguation and research directions (2019). <https://doi.org/10.1016/j.jnca.2018.10.021>

6. Garcia-Valls, M., Dubey, A., Botti, V.: Introducing the new paradigm of social dispersed computing: applications, technologies and challenges. *J. Syst. Architect.* **91**, 83–102 (2018). <https://doi.org/10.1016/j.sysarc.2018.05.007>
7. Hu, D., Krishnamachari, B.: Throughput optimized scheduler for dispersed computing systems. In: *Proceedings - 2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering, MobileCloud 2019* (2019). <https://doi.org/10.1109/MobileCloud.2019.00018>
8. Hu, N., Teng, Y., Zhao, Y., Yin, S., Zhao, Y.: IDV: internet domain name verification based on blockchain. *CMES-Comput. Model. Eng. Sci.* **129**(1), 299–322 (2021)
9. Hu, N., Tian, Z., Du, X., Guizani, M.: An energy-efficient in-network computing paradigm for 6g. *IEEE Trans. Green Commun. Netw.* **5**, 1722–1733 (2021)
10. Hu, N., Tian, Z., Du, X., Guizani, N., Zhu, Z.: Deep-green: a dispersed energy-efficiency computing paradigm for green industrial IoT. *IEEE Trans. Green Commun. Netw.* (2021). <https://doi.org/10.1109/TGCN.2021.3064683>
11. Hu, N., et al.: Building agile and resilient UAV networks based on SDN and blockchain. *IEEE Netw.* **35**(1), 57–63 (2021)
12. Hu, N., Yin, S., Su, S., Jia, X., Xiang, Q., Liu, H.: Blockzone: a decentralized and trustworthy data plane for DNS. *CMC-Comput. Mater. Continua* **65**(2), 1531–1557 (2020)
13. Jia, X., et al.: IRBA: an identity-based cross-domain authentication scheme for the internet of things. *Electronics* **9**(4), 634 (2020)
14. Jia, X., Hu, N., Yin, S., Zhao, Y., Zhang, C., Cheng, X.: A2 chain: a blockchain-based decentralized authentication scheme for 5g-enabled IoT. In: *Mobile Information Systems 2020* (2020)
15. Knezevic, A., et al.: DEMO: CIRCE - a runtime scheduler for DAG-based dispersed computing. In: *2017 2nd ACM/IEEE Symposium on Edge Computing, SEC 2017* (2017). <https://doi.org/10.1145/3132211.3132451>
16. König, L., Unger, S., Kieseberg, P., Tjoa, S., Josef Ressel Center for Blockchain: The risks of the blockchain a review on current vulnerabilities and attacks. *J. Internet Serv. Inf. Secur. (JISIS)* **10**(3), 110–127 (2020)
17. Liu, N., Yu, M., Zang, W., Sandhu, R.: Cost and effectiveness of trustzone defense and side-channel attack on arm platform. *J. Wirel. Mob. Netw. Ubiquit. Comput. Dependable Appl. (JoWUA)* **11**(4), 1–15 (2020)
18. Marra, A.L., Martinelli, F., Mercaldo, F., Saracino, A., Sheikhalishahi, M.: D-BRIDEMAID: a distributed framework for collaborative and dynamic analysis of Android malware. *J. Wirel. Mob. Netw. Ubiquit. Comput. Dependable Appl. (JoWUA)* **11**(3), 1–28 (2020)
19. Rahimzadeh, P., et al.: SPARCLE: stream processing applications over dispersed computing networks. In: *Proceedings - International Conference on Distributed Computing Systems* (2020). <https://doi.org/10.1109/ICDCS47774.2020.00112>
20. Schurgot, M.R., Wang, M., Conway, A.E., Greenwald, L.G., Lebling, P.D.: A dispersed computing architecture for resource-centric computation and communication. *IEEE Commun. Mag.* (2019). <https://doi.org/10.1109/MCOM.2019.1800776>
21. Talavera, L.E., Endler, M., Vasconcelos, I., Vasconcelos, R., Cunha, M., Da Silva Silva, F.J.: The mobile hub concept: enabling applications for the internet of mobile things. In: *2015 IEEE International Conference on Pervasive Computing and Communication Workshops, PerCom Workshops 2015* (2015). <https://doi.org/10.1109/PERCOMW.2015.7134005>
22. Talegaon, S., Krishnan, R.: Administrative models for role based access control in Android. *J. Internet Serv. Inf. Secur. (JISIS)* **10**(3), 31–46 (2020)

23. Wong, S.K., Yiu, S.M.: Location spoofing attack detection with pre-installed sensors in mobile devices. *J. Wirel. Mob. Netw. Ubiquit. Comput. Dependable Appl. (JoWUA)* **11**(4), 16–30 (2020)
24. Wu, H., et al.: Resolving multi-task competition for constrained resources in dispersed computing: a bilateral matching game. *IEEE Internet Things J.* (2021). <https://doi.org/10.1109/JIOT.2021.3075673>
25. Yang, C.S., Pedarsani, R., Avestimehr, A.S.: Communication-aware scheduling of serial tasks for dispersed computing. *IEEE/ACM Trans. Netw.* (2019). <https://doi.org/10.1109/TNET.2019.2919553>
26. Yang, H., et al.: Dispersed computing for tactical edge in future wars: vision, architecture, and challenges. *Wirel. Commun. Mob. Comput.* (2021). <https://doi.org/10.1155/2021/8899186>