International Journal of Innovative Computing, Information and Control
Volume **15**, Number **2**, April **2019**

ICIC International ©2019 ISSN 1349-4198
pp. 713–723

THE FIRST PROTOCOL OF REACHING CONSENSUS UNDER UNRELIABLE MOBILE EDGE COMPUTING PARADIGM

Shu-Ching Wang[1], Wei-Shu Hsiung[1], Kuo-Qin Yan[1,*] and Yao-Te Tsai[2,*]

[1]Department of Information Management
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng, Taichung 41349, Taiwan

[2]Department of International Business
Feng Chia University
100 Wenhwa Rd., Seatwen, Taichung 40724, Taiwan

Received June 2018; revised October 2018

Abstract. *Mobile Edge Computing (MEC) is an emerging technology that enables computing directly at the edge of the cloud computing network. Therefore, it is important that MEC is applied with reliable transmission. The problem of reaching consensus in the distributed system is one of the most important issues in designing a reliable transmission network. However, all previous protocols for the consensus problem are not suitable for an MEC paradigm. It is the first time an optimal protocol of reaching consensus is proposed for MEC paradigm. The protocol makes all fault-free nodes communicate with each other and collect the exchanged messages to decide a common value. Based on the common value, the protocol ensures all fault-free nodes reach consensus without the influence of unreliable transmission. Finally, we proved theoretically that the proposed protocol can tolerate the maximum number of faulty components and using only two rounds of message exchanges.*

Keywords: Internet of Things, Mobile edge computing, Cloud computing, Consensus

1. Introduction.

In recent years, a new trend in computing is happening with the function of clouds being increasingly moving towards the network edges [1]. Harvesting the vast amount of the idle computation power and storage space distributed at the network edges can yield sufficient capacities for performing computation-intensive and latency-critical tasks at mobile devices. This paradigm is called MEC [2]. The success of the Internet of Things (IoT) and rich cloud services have helped create the need for edge computing, in which data processing occurs in part at the network edge, rather than completely in the cloud [3]. Edge computing could address concerns such as latency, mobile devices' limited battery life, bandwidth costs, security, and privacy [4].

To address the problem of a long latency, the cloud services should be moved to proximity of the User Equipment (UE). The MEC can be understood as a special case of the Mobile Cloud Computing (MCC). Nevertheless, in the conventional MCC, the cloud services are accessed via the Internet connection [5] while in the case of the edge computing, the computing/storage resources are supposed to be in proximity of the UEs (in sense of network topology). Hence, the MEC can offer significantly lower latencies and jitter when compared to the MCC.

DOI: 10.24507/ijicic.15.02.713

Even so, MEC paradigm still needs to have a higher fault tolerant ability to accomplish different tasks of distributed computation. The protocol of reaching consensus is a method of improving the fault tolerant capability of distributed system. Up to now, there have none related studies involving consensus issue in the MEC paradigm. It is the first time a protocol is proposed to reach consensus underlying MEC paradigm.

The *consensus* problem is defined by Meyer and Pradhan [6]. The solutions of *consensus* problem are defined as protocols, which achieve a consensus and hope to use the minimum number of rounds of message exchanges to achieve the maximum number of allowable faulty capability. In this study, the solution of consensus problem is concerned in the MEC paradigm. The definition of the problem is to make the fault-free nodes (MEC nodes) in the MEC paradigm to reach consensus. Each MEC node of MEC paradigm chooses an initial value to start with, and communicates to each other by exchanging messages. The MEC nodes are referred to make a consensus if it satisfies the following conditions [6].

Consensus: All fault-free MEC nodes agree on a common value.

Validity: If the initial value of each fault-free MEC node n_i is v_i then all fault-free MEC nodes shall agree on the value v_i.

In a consensus problem, many cases are based on the assumption of node failure in a fail-safe network [7]. Based on this assumption, a Transmission Medium (TM) fault is treated as a node fault, whatever the correctness of an innocent node, so an innocent node does not involve consensus. However, the definition of a consensus problem requires all fault-free nodes to reach a consensus. Therefore, the consensus problem is to be solved on the MEC paradigm within fallible TMs in this study. The proposed protocol is named *Reliable Consensus of MEC* paradigm (RCMEC). The RCMEC can solve the consensus problem in an MEC paradigm by using the minimum number of rounds of message exchanges (2 rounds, in fact) and increase the fault tolerance capability by allowing the maximum number of malicious faulty TMs.

The remainder of this paper is arranged as follows. Section 2 illustrates the topology of MEC paradigm. Section 3 illustrates the concept of the RCMEC. For easy understanding, an example of RCMEC executed is given in Section 4. The complexity of the proposed protocol is explained in Section 5. Finally, conclusions and future works are presented in Section 6.

2. The Topology of MEC Paradigm.

The term MEC was first used in 2013 to describe the implementation of services at the network edge [8]. In recent years, user demand for data rates and Quality of Service (QoS) has grown exponentially. In addition, the development of mobile user devices such as smartphones or laptops and the development of new mobile applications are rapidly advancing [9].

A novel architecture of MEC is proposed by Arif and Ejaz [10], as shown in Figure 1. There are three basic components in the architecture. 1) Edge devices include all types of mobile devices (UE) connected to the Internet. 2) Edge cloud is the less resourceful cloud deployed in each of the mobile base station. Edge cloud is responsible for traditional network traffic control, including forwarding and filtering, as well as hosting a variety of mobile edge applications such as edge health care, smart tracking, and more. 3) Public cloud is the cloud infrastructure hosted in the Internet.

The prime objectives of MEC are [10]:

1) Optimization of mobile resources by hosting compute intensive application at the edge network;
2) Optimization of the large data before sending to the cloud;
3) Enabling cloud services within the close proximity of mobile subscribers;

4) Providing context-aware services with the help of radio access network information.

In order to provide a high flexible and reliable platform of IoT, a topology of MEC paradigm is redefined in this study. The topology of MEC is shown in Figure 2. There are three layers in the MEC paradigm: *Sensing-layer*, *MEC-layer* and *CC-layer*. The Sensing-layer is composed of sensor nodes (UEs), which is responsible for sensing the data required by the application. The MEC-layer is constructed by a set of MEC groups; each MEC group is composed of a large number of MEC servers (MEC nodes), responsible for

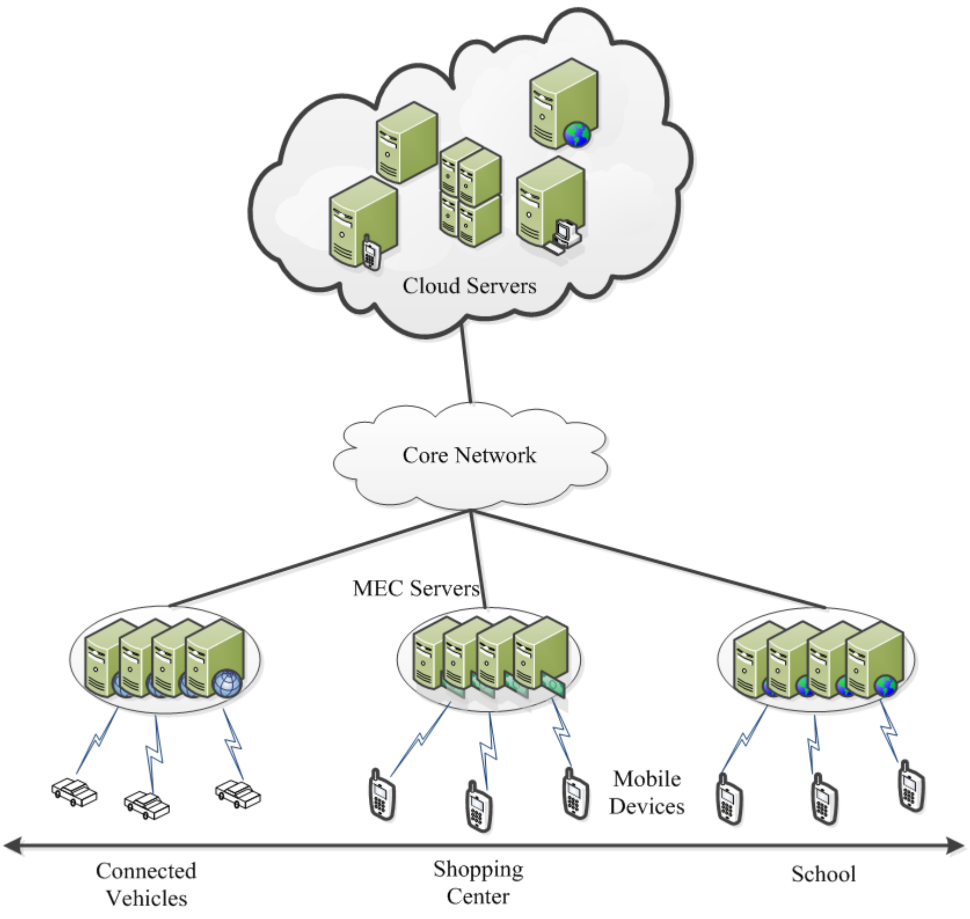


FIGURE 1. The architecture of MEC proposed by Arif and Ejaz [10]

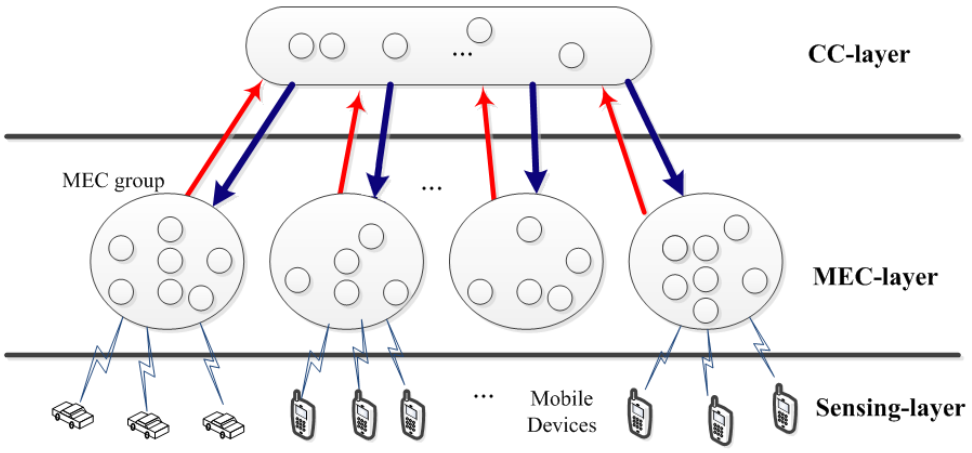


FIGURE 2. The topology of MEC paradigm

the processing of specific information and judgments. The CC-layer is made up of many cloud nodes, which provide cloud users' services. In the MEC paradigm, through the combination of a large number of UEs, various types of requesting data in real life can be collected. These huge requesting data from all over are used, and then a wide range of application services can be provided.

In MEC paradigm, data can be analyzed and processed by the MEC-layer instead of being centralized in the cloud computing. By coordinating and managing the computing and storage resources at the edge of the network, more and more connected devices and the emerging needs can be processed by the MEC. When the technological requirements and constraints of the applications are properly fulfilled, it is up to the platform designer to decide whether an endpoint should be served by the cloud computing, the MEC, or an adequate combination of the two at any given time during the service lifetime. Based on the above characteristics, the MEC can serve as a suitable platform for providing key services and applications, including connecting vehicles, smart cities, and shopping centers.

In an MEC paradigm, the nodes are interconnected. Reaching consensus on a same value in a distributed system, even if certain TMs in distributed system were failed (inner damage or outer intruder), the protocols are required so that systems still can be executed fault-freely. In this study, a distributed system whose nodes are reliable during the consensus is executed in MEC paradigm; while the TMs may be fault by interference from some noise or a hijacker and results in the exchanged messages can exhibit arbitrary behavior. When all nodes reach consensus in MEC paradigm, the fault-tolerance capacity is enhanced due to the fact that each node can transmit its messages directly even if TM is fault.

Actually, the symptom of a faulty TM can be classified into two types: dormant (such as crash, stuck-at, or delay) and malicious. In the event of a malicious failure, the behavior of the faulty TM is unpredictable and arbitrary. The message transmitted by the malicious faulty TM is random or arbitrary. This is the most destructive type of failure and leads to the most serious problems. That is, if the consensus problem can be resolved in the case of a malicious fault, then the consensus problem can also be resolved in other failure modes. Therefore, the consensus problem is revisited with the assumption of TM failure on malicious faults in the MEC paradigm in this study.

3. The Proposed Protocol.

In this study, the consensus problem is discussed in an MEC paradigm, and there is no delay of nodes or TMs included in our discussion. Therefore, the nodes executing our new protocol should receive the messages from other nodes within a predictable period of time [11]. If the message is not received on time, the message must have been influenced by faulty components.

In this research, RCMEC is used to solve the consensus problem in an MEC paradigm with malicious fallible TMs. With consideration for efficient consensus, the UEs of Sensing-layer are used to sense the required data of a specific application. And then, a procedure of making the MEC nodes reaching a common value is applied, and the cloud node in CC-layer is used to serve the cloud services. The variables used in this study are set as follows.

- S_j is a sensing area of Sensing-layer.
- E_j is an MEC group of MEC-layer.
- e_{ij} is a node in the MEC group E_j of MEC-layer, $1 \leq i \leq n_{Ej}$ where n_{Ej} is the number of MEC nodes in MEC group E_j of MEC-layer.
- TM_{SEj} is the total number of TMs between sensing area S_j of Sensing-layer and the MEC group E_j of MEC-layer.

- f_{SEj} is the total number of allowable malicious faulty TMs between sensing area S_j of Sensing-layer and the MEC group E_j of MEC-layer.
- TM_{Ej} is the number of TMs in MEC group E_j of MEC-layer.
- f_{Ej} is the total number of allowable malicious faulty TMs in MEC group E_j of MEC-layer.
- TM_{ECj} is the number of TMs between group MEC E_j of MEC-layer and the cloud nodes in CC-layer.
- f_{ECj} is the total number of allowable malicious faulty TMs between group MEC E_j of MEC-layer and the cloud nodes in CC-layer.

According to the previous researches [6,7,12-14], the number of allowable faulty TMs in consensus problem is determined by the total number of TMs in a distributed system. In Wang et al.'s protocol [14], the constraint is $\lceil (t-1)/2 \rceil - 1$ faulty TMs where t is the total number of TMs in a distributed system. Therefore, the constraints of the RCMEC as shown in the following.

- **(Constraint between Sensing-layer and MEC-layer)**: $TM_{SEj} > \lfloor (TM_{SEj} - 1)/2 \rfloor + f_{SEj}$ where TM_{SEj} is the total number of TMs and f_{SEj} is the total number of allowable malicious faulty TMs between sensing area S_j of Sensing-layer and the MEC group E_j of MEC-layer. This constraint specifies the number of TMs required between Sensing-layer and MEC-layer.
- **(Constraint of MEC-layer)**: $TM_{Ej} > \lfloor (TM_{Ej} - 1)/2 \rfloor + f_{Ej}$ where TM_{Ej} is the number of TMs and f_{Ej} is the total number of allowable malicious faulty TMs in MEC group E_j of MEC-layer. This constraint specifies the number of TMs required in MEC group E_j of MEC-layer.
- **(Constraint between MEC-layer and CC-layer)**: $TM_{ECj} > \lfloor (TM_{ECj} - 1)/2 \rfloor + f_{ECj}$ where TM_{ECj} is the number of TMs and f_{ECj} is the total number of allowable malicious faulty TMs between group MEC E_j of MEC-layer and the cloud nodes in CC-layer. This constraint specifies the number of TMs required between MEC-layer and CC-layer.

In this study, RCMEC is proposed to solve the consensus problem with fallible TMs underlying an MEC paradigm. The proposed protocol RCMEC is divided into three parts based on the three layers of MEC paradigm. The progression steps of RCMEC are shown in Figure 3. And, the scenario of RCMEC is shown in Figure 4.

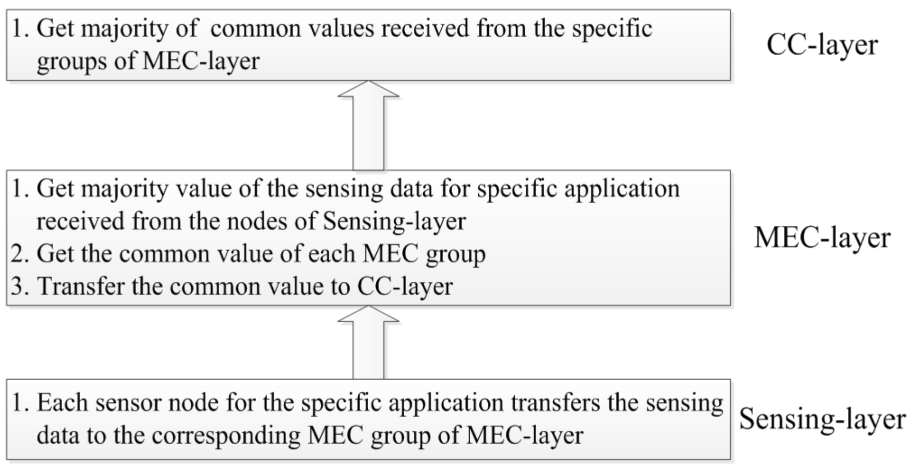

1. Get majority of common values received from the specific groups of MEC-layer
CC-layer
1. Get majority value of the sensing data for specific application received from the nodes of Sensing-layer
2. Get the common value of each MEC group
3. Transfer the common value to CC-layer
MEC-layer
1. Each sensor node for the specific application transfers the sensing data to the corresponding MEC group of MEC-layer
Sensing-layer

FIGURE 3. The progression steps of RCMEC

```
RCMEC
1. The requests for the application services are sent to the corresponding MEC group
   of MEC-layer by UEs.
2. The MEC nodes of MEC-layer execute the procedure Pcons.
3. The cloud nodes of CC-layer get the values $DEC_i$ received from the MEC nodes of
   MEC-layer, and take the majority of all the values $DEC_i$'s received to reach conse-
   nsus.

Procedure Pcons (for the node $e_{ij}$ in the MEC group $E_j$ of MEC-layer, $1 \leq i \leq n_{Ej}$
where $n_{Ej}$ is the number of MEC nodes in MEC group $E_j$ of MEC-layer)
Step 1. The node $e_{ij}$ receives the requests sent from UE.
Step 2. The received requests are taken as the majority.
Step 3. The majority value is used as the initial value ($v_i$) of $e_{ij}$.
Step 4. Procedure Mesgat is executed, and then the common value $DEC_i$ is obtained.
Step 5. The common value $DEC_i$ is transferred to CC-layer.

Procedure Mesgat (node $e_{ij}$ with initial value $v_i$)
  Message Exchange Phase:
  Round 1: Node $e_{ij}$ broadcasts $v_i$, and then receives the initial value from the other
           nodes in the same group, and construct vector $V_i = [v_1, v_2, \ldots, v_n]$, $1 \leq i$
           $\leq n_{Ej}$.
  Round 2: Node $e_{ij}$ broadcasts $V_i$, receives $V_k$ from node $e_{kj}$, $1 \leq k \leq n_{Ej}$. Construct
           a matrix $MAT_i$ (Setting the vector $V_k$ in the $k$-th column, for $1 \leq k \leq n_{Ej}$)
  Decision Making Phase:
  Step 1: Make $MAJ_k$ be the majority value of the $k$-th row of $MAT_i$ for $1 \leq k \leq n_{Ej}$.
  Step 2: Search for any $MAJ_k$. If ($\exists MAJ_k = \neg v_i$), then $DEC_i = \phi$;
  Step 3:   else if ($\exists MAJ_k = ?$) AND ($v_{ki} = v_i$), then $DEC_i = \phi$;
              else $DEC_i = v_i$.
```

Figure 4. The proposed protocol RCMEC

In this study, RCMEC is proposed to solve the consensus problem even if the faulty TMs change the transmitted messages to influence the system to achieve consensus in an MEC paradigm. There are two procedures embedded in the RCMEC: *Pcons* and *Mesgat*. Procedure *Pcons* is the core of RCMEC executed by the MEC nodes of MEC-layer to solve the consensus problem.

There are two phases of procedure *Mesgat*: the *message exchange phase* and *decision making phase*. In our case, the procedure *Mesgat* only needs two rounds of message exchanges to solve the consensus problem. In the first round of the *message exchange phase*, each MEC node e_{ij} multicasts its initial value v_i through TMs and then receives the initial values of other nodes as well. In the second round, each MEC node e_{ij} acts as the sender, sends the vector V_i received in the first round, and constructs a matrix $[V_1, V_2, \ldots, V_i]$, denoted by MAT_i, $1 \leq i \leq n_{Ej}$. And then, in the *decision making phase*, the function MAJ_k is used to determine the common value DEC_i by taking the majority value of the k-th row of MAT_i for $1 \leq k \leq n_{Ej}$. Finally, the common value DEC_i will be used to reach consensus for all cloud nodes of CC-layer.

4. **An Example of Executing RCMEC.** Taking the traffic control system constructed by MEC paradigm as an example to execute RCMEC is presented in Figures 5-10. The example of MEC paradigm is shown in Figure 5. Firstly, each UE senses the traffic status. The TM between UE_{12} and MEC-layer, and the TM between UE_{14} and MEC-layer are

assumed in malicious fault, and the sensing data of each UE is shown in Figure 6. Then, the sensing traffic statuses of the specific road intersection are transferred to MEC group E_1 of MEC-layer. Because the TM between UE_{12} and MEC-layer, and the TM between UE_{14} and MEC-layer are malicious failures, the message transmitted by the MEC node

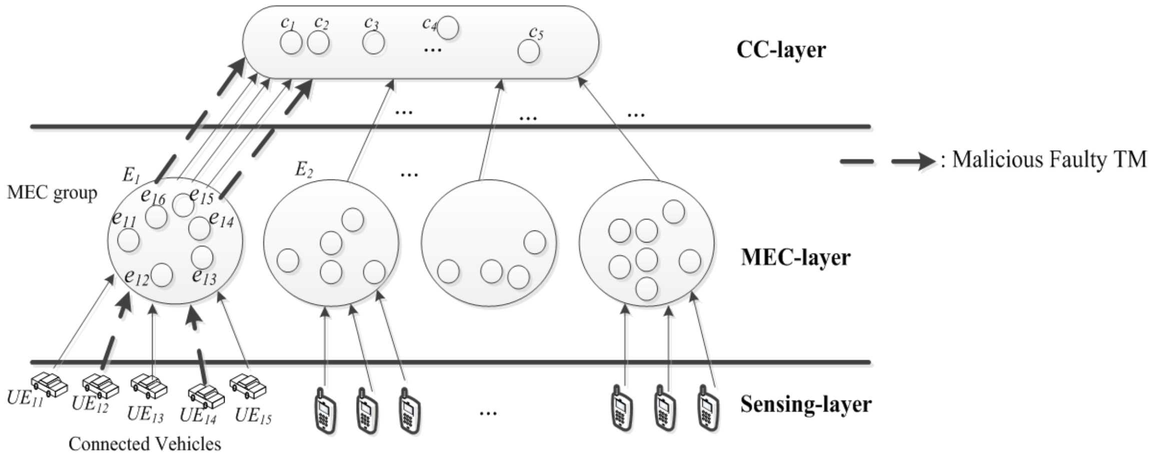
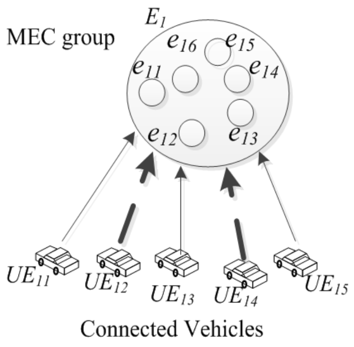


FIGURE 5. An example of MEC paradigm



UE_{11}	UE_{12}	UE_{13}	UE_{14}	UE_{15}
1	1	1	1	1

FIGURE 6. The sensing data of each connected vehicle node in the Sensing-layer

Step 1: The node e_{1j} receives the requests sent from mobile devices.
Step 2: The received requests are taken as the majority.
Step 3: The majority value is used as the initial value (v_i) of e_{1j}.

	UE_{11}	UE_{12}	UE_{13}	UE_{14}	UE_{15}	*Majority*
e_{11}	1	***1***	1	***0***	1	1
e_{12}	1	***0***	1	***0***	1	1
e_{13}	1	***0***	1	***1***	1	1
e_{14}	1	***0***	1	***0***	1	1
e_{15}	1	***1***	1	***0***	1	1
e_{16}	1	***0***	1	***0***	1	1

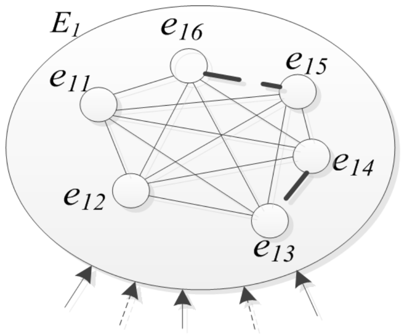
The received requests sent from UEs and the majority

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}
1	1	1	1	1	1

The initial value of each MEC node

FIGURE 7. The initial value of each MEC node in MEC group E_1 of MEC-layer

Step 4: Procedure *Mesgat* is executed.



	e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}
e_{11}	1	1	1	1	1	1
e_{12}	1	1	1	1	1	1
e_{13}	1	1	1	***0***	1	1
e_{14}	1	1	***0***	1	1	1
e_{15}	1	1	1	1	1	***0***
e_{16}	1	1	1	1	***0***	1

FIGURE 8. The vector received in the first round of MEC group E_1 of MEC-layer

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
1	1	1	0	1	1	
1	1	0	1	1	1	$DEC_{11} = 1$
1	1	1	1	1	0	
1	1	1	1	0	1	
1	1	1	1	1	1	
MAJ_{11} of MAT_{11}						

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
1	1	1	0	1	1	
1	1	0	1	1	1	$DEC_{12} = 1$
1	1	1	1	1	0	
1	1	1	1	0	1	
1	1	1	1	1	1	
MAJ_{12} of MAT_{12}						

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
1	1	1	0	1	1	
0	***0***	***1***	***0***	***0***	***0***	$DEC_{13} = 1$
1	1	1	1	1	0	
1	1	1	1	0	1	
1	1	1	1	1	1	
MAJ_{13} of MAT_{13}						

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
0	***0***	***1***	***0***	***0***	***0***	
1	1	0	1	1	1	$DEC_{14} = 1$
1	1	1	1	1	0	
1	1	1	1	0	1	
1	1	1	1	1	1	
MAJ_{14} of MAT_{14}						

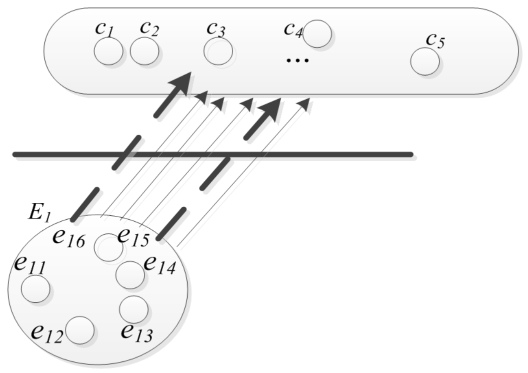
e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
1	1	1	0	1	1	
1	1	0	1	1	1	$DEC_{15} = 1$
1	1	1	1	1	0	
0	***0***	***1***	***0***	***0***	***0***	
1	1	1	1	1	1	
MAJ_{15} of MAT_{15}						

e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	
1	1	1	1	1	1	
1	1	1	1	1	1	
1	1	1	0	1	1	
1	1	0	1	1	1	$DEC_{16} = 1$
0	***0***	***1***	***0***	***0***	***0***	
1	1	1	1	0	1	
1	1	1	1	1	1	
MAJ_{16} of MAT_{16}						

FIGURE 9. Constructing MAT_1 in the second round and MAJ_1 of MAT_1 as common value

through the malicious damage TMs will be maliciously changed. In Figures 5-10, the messages are represented by bold and italics that indicate the messages had been maliciously modified.

Step 5: The common value is transferred to CC-layer.
The cloud nodes of CC-layer get the consensus values



	e_{11}	e_{12}	e_{13}	e_{14}	e_{15}	e_{16}	*Majority*
c_1	***0***	1	1	1	***0***	1	1
c_2	***0***	1	1	1	***0***	1	1
c_3	***1***	1	1	1	***0***	1	1
c_4	***0***	1	1	1	***0***	1	1
c_5	***1***	1	1	1	***0***	1	1

FIGURE 10. The consensus value of each node in CC-layer

The nodes of MEC-layer execute procedure *Pcons*. The MEC node receives the requests sent from UEs, and the received requests are taken as the majority. The majority value is used as the initial value (v_i) of MEC node. The initial value of each node in MEC group E_1 of MEC-layer is shown in Figure 7. Then the procedure *Mesgat* is executed. In the *message exchange phase* of the procedure *Mesgat*, each MEC node communicates with other nodes and itself. Then, the *decision making phase* will reach consensus among the nodes.

In the first round of the *message exchange phase*, each MEC node e_{ij} broadcasts v_i, and then receives the initial value from the other MEC nodes in the same group, and construct vector V_i. In this example, the TM between e_{13} and e_{14}, the TM between e_{15} and e_{16} are assumed in malicious fault. Then, the vector received in the first round of MEC group E_1 of MEC-layer is shown in Figure 8. In the second round of *message exchange phase* in procedure *Mesgat*, MEC node e_{ij} broadcasts V_i, and then receives the vectors broadcast by other MEC nodes, and construct MAT_i. After that the *decision making phase* takes the majority value of MAT_1 to construct the matrix MAJ_1, and the common value DEC_1 ($= 1$) can be obtained for group E_1's MEC nodes. The MAT_1 is constructed in the second round and MAJ_1 of MAT_1 as decision value and is shown in Figure 9.

Finally, the common value of each MEC node in MEC group E_1 is transferred to CC-layer. In this example, the TM between e_{11} and CC-layer, the TM between e_{15} and CC-layer are assumed in malicious fault. The cloud nodes in CC-layer receive the common value of each MEC node in MEC group E_1, and the majority is taken on the received common values. The majority value is the consensus value of MEC group E_1. The consensus value of each cloud node in CC-layer is shown in Figure 10.

5. **The Complexity of the RCMEC Protocol.**

The following theorems are used to prove the complexity of RCMEC. The complexity of RCMEC is evaluated in terms of 1) the minimal number of rounds of message exchanges, and 2) the maximum number of allowable faulty components. Theorems 5.1 and 5.2 will show that the optimal solution is reached.

Theorem 5.1. *One round of message exchange cannot solve the consensus problem.*

Proof: Message exchange is necessary. A node cannot derive whether or not a disagreeable value exists in other nodes without message exchanging. Therefore, the consensus problem cannot be implemented. In addition, one round of message exchange is not enough to solve the consensus problem. If node n_i is connected with node n_m by faulty TM, node n_i may not know the initial value in node n_m by using only one round of message exchanges. Hence, it is possible to reach a consensus by using one round of message exchanges.

Theorem 5.2. *The total number of allowable faulty TMs by RCMEC is optimal.*

Proof: The total number of allowable faulty TMs by RCMEC can be discussed by three parts of MEC paradigm.

(1) **TMs between Sensing-layer and MEC-layer:** Since the number of faulty TMs between sensing area S_j of Sensing-layer and MEC-group E_j of MEC-layer does not exceed half, and the majority value of the sensing data can be determined. F_{SE} is the total number of allowable faulty TMs between sensing area and MEC-group. $F_{SE} = \sum_{j=1}^{S} f_{SEj}$ where S is the total number of sensing areas in Sensing-layer and f_{SEj} is the total number of allowable malicious faulty TMs between sensing area S_j and MEC-group E_j. In addition, $f_{SEj} \leq \lfloor (TM_{SEj} - 1)/2 \rfloor$ where TM_{SEj} is the number of TMs between sensing area S_j and MEC-group E_j.

(2) **TMs in MEC-layer:** Since the number of faulty TMs in each MEC group E_j of MEC-layer does not exceed half, and the majority value of the MEC group can be determined, F_E is the total number of allowable faulty TMs in MEC-layer. $F_E = \sum_{j=1}^{E} f_{Ej}$ where E is the total number of MEC groups of MEC-layer and f_{Ej} is the total number of allowable malicious faulty TMs in MEC group E_j. In addition, $f_{Ej} \leq \lfloor (TM_{Ej} - 1)/2 \rfloor$ where TM_{Ej} is the number of TMs in MEC group E_j.

(3) **TMs between MEC-layer and CC-layer:** Since the number of faulty TMs between MEC group E_j of MEC-layer and CC-layer does not exceed half, and the majority value of the consensus value can be determined, F_{EC} is the total number of allowable faulty TMs between MEC-layer and CC-layer. $F_{EC} = \sum_{j=1}^{E} f_{ECj}$ where E is the total number of MEC group in MEC-layer and f_{ECj} is the total number of allowable malicious faulty TMs between MEC group E_j and CC-layer. In addition, $f_{ECj} \leq \lfloor (TM_{ECj} - 1)/2 \rfloor$ where TM_{ECj} is the number of TMs between MEC group E_j of MEC-layer and CC-layer.

In short, the maximum number of allowable faulty components by RCMEC is $F = F_{SE} + F_E + F_{EC} = \sum_{j=1}^{S} f_{SEj} + \sum_{j=1}^{E} f_{Ej} + \sum_{j=1}^{E} f_{ECj}$. And, F is the maximum number of allowable faulty TMs in an MEC paradigm.

6. Conclusions.

In this study, the reliable consensus problem was redefined by the RCMEC protocol in an MEC paradigm. The proposed protocol ensures that all nodes in the network can reach a consensus value to cope with the influences of the faulty TMs by using the minimum number of message exchanges, while tolerating the maximum number of faulty TMs at any time. Our protocol is the first time to visit the consensus problem under the MEC paradigm.

Furthermore, only considering TM faults in the consensus problem is insufficient for the highly reliable MEC. In the real world, not only might TMs be fault, node might be fault. Therefore, our protocol will be extended to reach consensus in a generalized case when faulty TMs or nodes exist simultaneously in the future.

Acknowledgment. This work was supported in part by the Ministry of Science and Technology MOST 107-2221-E-324-005-MY3.

REFERENCES

[1] M. Chiang and T. Zhang, Fog and IoT: An overview of research opportunities, *IEEE Internet of Things Journal*, vol.3, no.6, pp.854-864, 2016.

[2] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta and A. Neal, *Mobile-Edge Computing Introductory Technical White Paper*, Mobile-Edge Computing (MEC) Industry Initiative, 2014.

[3] A. Whitmore, A. Agarwal and L. Xu, The Internet of Things – A survey of topics and trends, *Information Systems Frontiers*, vol.17, no.2, pp.261-274, 2015.

[4] Tadapaneni, N. R. (2016). Overview and Opportunities of Edge Computing. Social Science Research Network.

[5] A. R. Khan, M. Othman, S. A. Madani and S. U. Khan, A survey of mobile cloud computing application models, *IEEE Communications Surveys & Tutorials*, vol.16, no.1, pp.393-413, 2014.

[6] F. J. Meyer and D. K. Pradhan, Consensus with dual failure modes, *IEEE Trans. Parallel and Distributed Systems*, vol.2, no.2, pp.214-222, 1991.

[7] M. Fischer, The consensus problem in unreliable distributed systems (a brief survey), *International Conference on Fundamentals of Computation Theory*, vol.158, pp.127-140, 2002.

[8] O. Khalid, M. U. S. Khan, S. U. Khan and A. Y. Zomaya, Omnisuggest: A ubiquitous cloud-based context-aware recommendation system for mobile social networks, *IEEE Trans. Services Computing*, vol.7, no.3, pp.401-414, 2014.

[9] J. Jin, J. Gubbi, S. Marusic and M. Palaniswami, An information framework for creating a smart city through Internet of things, *IEEE Internet of Things Journal*, vol.1, no.2, pp.112-121, 2014.

[10] A. Arif and A. Ejaz, A survey on mobile edge computing, *Proc. of the 10th IEEE International Conference on Intelligent Systems and Control*, Coimbatore, Tamilnadu, India, 2016.

[11] X. Li, X. Chen and Y. Xie, Agreement of networks of discrete-time agents with mixed dynamics and time delays, *Mathematical Problems in Engineering*, http://dx.doi.org/10.1155/2015/957028, 2015.

[12] L. Lamport, R. Shostak and M. Pease, The Byzantine generals problem, *ACM Trans. Programming Languages and Systems*, vol.4, no.3, pp.382-401, 1982.

[13] M. J. Fischer and N. A. Lynch, A lower bound for the time to assure interactive consistency, *Infor- mation Processing Letters*, vol.14, no.4, pp.183-186, 1982.

[14] S.-C. Wang, S.-C. Tseng and K.-Q. Yan, To achieve optimal trustworthy agreement in the unreli- able mobile cloud computing environment, *ICIC Express Letters, Part B: Applications*, vol.8, no.6, pp.937-943, 2017.

[15] W. Shi and S. Dustdar, The promise of edge computing, Computer, vol.49, no.5, pp.78-81, 2016.

[16] Tina, F. (2020). A Comparison of Execution Mechanisms: Fog and Edge Cloud Computing.