# Draft - A Collaborative Mobile Edge Computing platform with Permissioned Blockchain

Roshan Singh
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati, India
Email: roshancs@iitg.ac.in

Sukumar Nandi
Department of Computer Science and Engineering
Indian Institute of Technology Guwahati, India
Email: sukumar@iitg.ac.in

***Abstract*—Edge computing is a computing paradigm that offers low latency services by bringing computation and data storage closer to data sources. Due to the resource constrained nature of edge devices they are not able to handle all computations independently. In such cases task offloading to the peer edge devices is looked for. In the case of Mobile Edge Computing(MEC) the edge devices are deployed by different vendors with no way to operate in collaboration with devices deployed by other vendors. Thus minimising the overall resource utilisation. Providing a common platform for these devices from different stakeholders can increase resource utilisation of the ecosystem. In this work we introduce a permissioned blockchain based platform for collaborative computing among the edge devices. We automate the computation offloading tasks via smart contracts**

**Keywords:** Mobile Edge Computing, Permissioned Blockchain

## I. Introduction

Mobile Edge Computing (MEC) enables computing services to be extended to the edge of the network through base stations and access points [1]. This enables computation intensive and latency critical applications to be executed at the network edge. Unlike cloud computing paradigm which has no assurance on latency bounds MEC provides low latency services and seamless integration of multiple applications from different service providers and vendors toward mobile subscribers. User applications can offload their computations on to the edge devices, processing of user jobs at the edge reduces service latency and backhaul traffic [2]. However, edge devices do suffer from constrained computing capabilities in comparison to the cloud and cannot perform all the computations on their own when they operate individually. Under such scenarios user task can be offloaded to peer edge devices in the network leveraging the available computational resources in the network. However, in a MEC environment edge devices might be deployed by various vendors having varying computational as well as storage capabilities. As such it is not in the personal interest of a vendor to process computation requests coming from edge devices operated by other vendors. Incentivising edge devices for processing computation requests can motivate them to participate in task offloading activities. In the mean time validity of the computation performed on the offloaded task need to be assesed. It might be the case that an edge device takes an offloaded task from another device and returns a false result. As the offloading device may not have its resources available at the moment to access the results the submitted false result might get unnoticed at the moment and can harm the performance of the offloading devices. Another strategic challenge while assigning a task to an end device is to select the most appropriate parameter for task offloading such as a device having minimum latency might not be always the best to process the task as in the MEC setup a misbehaving device might be much closer than other devices. Such situations in turn asks for proper and an effective reputation mechanism so that misbehaving edge should not get an offloaded task. However, as the participants in the MEC network may belong from different vendors which might not have an established trust among them in such situations a centralised approach for reputation as well as incentives might not work and might not be acceptable. In this work we propose a permissioned blockchain based platform for collaborative mobile edge computing.

## II. Steps in our proposed approach

### A. Setup:

Stakeholders interested to deploy a MEC network in a region can come up with edge devices and establish dedicated network among themselves. The proposed blockchain based architecture is shown in Fig. 1 . They reach a consensus to maintain a consortium blockchain among themselves. The Fig. 1 shows a Blockchain network formed by 3 different service providers.

- An edge server publishes a task offloading request with a deadline, minimum number of resources required, minimum reputation that a device should hold to contend for getting the offloaded task. The offloading request is made publically available to the edge servers in the MEC network as the request is published via the smart contracts. Edge devices subscribed to the event gets notified about the task.
- Each edge server in the MEC network opens up with its status such as available resources, bandwidth and reputation.

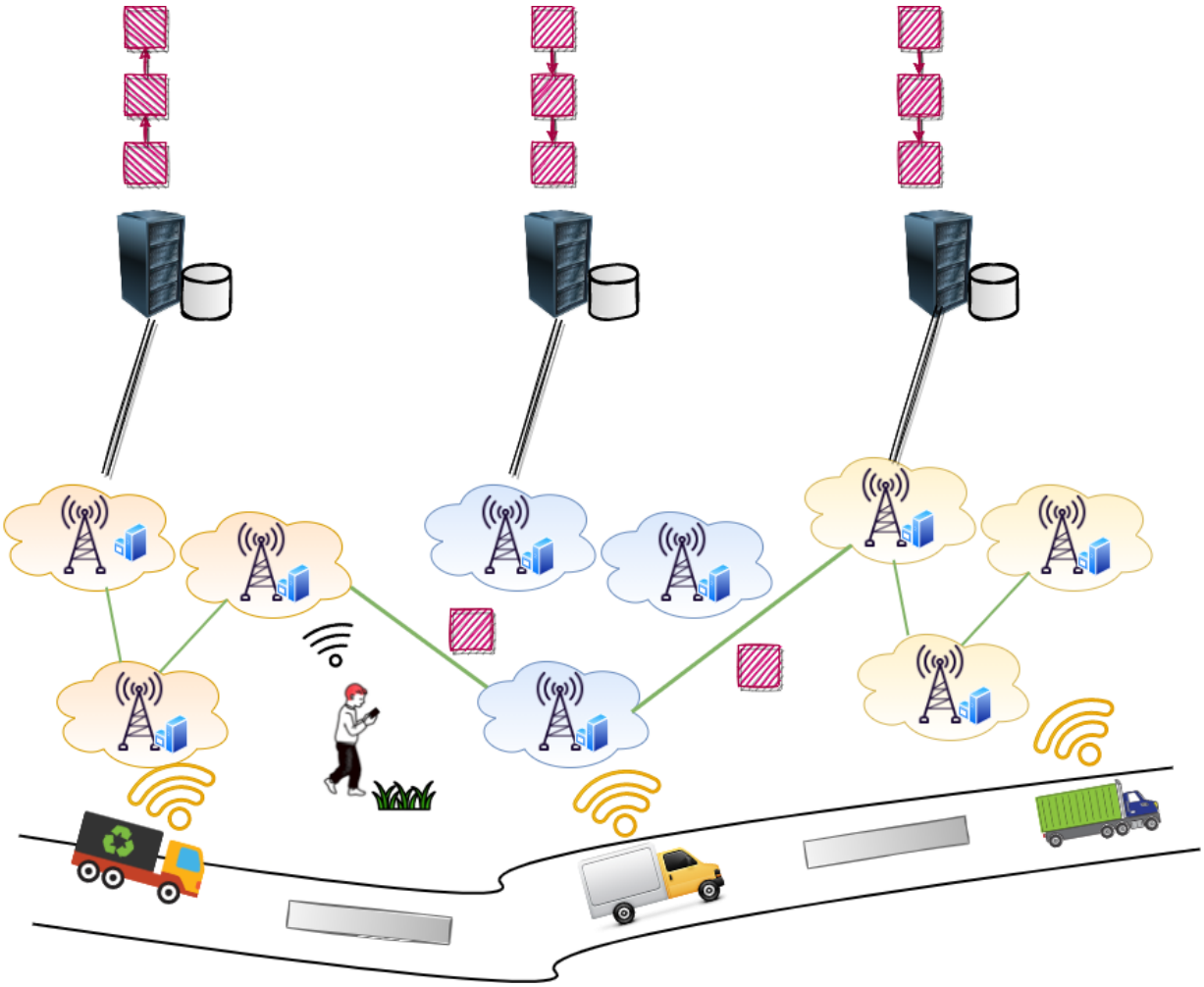

Fig. 1. Proposed Blockchain Based Mobile Edge Computing Framework

- The smart contract checks for a device having maximum reputation, sufficient resources and minimum latency from the offloading device.
- If none of the device is having the sufficient resources.
- Then task T is fractioned into $t_1, t_2 ... t_n$ and rewards are also fractioned into $r_1, r_2 ... r_n$ respectively.
- The best server is selected by the smart contract.
- The executor executes the task and logs the hash of the result to the smart contract.
- Later on the task is evaluated by the cloud service providers. And accordingly the reputation of the device is affected.

Fig. 2 shows the workflow of our approach and Algorithm 1 describes the device selection strategy.

| | **Meaning** |
|---|---|
| D | An Edge Device |
| T | A whole task |
| $t_i$ | ith portion of T |
| $d_T$ | Deadline of T |
| $\delta_T$ | Minimum Reputation for contending for T |
| $\text{Latency}_{D_i}^{D_j}$ | Latency from $D_i to D_j$ |
| d | Deadline of the task T |
| p | Processing time for T by $D_j$ |
| $\Delta$ | Reward for processing other Org. Task |

TABLE I
MEANINGS OF SYMBOL

### *B. Task Offloading*

The objective of task offloading is to assign the surplus task to other peer edge devices which an edge device is unable to perform by itself within the deadline. A good scheme for task allocation should take in account the cost of offloading the task to the peer device as well as assuring correctness of the result submitted by the executing peer. Works have taken into account the latency as one of the parameters while deciding for task offloading. However, latency on itself may not be the best parameter over correctness. In MEC setting where verification of the result might not be possible by the offloading edge device within the deadline due to the resource constraints. Also, as each offloading of task by an edge device is going

to incur it some costs. Tasks may again be prioritised based upon criticalities of the deadlines. A trade-off between the cost and strictness of the deadline has to be made. Similarly, situations may arise when peer edge devices might not be able to perform the offloaded task individually within the deadline. In such cases the task need to be fractioned and offloaded to the potential peers.

**Algorithm 1** Task Offloading

0: For a task $T$ from $D_i$ choose a set $S$ of edge devices where for each $D_j \in$ S $Latency_{D_i}^{D_j} \leq \delta$ and $D_j^{rep} \geq \alpha$
0: If S is empty repeat above step
0: If S is not empty select a $D_j \in$ S with minimum $Latency_{D_i}^{D_j}$ and maximum $D_j^{rep}$
0: If $D_j^r$ less $D_i^r$ remove $D_j$ from S and repeat
0: If no $D_j$ is found
0: Divide T into $t_1, t_2, t_3, ..., t_n$
0: **for** each $t_i \in t_1, t_2, t_3 ... t_n$ **do**
0: Assign $t_i$ to a $D_j \in$ S satisfying
0: Remove $D_j$ from S
0: **end for**=0

*1) Result Submission:* A device $D_j$ post completion of the processing of offloaded task T submits the result to the offloading device and logs the result submission onto the smart contract in the form of a hashed commitment.

### C. Reputation Management

Reputation Management is the core of our approach. Reputation of an edge server plays a crucial role in deciding whether an offloaded task will be allocated to it or not. Higher the reputation higher is the chances of a device to act as a task executor. A reputation management scheme must take into account the way the reputations are increased or decreased. Edge devices should be encouraged to maintain consistent good reputation score throughout its lifetime instead of maintaining an inconsistent score. A malicious edge device can earn a high reputation score by correctly executing the offloaded tasks over a period of time and may launch an attack on peer device by obtaining the offloaded task and executing it in incorrect manner. Our reputation scheme is developed on the idealogy that it takes long time to make a reputation and takes no time to loose it. The scheme uniformly increases the reputation but strongly penalises in case of misbehaviours. Also, correct execution of offloaded tasks from other organisations brings more credits than task executed by the same organisation.

The reputation updation for an edge device e for execution of task t is made post verification by the cloud devices across the organisations. The reputation updation for a device might not be instantaneous but it will happen eventually as the cloud devices verify the tasks individually and provide their feedbacks. The edge device shares the task with the cloud via the backhaul links. The following logic is used for reputation update for a device $D_j$ which has performed an offloaded task $T$ from the device $D_i$.

1) If $D_j$ is not from the same organisation of $D_i$ and result submitted for task $T$ is correct.
2) If $D_j$ is from the same organisation of $D_i$ and result submitted for task $T$ is correct.
3) If $D_j$ is not from the same organisation of $D_i$ and result submitted for task $T$ is incorrect.
4) If $D_j$ is from the same organisation of $D_i$ and result submitted for task $T$ is incorrect.

$$D_j^{rep} = \begin{cases} 1 & \text{(d/p) * } \Delta \\ 2 & \text{(d/p)} \\ 3 & \text{-2} * 1/(d/p) * \Delta \\ 4 & \text{-1}/(d/p) \end{cases} \quad (1)$$

The logic increases the reputation score of a device $D_j$ in proportion of the earliness of its task finish time. Consecutively correct result performed by a device for the task offloaded from another organisation will bring more reputation. At the same time incorrect result performed by a device for the task offloaded from another organisation will penalise the device $D_j$ much strictly. Consecutive good jobs of other organisations have higher value than the same organisation.

## III. Attack Model

The security of our approach lies on the fact that a malicious entity trying to defame a legitimate edge device by giving false confirmation for a task that it performed, need to control at least 51% of the cloud service providers. A successful attack on our approach can be made when at least 2/3 of the cloud service providers in the network collude against an entity, which is not an easy scenario to happen in large scale MEC network.

## IV. Experiments

We setup a private permissioned instance of Ethereum[3] blockchain with 3 nodes. The blockchain was configured to run the Clique consensus algorithm, a Proof of Authority[4] based algorithm. The choice of the PoA algorithm is made because of its lightweight characteristics and is less energy consuming than the challenge response based PoW algorithm. Out of the 3 nodes 2 were designated as sealers or the validator and one was made as a Full Node. The algorithm 1 was deployed onto the blockchain in the form of a smart contract[5]. We execute a workload of 10000 transactions to check the performance of the approach. The transactions were executed from a non validator node. The statistics can be found as below:-

| | |
|---|---|
| **Workload Execution Time** | 62.212 seconds |
| **Total Number of Blocks** | 63 |
| **Average Transaction per Block** | 159 (approx) |

The relationship between number of blocks and workload execution time is inversely proportional.

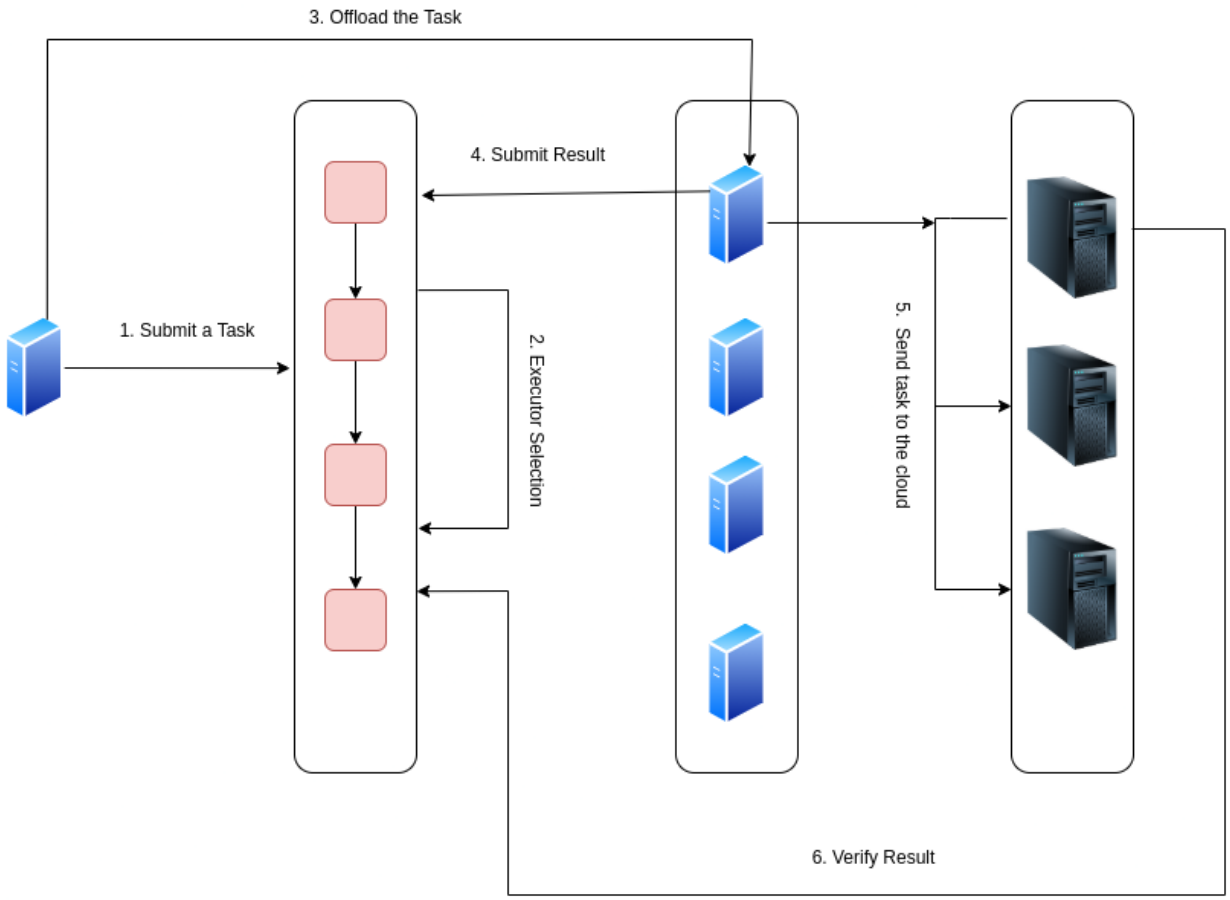


Fig. 2. Workflow of our approach

## V. Conclusion

In this work we proposed a novel permissioned blockchain based approach for task offloading in Mobile Edge Computing. Our approach allows edge devices to offload their task to other peer devices. A task can be offloaded as a whole or in fractions. The decision of task offloading is made with the help of smart contract which assures trustless execution. No entity can offload a task to its favourable party also no edge device can deny taking a task. We also Incorporated the idea of fractional task offloading that allows a task to be divided into smaller tasks and the smaller tasks can be offloaded to different peer edge devices capable of executing the portion of whole task. The approach will be able to provide a common platform to different vendors providing edge computing services which might not trust each other.

## VI. Related Works

The number of end-devices, such as wearable devices, mobile phones, tablets and Internet-of-Things (IoT) devices has grown explosively in the past decade. Many mobile and Internet of Things applications have become increasingly resource-intensive and latency-sensitive, such as web online gaming[6], AR[7], autonomous vehicles [8]. These applications render the conventional cloud computing paradigm obsolete for its high and often unpredictable network latency. Efforts have been devoted to enable edge servers to cooperate with each other. High-speed links are established between adjacent edge servers to allow them to communicate with each other and transmit data[9][10] over the edge server network[11] or via macro base stations based on the edge cloud architecture[12].

## VII. Acknowledgement

The work is supported by Information Security Education and Awareness (ISEA) Phase II Project at IIT Guwahati sponsored by MeitY, India.

## References

[1] 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, 2017.

[2] C.-F. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4132–4150, 2019.

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

[4] S. De Angelis, L. Aniello, R. Baldoni, F. Lombardi, A. Margheri, and V. Sassone, "Pbft vs proof-of-authority: Applying the cap theorem to permissioned blockchain," 2018.

[5] N. Szabo, "Formalizing and securing relationships on public networks," *First monday*, 1997.
[6] Y. Lin and H. Shen, "Cloudfog: Leveraging fog to extend cloud gaming for thin-client mmog with high quality of service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 2, pp. 431–445, 2016.
[7] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2018, pp. 468–476.
[8] A. Narayanan, E. Ramadan, J. Carpenter, Q. Liu, Y. Liu, F. Qian, and Z.-L. Zhang, "A first look at commercial 5g performance on smartphones," in *Proceedings of The Web Conference 2020*, 2020, pp. 894–905.
[9] L. Chen, S. Zhou, and J. Xu, "Computation peer offloading for energy-constrained mobile edge computing in small-cell networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1619–1632, 2018.
[10] H. Guo and J. Liu, "Collaborative computation offloading for multiaccess edge computing over fiber–wireless networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 4514–4526, 2018.
[11] X. Xia, F. Chen, Q. He, J. C. Grundy, M. Abdelrazek, and H. Jin, "Cost-effective app data distribution in edge computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 1, pp. 31–44, 2020.
[12] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. IEEE, 2016, pp. 1–9.