A Recommendation of a System Integrated With Edge Nodes, Multi-Cloud Instance and Decision-Making Mechanism for Voting System

[1]Kanev Boris Lisitsa
[1] Faculty of Science and Engineering, University of Liverpool, UK.
[1]borislisitsa@hotmail.com

ArticleInfo
Journal of Machine and Computing (http://anapub.co.ke/journals/jmc/jmc.html)
Doi: https://doi.org/10.53759/7669/jmc202202016
Received 25 January 2022; Revised form 20 March 2022; Accepted 12 May 2022.
Available online 05 July 2022.

Abstract – Edge Computing (EC) services and hardware help mitigate issues by being a local resource of storage and processing for various systems. Edge gateway, for instance, is capable of processing datasets from edge devices, and then sending only essential datasets back via the cloud systems, hence minimizing the bandwidth required. Or it can send data to the edge device in case of actual-time application needs. Individual users are served through a distributed computing system that includes data processing and storage as part of the edge computing concept. An important role is played by the framework in the IoT (Internet - Of - things) area (IoT). Fully distributed edge clouds raise additional issues, such as the potential for recovery and privacy breaches. Our solution is based on multi-cloud installations and the edge nodes, which are essential for a voting mechanism in this article. In order to verify calculations that occur in the centralised environment, it is advised to use the edge networking.

Keywords – Edge Computing (EC), Internet of Things (IoT), Byzantine Fault Tolerance (BFT), Cloud Environment.

I. INTRODUCTION

According to [1,] EC is an element of a distributed software infrastructure wherein processing of data is performed close to the edge, where items and people produce or ingest this data. With edge computing, computations and data management are moved closer to where they are being collected, rather than being delegated to some remote point. As a result, an application's performance will not suffer from delay concerns as a result of this. The quantity of data that has to be processed centrally or on the cloud may be reduced by having the computing done locally, which saves money for the company. IoT devices, which link to the network for either obtaining data from the cloud or presenting data back to the cloud, spurred the development of edge computing. It's not only smartphones that generate a big data while they are in use.

The approach of Edge Computing (EC) is vital for enhancing the cloud application by shifting large amounts of services or data from centralized cloud units to the units that are nearer to the end users. Whereas most common interlinked objects leverage IoT (Internet of Things), and cloud technology, application builders or makers are starting to realize the benefits of undertaking more analysis and processing of interlinked equipment. Edge processing [2] is used in essential systems to minimize latency and do real-time computations, as well as to better handle the huge amounts of data created by IoT devices and reduce reliance on network access to the cloud. The agricultural business, for example, needs real-time soil property surveillance to identify soil health, moisture, and agricultural production type. All agriculture organizations must be linked in order to create a decision-making system which can help enhance productivity and make farm produce transportation from producers to marketing companies and merchants to farmers easier.

Edge computing, on the other hand, cannot compete with the advantages of a central approach, which include superior overall information and program management due to their centralization. To overcome the many cloud restrictions, businesses are typically searching for ideal cloud ecosystem, which is both simpler and safer to configure: Data degradation, vendor lock-in, and loss of accessibility and privacy Using numerous clouds to duplicate services may seem to be a cost-effective approach, however it is not. Using several clouds is initially more costly than using just one cloud in every way. Compatibility and accessibility, however, are critical to the architecture's effectiveness. Companies can save long-term expenses, increase resiliency and safety, and prevent vendor lock-in by using several clouds.

Cloud computing [3] has the potential to increase the efficiency of smart-city services in **Fig. 2** by providing storage, analysis, and information extraction from raw data. Academics and professionals presented a plan to assist in the development of smart cities by using a secure heterogeneous cloud architecture in a current vision of technology for smart surroundings. Cloud-based proposed techniques can robustly safeguard sensitive dataset, generate cloud uploads for disasters

response, and save of the expenses generally, whereas edge computing can deal with time-crucial necessities of IoT application and the applications of finite resource of the edge node.

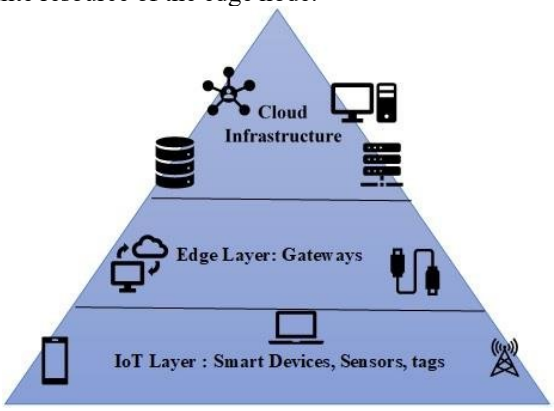

Fig 1. 3-layer edge network infrastructure

Fig. 1 above represents a 3-tier IoT system, with the current infrastructural systems serving as the bottom two tiers of the network (e.g., cloud and cellular networks). In this design, there are three layers: the "cloud," the "edge," and the "device" layers. Information from IoT devices is collected at the gadget tier, which is located at the platform's exit points. Edge routers may be used to link certain IoT devices. The 2nd tier is where edge computing skillset dwells. This is the layer that integrates web devices including routers, gateways, and switches, which may process, compute, and temporarily store incoming information. Organized hierarchically and positioned among the cloud services and the IoT devices, the edge layer consists of a set of edge devices. Network traffic may be routed via edge nodes, which often have a lot of processing power. Base stations, routers, and switches, as well as small-scale data centers, are all examples. Data centers' capabilities determine how much can be stored and processed in the cloud layer, which correlates to cloud cognition.

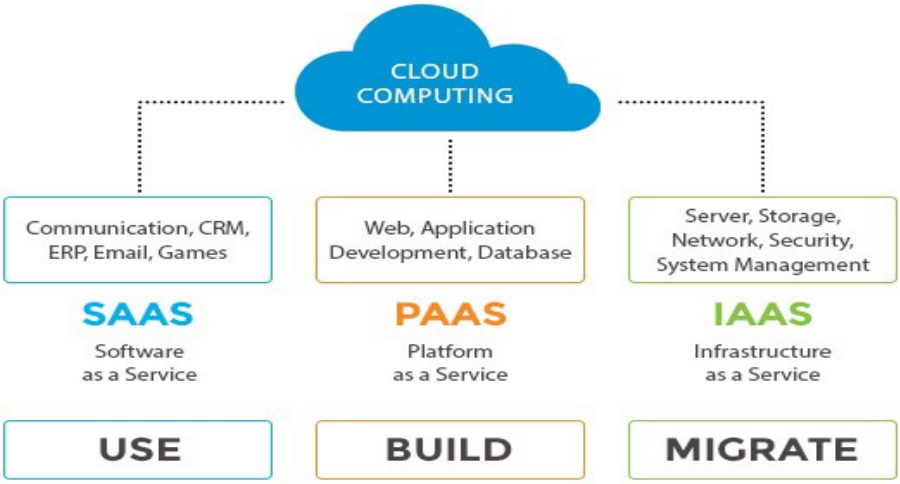

Fig 2. Cloud computing services

There is still the primary benefit of utilizing clouds as a supportive system, but control and trust choices are now placed on the edge devices rather than the network infrastructure. Additional layers are used in other systems that augment this design. These extra tiers are purposed to integrate business needs to standardized infrastructure. Using a generic three-tier design, this problem can be solved. For example, in a distributed network, where elements might fail at any time and supply incorrect data, Byzantine Fault Tolerance (BFT) is concerned with the reliability of computer networks.

Littlewood [4] originally raised the issue of Byzantine commanders, and it continues to be of interest today. Byzantine failures may arise in distributed system settings like centralised clouds or edge nodes. Failure modes in this category provide the greatest challenge since there are no limits or decisions taken about the algorithm parameters. Often, the data that passes via the edge network is left unmanaged and uncontrolled in today 's systems. A failure of the edge node indicates that there may be a lack of information about whether a certain node has ceased operations. Due to a lack of crucial security features, a dispersed ecosystem's decentralised administration might contribute to confidentiality violations. As a result, existing cloud-based authentication and authorization systems must be adapted for the edge environment. It's become vital to devise strategies for guaranteeing "correctness" in a dangerous setting. Edge computing presents a set of dangers that are distinct from those faced in a centralized cloud computing. Edge computing safety is far more difficult to maintain than cloud security

since it requires dispersed processing. Data or computational corruption may be detected using a series of Byzantine agreements protocols, which can help increase safety.

Cloud computing services (**see Fig. 2**) operating in data centers are likewise susceptible to byzantine failures. Random errors are known to occur in the information center, which might alter the 'accurateness of the findings' and contaminate the processing. It is also possible for harmful assaults done by cloud employees or exterior hackers to compromise data processing and its outcomes. Third, cloud disruptions may cause cloud services to be unavailable. Facebook and Cloudflare, to name a few platforms, have been known to have outages lasting for hours that may affect the whole world's traffic.

Here, we present an edge-nodes-based voting strategy for multi-cloud computational validation, which operates across a segment of various edge nodes all found in a similar geographic area. The uniqueness of this approach lies in its ability to parallelize computation while simultaneously transparently tolerating Byzantine errors by combining different clouds and edge nodes. This remedy can be vital for data-centric IoT devices and applications, which necessitate the following: For various computational assessments, IoT content relies on the infrastructures; for computational accuracy, it is critical; and for low latencies should be considered when developing IoT systems and sensors. The condition of multiple IoT devices and application is considered to possess the features since end users desire to accept meaningful skillset from the dataset. The edge ecosystem presents various non-trivial problems for our strategy in achieving this goal. In the first place, it attempts to be visible to any IoT device or user. Secondly, this is a generic architecture for relaying computations from edge nodes to the cloud. Third, this system makes use of a voting mechanism to verify edge network cloud computing.

This paper aims at recommending a system integrated with edge nodes and multi-cloud instance and a decision-making mechanism for voting. This objective has been approached at as follows: Section II presents a literature review of related works. Section III provided an analysis of the proposed system model and its architecture. Section IV provides an evaluation and discussion of the results. Lastly, Section V finalizes the research and proposes directions for future research.

II. LITERATURE REVIEW

According to [5], by 2025, global data is anticipated to increase by 61percent to 175 zettabytes. It is estimated that 10 percent of the enterprise's data is produced and handled outside of a typical information center or cloud. According to the business, this percentage will rise to 75 percent by 2025. To store and use all the information being generated by IoT devices in a cloud computing environment would demand an ever-increasing quantity of network bandwidth. Despite advances in networking technologies, data centers cannot ensure adequate transfer rates and reaction times, which, nonetheless, is a fundamental need for many applications. Moreover, gadgets at the edge are continually consuming data from the cloud, requiring enterprises to delegate storage of data and service delivery, utilizing the physical closeness of the end user.

According to Peng et al. [6], a related goal of edge computing is to migrate processing away from information centers and toward the network edge, using smart devices such as smartphones and networking gateways to conduct activities on purpose in the cloud. By pushing operations to the edge, it is feasible to offer material caching, delivery of services, permanent information storage, and IoT management, culminating in improved reaction times and transmission rates. At the very same time, spreading the logic over several network nodes raises additional concerns and difficulties.

Privacy and Security

Siriweera and Naruse [7] argue that as a result of the cloud's decentralized architecture, traditional security measures must be rethought. Data may transit between dispersed nodes linked through the Internet in edge computing, necessitating the need of encryption techniques separate from the cloud. As resource-constrained devices, edge nodes might potentially restrict the range of security options. Decentralized trust models are needed to replace the current centralized top-down infrastructure. Aside from the fact that the cloud is a great place to store and process sensitive information, it is also feasible to improve privacy by storing and processing data on-site. Furthermore, service providers no longer own the obtained data; instead, end users now do.

Scalability

In [8], distributed networks have unique challenges when it comes to scalability. In the first place, the devices' heterogeneity, with their varying performance and energy requirements, the highly dynamic environment in which they operate, and the dependability of their connections must be taken into consideration in comparison to the more resilient architecture of cloud service centers. This might slow down the expansion process even more because of additional delay introduced by security requirements.

Reliability

As per [9], proper handling of failovers is critical to the continued operation of a service. To provide uninterrupted service, a single node must fall down and become inaccessible. Edge computing platforms must also be able to recover from failures and warn the user. Error identification and recovery may be made much easier if the network structure of the whole distributed system is maintained by each device. There are a number of other aspects that might affect this feature, including the dependability of the connections and the quality of data provided at the edge, which may be affected by environmental circumstances. Local customers may be served by edge computing devices such as voice assistants even if the cloud or the internet is down.

Speed

Applications may be more reactive and more efficient thanks to edge computing, which puts processing resources closer to end users. With the right architecture, an edge platform would be able to exceed a cloud-based platform in most cases. Because certain applications need rapid response, edge computing is a better choice than cloud computing in many cases. Everything from the Internet of Things (IoT) to self-driving cars, health and human/public safety issues, and everything requiring human perception like face recognition, which generally takes an individual person within the range of 370 to 620 milliseconds. An augmented reality headset has to be able to detect a person at the same moment that the user does, and edge computation is more certain to be able to achieve this than a more traditional computer platform.

Efficiency

When analytical facilities are close to end users, advanced analytic and AI techniques may be executed on-the-fly. The system's various benefits may be attributed to this positioning near the system's periphery. There are also efficiency benefits that may be illustrated in the given example by using edge computation as a middle step between client computers and the broader internet. It is necessary for a client device to use external servers to execute technologically heavy video processing tasks. Because the calculations are being performed on edge servers in the local network, the file format only has to be transferred locally. In order to save bandwidth, it is best to avoid transmitting data over the internet. It's also possible to use speech to text. Recognition may be conducted locally, allowing for a large reduction in bandwidth requirements by sending the text towards the cloud instead of audio recordings.

Edge computing has proven effective in a variety of industries. Using cloudlets for wearable cognitive aid, researchers offloaded computing tasks CloneCloud, proposed by researchers, allows mobile apps to take use of the cloud in a novel way. A cloud-based solution may lower the amount of energy used by both systems. Depending on the extensively used MapReduce architecture, researchers developed an in-network decentralised computing for IoT data dubbed MR-IoT. Improved system support for general IoT applications is provided by the use of MR-IoT, which is based on an information-centric networking infrastructure. The naming strategy determines how MapReduce jobs are deployed and executed over the whole network. The processing capability of the network nodes and security concerns limit this method. Consumers can't use MR-IoT to transfer big amounts of data since the network would be overwhelmed by it. Another problem is that MR-IoT does not address authentication or confidentiality, disregarding the need for network security.

Only a few studies have looked into edge computing's failure tolerance. Using edge nodes to process information and then sending it to the clouds for additional processing and storage if needed, researchers have suggested a novel fault-tolerant structure for Internet of Things (IoT) applications. There are three levels to this design; first, applications isolations, second, information transmissions, and thirdly, the multi-cluster analysis. This allows the computer to be placed on the edge or in the cloud, depending on the need. During their research, the authors looked at edge node surveillance technology that could function even if their hardware failed. Crash failures are still tolerated by duplicating data such that a problem on either side of the network will not affect the analysis of data.

EdgeCons is only identified consensus approach with Paxos-centred approach, which can attain prompt ordering in the edge node. The EdgeCon element alludes to the gathering of Paxos sample into the edge node; however, it allocates Paxos governance centred on recent historical backgrounds of protocol consensus. The key objective of the protocol is that the simultaneous query of the edge network is ordered in a correct manner. For the large-scale distributed operations onto the edge computing network, the protocol enables quick event purchasing. In any case, this protocol is predicated on the unreasonable assumption that a cloud exists behind the edge network that is impervious to failure or partitioning of the internet.

It is possible to obtain consensus on a single information element or state using a variety of distinct consensus procedures in blockchain networks. For the execution of essential distributed procedures in a safe environment, Stanciu has examined the usage of blockchain as an edge computing platform. Transactions are validated and connected to certifying peers using Hyperledger Fabric in this continuing experiment. All identities are known on Hyperledger Fabric since it is an authorized blockchain platform. Real-time processing capacity is limited in terms of the amount of data that can be handled. There might be a disagreement about whether or not BFT state machine duplication is better than blockchain duplication.

According to [10], BFT state machine duplication has scale and speed advantages over PoW blockchains. These protocols can handle thousands of orders at about the same network pace even though BFT is notorious for its poor scalability. Block size and block regularity are what determine how well blockchain performs in comparison to other types of networks. PeerCensus, a novel system developed by researchers, operates as an authority that has been certified and is dependent on blockchain to effectively control firms entering the programs and the CA (Chain Agreement) protocols for conducting the essential transactions, despite the blockchain's constraints. The system guarantees that two-thirds of the terminals are in a "safe state" at any given time t, maintains account of systems membership, settles disagreements in the event of a blockchain split, and adds consistent quality to the system. This means that BFT (Byzantine Fault-Tolerance) protocols such as Zyzzyva or PBFT may work correctly in the system thanks to CA. Section III focusses on an analysis of the system whereby the architecture and the model analyses have been made.

III. SYSTEM ANALYSIS

Starting with an overview of the system's design, this part details the system paradigm and provides examples of threats that may be used against it.

Architecture

Originally, the IoT was not intended to include any kind of ambient awareness or self-control. Internet frameworks aren't necessary for ambient intelligence and automated control, though. Researchers (such as Intel) are now looking to combine the IoT with autonomous control, with first results suggesting that objects are driving the autonomous IoT. Deep reinforcement training is a potential strategy in this situation since most IoT systems are dynamic and interactive. Conventional computer learning techniques, like supervised learning, cannot be used to train agents (IoT devices) to act effectively in this context. When using a reinforcement learning approach, a training agent could identify and detect the current condition of its surroundings (such as the temperatures in the house), take action (such as turning on or off the HVAC system), and learn by increasing the total amount of rewards it gets over time.

This intelligence may be provided in three ways: via devices, through Edge/Fog nodes, and through Cloud computing. To some extent, this is a function of the IoT program itself, as well as its inherent temporal sensitivity. Avoiding a collision requires the camera on an autonomous car to identify obstacles in real time. It would be impossible to make these quick decisions if the data was sent from the car to the cloud and then returned to the vehicle. As a result, the whole procedure should be carried out in the vehicle itself. An ongoing study field in the development of smart objects is the incorporation of powerful machine learning techniques, such as deep learning, into IoT devices. Data mining and prediction of control actions are two further ways to maximize the value of Internet of Things installations. Traditional approaches like segmentation, support vector network, and random forests, as well as more modern methods like convolutional neural systems, LSTM, and finite difference autoencoder, have all been applied in the IoT area.

Avatars (Virtual elements), SOA elements, and web services will be more compatible and capable of operating autonomously (following they own objectives or sharing their most typical ones) with respect to their surroundings, conditions, and circumstances in which IoTs will be in the next few decades. One of the study trends in IoT is the collection, evaluation and processing of context data, and the object's capability to recognise the transformations in its ecosystems (faults that affect sensors) and apply effective mitigation approaches, which are plainly fundamental. To allow context-aware automations, modernized IoT initiatives and solutions apply different technologies that are plainly fundamental in real-life, but advanced types of understanding are essential to permit sensor units and intelligence cyber-physical systems to be employed in real-life settings.

Simplified, the IoT system design comprises gadgets at Tier 1, gateway at Tier 2, and cloud at Tier 3. In IoT equipment, devices comprise connected objects like sensors and actuators, especially those that link to an Edge Gateway using protocols like Modbus, Bluetooth, Zigbee, or custom protocols. Data gathering systems termed Edge Gateways offer features like pre-processing the information, ensuring cloud connection, employing frameworks e.g., WebSockets or the incident hubs, and also particular setting, fog computing or edge analysis. It is fundamental to provide a common image of devices to the top tiers so that they can manage them more easily. At the top of the stack is an IoT cloud application developed using microservices structure and secured with HTTPS/OAuth and often polyglot. Time series databases and asset stores that use backend information storage technologies are included in this category (e.g., Cassandra, PostgreSQL). It is common for cloud-based IoT systems to include an event queue and messenger app on the cloud layer that manages interaction across all layers.] According to some specialists, IoT systems are broken down into three tiers: edge/platform, platform/enterprise, and proximity/access/service.

IoT alludes to the architecture for the applications level of IoT, which considers the converging of data from IoT objects into the web apps in order to enable new use-cases for the IoT. There has been a trend toward combining traditional management systems with data mining and other new features to potentially automate the organizations of multiple numbers of interlinked objects to better control and flow of data in the IoT systems. This new architectural position is called BPM Everywhere.

The proposed system is integrated with a set of distributed procedures, which operate in cloud nodes and at the edge. Every edge node is integrated with a service, which handles edge object requests. The procedure within the edge node is known for its key mandate of acknowledging the vital requests, which originate from the end users, and responses of multiple instances of the cloud system. The infrastructures, the prevailing cloud computing systems and services will prevail wait period for the request from the multi-cloud nodes of the intelligent edge. Cloud instance is meant to acknowledge requests from the edge node and potentially perform the computations; create a cryptographic hash of the computation's output and send it to the edge node as an attachment. One cloud instance connects each edge node, and the request is sent to a group of edge nodes. Each edge node may deduce from the client request which peers also sent in a request. No communications are lost, replicated, or transmitted in the wrong sequence since the system's entities are all linked together through dependable channels (e.g., TCP). SHA 512 (cryptographic hash code) and the public key encryption provide data integrity and anonymity (e.g., TLS).

The result of the computation on the cloud is hashed using a cryptographic hash function. In order to keep the calculation secret from the edge nodes, only the decoder, or digest, is delivered. To avoid eavesdropping and ensure that each participant belongs to the system, all communications between the parties are encrypted and authenticated using a session key. **Fig. 3**

shows the multi-layer architecture integrated with cluster of edge nodes, denoted by CP (such as $1, CP, 2, CP, 3, CP$), which are in a similar location are interconnected to a collection of the multi-cloud instances denoted by c (such as c, 3, c, 2, and c, 1) through the backbone of the web. Contrary to the edge node, there are essential requirements concerning the geographical distributions of multi-cloud instances. Whenever clients submit a specific operation σ, these operations are duplicated to the closest cluster CP, and transferred to c.

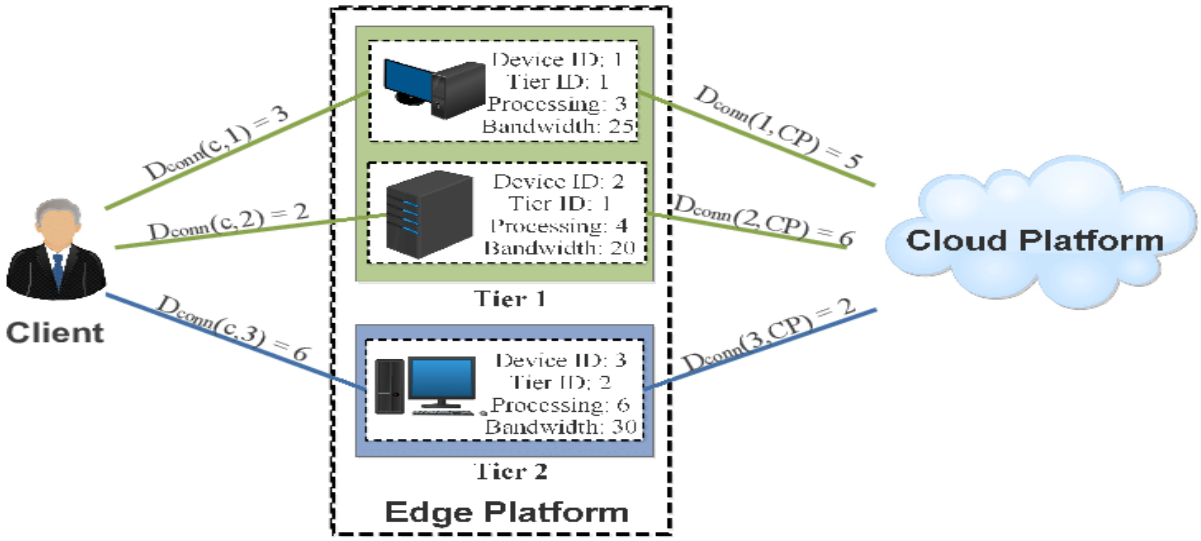

Fig 3. The architecture for the edge network with multi-cloud instances

The finding of each operation from each cloud system denoted by $v\sigma$, which appears as $h(v\sigma)$ message digest, is transmitted to CP, before being validated by $1, CP: i \in E$. It is visualized that $v\sigma$ is accurate whenever CP received $f + 1$ as $h(v\sigma)$. The recommended model has been configured with the f parameters. Within the distributed model, f is normally an upmost faulty result number. Given the $vi \in V\sigma$ output, the recommended model is able to tolerate not just the f fault, but the faulty replica. This is the case in case no faulty f replica is returning to the same inaccurate output. Visualize systems integrate n number of edge nodes within the message-communicating system, $n = 2f + 1$, each edge node $1, CP \rightarrow i \ldots . \approx 1,2,3 \ldots, n - 1.0\}$ is linked to single instance of cloud that is represented by $1, CP: i = \{0, 1, \ldots, n - 1\}$. From the n number of node, which incorporate the edge nodes n instance, our proposed model is precise in the cases where the nodes may also vote for the best value that is shown by v whether faulty edge nods, instances of the cloud or an incorporation of the two are considered.

Earlier on, this remedy seemed costly based on the numbers of resources required. All cloud instances are a collection of different nodes with the objective of executing computer-intensive activities. The edge node is tasked with the obligation of determining the methodology of voting, secluding the computations of more intricate activities to multi-instance within the cloud network. Whereas we apply $3f + 1$ multi-cloud instance with the variables sizes to achieve merits of using the multi-cloud ecosystem, it is fundamental to integrate $2f + 1$ edge nodes to this remedy. It is widely accepted in the scientific community that Byzantine fault tolerant networks are among the best methods for ensuring the dependability of distributed applications. Every $n = 3f + 1$ copies generate finite state machines and executes an action from the end user in a similar sequence, which is characteristic of a BFT system. This is a pricey option, but it provides safety and liveliness in difficult settings.

Cloud failures are regular, and the approach to reducing dependency on a single cloud operator and increasing resilience has been to adapt services to utilise several clouds. Although it is more costly, it has been demonstrated that using numerous cloud instances to operate services may help to survive BFT and cloud failures. This method, like the previous one, makes use of many cloud instances for the equivalent purpose, but it also includes edge nodes to verify the centrally computed results. The cost of deploying edge nodes to a multi-cloud setup may be regarded as insignificant.

There are no faults and the network is synchronized enough that the leader does not need to be changed for the n2 messages exchanged in typical BFT algorithms, which need five communication stages and n2 messages for traditional BFT algorithms in pleasant executions. Compared to existing BFT algorithms, our method involves fewer communication stages. In addition, our method does not place the burden of calculation on the client, as is the case with. The edge networks and cloud instances are accountable for the system's operation.

System Model

As with previous BFT algorithms, there is no requirement for a leader to submit values in this solution. There is only one edge node for each cloud instance. When a value is sent from the right cloud instance, each edge node recognises it as valid information. Clients and edge nodes are discussed in depth.

Algorithm

Algorithm 1:
Start
Step 1: Voting
Terminate v single values
Step 2: Agreement
Select v single value
Step 3: Termination
Communicate/process delays bounded to Δ
End

2f+1 cloud instances are used to ensure that f+1 identical outputs are always produced, even if there are random or malicious failures. Each edge node is assigned a value v by the cloud instance. The voting process is started at the edge nodes. To guarantee appropriate functioning, the algorithm must impose a set of properties:

Table 1. Properties useful for ensuring correctness of operations

Property	Definition
Voting	Correct procedures have to terminate with v single value
Agreement	Correct procedures should select v single value
Termination	Communication or processing delays have been bounded to Δ.

The first two characteristics pertain to the protection of life and property. In order for the algorithm to work, it must ensure that no valid process makes a choice that contradicts the other correct processes. The third characteristic relates to the ability to sustain life. Commit, Prepare, and Propose are the three steps of the algorithm represented in **Table 1**. Three edge instances and three cloud nodes are required for example in **Fig.4**, which has $f = 1$. In the next section, we'll explain how the system works from both the client and server points of view.

Client
It is sent to the edge nodes in an orderly fashion by the client C, which sends a signal Request, seq. Session keys developed specifically for the operation are used to sign requests. No action is taken on requests with invalid signatures. Request identifier seq ensures that a single Request is processed. The cloud instances themselves are referred to as "instances" in this context. Edge nodes are expected to send the client a seq, $H(V\ \sigma)$" sequence. To put it another way, $H(V\sigma)$ is a messaging digest due to the $\sigma, V(\sigma)$ calculations. As long as at least $f + 1 = H(V\sigma)$ is received by the client, the calculation has been completed successfully.

Edge Node
The requests from the client are communicate to the multi-cloud instances with the signal 'Request', seq σ CP pk. The edge nodes, which transferred the requests, and will undertake the voting procedure is identified by CP. Inputs are signed using the pk session keys that were generated specifically for the transaction. H (V σ pk is the cloud-based computation's result; it is represented by the expression ci, H (V σ). ci indicates the cloud that was used to do the calculation. $Vi = c, 1, H(V\sigma\ pk)$ is the outcome of the calculation on cloud instance ci. As shown in **Fig. 4**, the voting method is divided into three stages that begin as soon as an edge node 1, CP receives the first response from the cloud instance $c, 1$. During the Propose stage, each edge node receives Propose, seq, v,' which indicates the calculation result.

The c, 1 communicates with the 1, CP via the Propose message type. Seq is the beginning sequence number. It has been recommended that the value vi be taken from the outcome of computing in the cloud ci. 2f peers receive vi after receiving it from an edge node 1, CP in the Prepare phase of the transmission. C, 1's suggested value, vi, is shown in the following figure. 1, CP is the node on the edge that got the value. Commit tier begins after the value has been broadcasted. To effectively recognize the computations, edge node utilizes simplified majority protocol to chose v single values. As the last tier, Response, every node shall pass on results that are direct to the end users with a good message. The server occurrences with use cases are as follows:

(i) 'c' client accepts $f + 1$ matching response with the contents, Response seq, seq $v, 1, CP\ pc$ to indicate that the computations were completed successfully; (ii) Each edge node transmits the voted value to clients; (iii) It received a response from each ei from the viewpoint of the client. If the client gets a majority of the same answers, the calculation was a success.

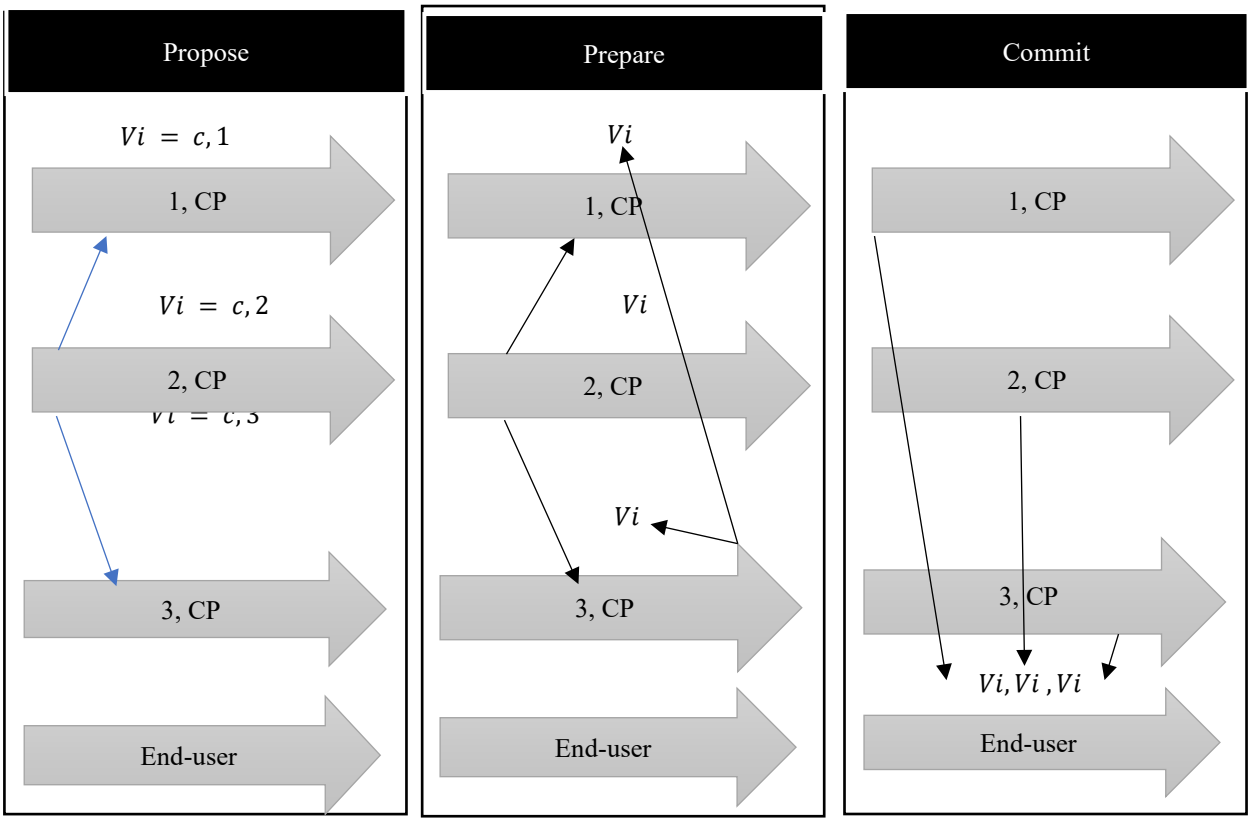

Fig 4. Precise execution of the voting approach

IV. RESULTS AND DISCUSSION

The edge nodes are not in sync with one another. To put it another way, any edge node may exhibit Byzantine behaviour or be in several stages at the same time. The following outcomes are possible when using this algorithm:

(i) Preparation is complete on one node, but the other peers haven't concluded the proposal process. Due to a lack of votes, the lone edge node which has already disseminated its value is unable to proceed.

(ii) If CP does not get vi from a ci or from its contemporaries, it cannot wait forever. Thus, a simple delay technique must be implemented to ensure that the service remains operational.

The system determines the parameter that ties operating and communications delays employing the following inequality: $|Tcj + Tkj\, Tci + Tki|$. Tc is the time it takes to process a message, and Tk is the time it takes for a message to arrive at its destination. A value will be sent to any $1, CP$ E node that gets vi from ci C during the predetermined time period after 1, CP receives vi. Furthermore, the time variation between $Tci + Tki$ and any other procedure j, $Tcj + Tkj$ is predicted to be smaller than. ej is judged to have failed if it receives no message after a certain amount of time. The Byzantine failure paradigm is used to represent an attacker. Random and malicious errors might occur on cloud and edge nodes. An 'omission failure' is interpreted by the algorithm as a cloud outage. **Fig. 5** depicts a situation in which a hostile insider compromises e1 and modifies the value of v1. Therefore, 1, CP sends the incorrect value v2 to the other nodes. There are just 2 left, and they may still vote accurately and give back a valid outcome.

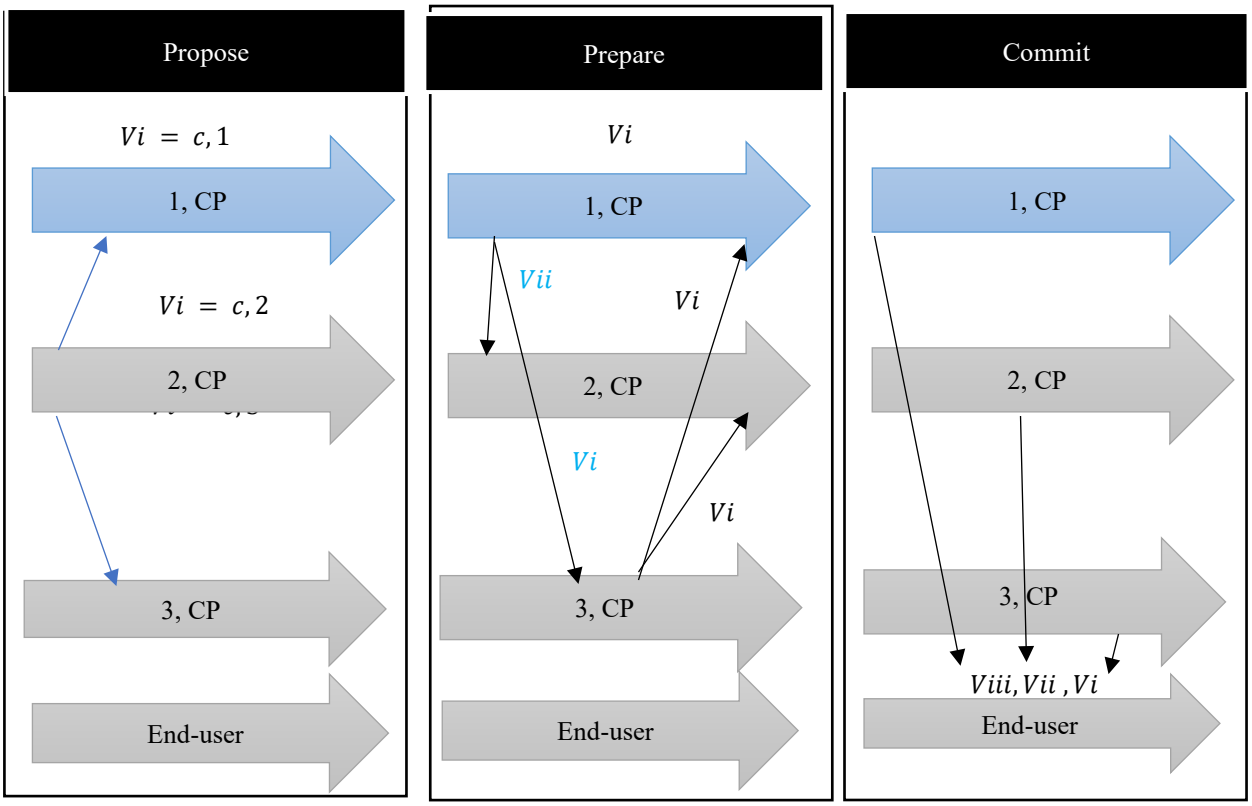

Fig 5. Execution of a single fault

2, CP experienced an arbitrary malfunction that changed the result to v3, and 1, CP was infiltrated by a hostile attacker as seen in **Fig. 6**.

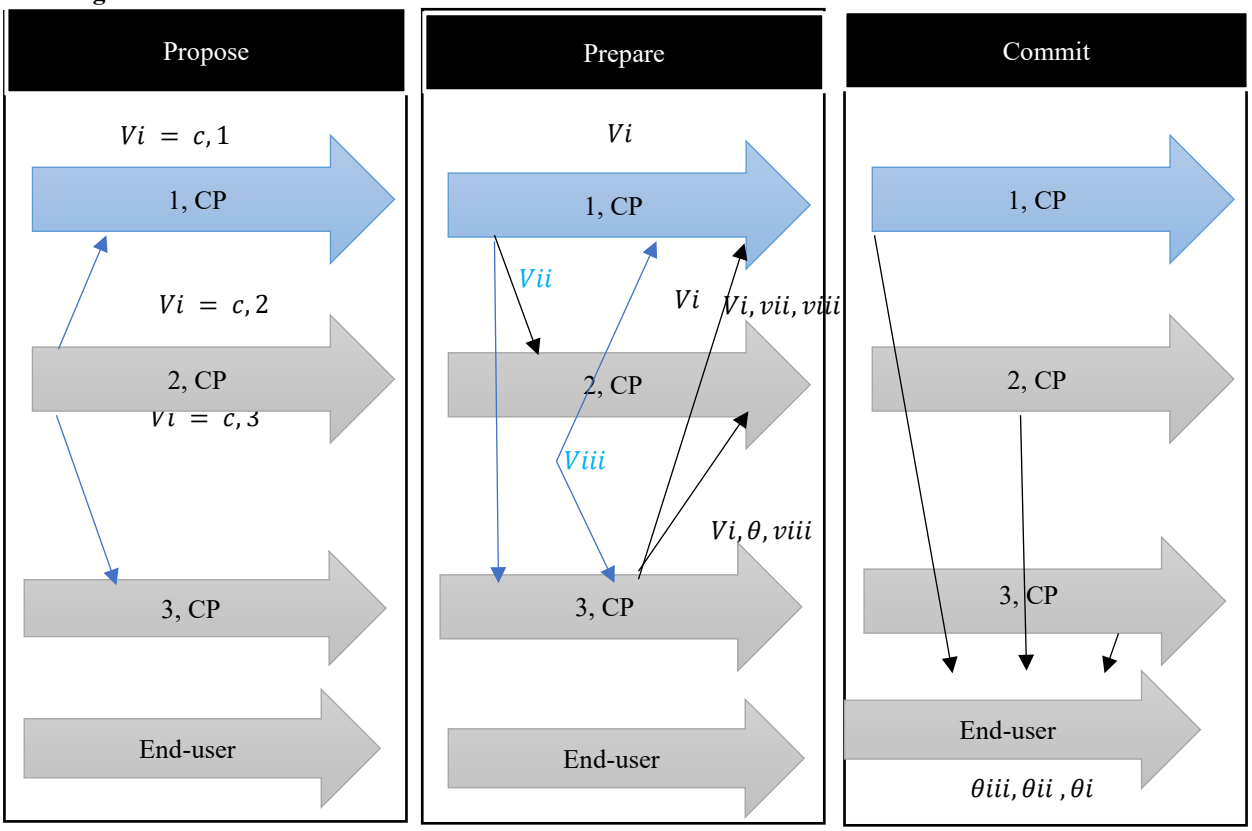

Fig 6. Executions of two faults

All of the 2, CP and 3, CP nodes are in a 'correct' state due to clouds c, 1 and c, 3 proposing the right value. Mixed variables are redistributed to the network edge in the Propose stage and shall probably be disseminated to the various

networking edge within the Prepare tier. For 1, CP, and 2, CP, there is a set of values known as "Vi, Vii, Viii," whereas for the node e3, the values are known as "Vi, ", "Viii. It's a result of 1, CP not transmitting the value v2 that is seen in 3, CP. It is because of this that Commit will fail to locate the right values, and therefore shall be communicated to the end-users. Thus, the model had two faults that are more compared to the original configuration, f = 1. 5 edge instances and 5 cloud nodes are required if we wish to tolerate f = 2.

V. ADDITIONAL APPLICATION

In addition to applications in voting, more application use scenarios, such as Internet of Things (IoT), factory automation and Augmented Reality, intelligent transportation, self-driving cars and other automated transportation infrastructure, are requiring edge computing clouds. Industry 4.0, for example, makes use of edge clouds to quickly solve a number of unanswered problems. Cooperative robots that need very low latency (microseconds or less) to provide a safe zone for their controllers have been suggested by researchers. In a constantly changing production system, Augmented Reality eyewear may help operators by conducting markerless item detection and precise monitoring in a plant. The only way to fulfil these use-cases is to cache and calculate the relevant data on servers that are near to each other.

If users want to implement virtual services at the edge of your network, you need the most cutting-edge techniques in virtualisation, NFV (Network Function Virtualization), SDN (Software Defined Networking). Utilizing sophisticated SFC (Service Function Chaining) approaches like, we envisioned a scenario in which users have discretion over which services are integrated in their network route. Using Domain Name System (DNS) based approaches, a client may identify specific edge computers that are offering the desired service in an open market. Services with extremely low network latency may help clients connect to the end-server much more quickly.

VI. CONCLUSION AND FUTURE RESEARCH

There are several ways to verify computations in a multi-cloud context, but one of the most common is to use edge computing. In the suggested method, edge nodes are used for two distinct functions. To begin, a collection of cloud instances will receive client-initiated operations. Second, to ensure that the computations made in the multi-cloud setting were accurate. Since the system assures reliability of the arithmetic in the presence of the Byzantine fault, this solution is suitable for the data-centric IoT application that depends on the system for various computational assessment. In the future, we plan to contrast our methodologies with those of others to learn more about how latency impacts system performance. An important issue in today's research is cloud database privacy. A large number of apps are being designed without any regard for security issues. Data processing and data storage are the primary functions of private cloud databases. Because of the higher level of security provided by the envisaged electronic voting machine, any vulnerabilities or fraud operations will be impossible throughout the voting process. Extending and implementing the suggested approach in public cloud secure platform; and implementing pre-processing and data selections in cloud server for voting applications are the main objectives of future development.

References

[1]. J. Ren, D. Zhang, S. He, Y. Zhang and T. Li, "A Survey on End-Edge-Cloud Orchestrated Network Computing Paradigms", ACM Computing Surveys, vol. 52, no. 6, pp. 1-36, 2020. doi: 10.1145/3362031.

[2]. "Australian edge in image processing", Data Processing, vol. 28, no. 7, p. 340, 1986. Available: 10.1016/0011-684x(86)90138-3.

[3]. A. Giannakoulias, "Cloud computing security: protecting cloud-based smart city applications", Journal of Smart Cities, vol. 2, no. 1, 2016. doi: 10.18063/jsc.2016.01.007.

[4]. A. Littlewood, "The erotic symbolism of the apple in late Byzantine and meta-Byzantine demotic literature", Byzantine and Modern Greek Studies, vol. 17, no. 1, pp. 83-104, 1993 doi: 10.1179/byz.1993.17.1.83.

[5]. F. Maggi, F. Tang, D. la Cecilia and A. McBratney, "PEST-CHEMGRIDS, global gridded maps of the top 20 crop-specific pesticide application rates from 2015 to 2025", Scientific Data, vol. 6, no. 1, 2019. doi: 10.1038/s41597-019-0169-4.

[6]. K. Peng, P. Liu, P. Tao and Q. Huang, "Security-Aware computation offloading for Mobile edge computing-Enabled smart city", Journal of Cloud Computing, vol. 10, no. 1, 2021. doi: 10.1186/s13677-021-00262-6.

[7]. A. Siriweera and K. Naruse, "Survey on Cloud Robotics Architecture and Model-Driven Reference Architecture for Decentralized Multicloud Heterogeneous-Robotics Platform", IEEE Access, vol. 9, pp. 40521-40539, 2021. doi: 10.1109/access.2021.3064192.

[8]. R. Brooks, "Distributed Sensor Networks: A Multiagent Perspective", International Journal of Distributed Sensor Networks, vol. 4, no. 3, pp. 285-285, 2008. doi: 10.1080/15501320701260816.

[9]. A. Alelaiwi, "Evaluating distributed IoT databases for edge/cloud platforms using the analytic hierarchy process", Journal of Parallel and Distributed Computing, vol. 124, pp. 41-46, 2019. doi: 10.1016/j.jpdc.2018.10.008.

[10]. H. Chai and W. Zhao, "Byzantine fault tolerance for session-oriented multi-tiered applications", International Journal of Web Science, vol. 2, no. 12, p. 113, 2013. doi: 10.1504/ijws.2013.056578.