sensors

MDPI

Article

BlockEdge: A Privacy-Aware Secured Edge Computing Framework Using Blockchain for Industry 4.0

Deepsubhra Guha Roy

Department of Internet of Things, School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, India; roysubhraguha@gmail.com or deepsubhra.guha@vit.ac.in

Abstract: Edge computing has its application in a lot of areas now, but with the increasing popularity and benefits, it suffers from some challenges such as data privacy and security. Intruder attacks should be prevented and only authentic users should have access to data storage. Most of the authentication techniques apply some trusted entity to undergo the process. Users and servers both have to be registered in the trusted entity to get permission of authenticating other users. In this scenario, the entire system depends on a single trusted entity; so, a single point of failure can cause the failure of the total system, and scalability issues are there also. To address these issues remaining in the existing systems, in this paper, a decentralized approach has been discussed which is capable of eliminating the concept of a single trusted entity by introducing a blockchain paradigm in edge computing where every time a user or server wants to enter the system, it does not have to register itself manually, but the authentication process is carried out throughout the scheme automatically. Experimental results and performance analysis prove that the proposed architecture is definitely beneficial and it outperforms the existing ones in the concerned domain.

Keywords: security threat; edge computing; distributed ledger; blockchain; digital signature; Industry 4.0

Citation: Guha Roy, D. BlockEdge: A Privacy-Aware Secured Edge Computing Framework Using Blockchain for Industry 4.0. *Sensors* **2023**, *23*, 2502. https://doi.org/10.3390/s23052502

Academic Editor: Nikos Fotiou

Received: 15 November 2022
Revised: 18 December 2022
Accepted: 19 December 2022
Published: 23 February 2023

1. Introduction

Cloud computing and its associated technologies are some emerging paradigms in today's world as they have proved their efficiency in having irresistible demand for pervasive intelligence in various fields. Despite having different benefits, the cloud still suffers from challenges such as resource centralization leading to end users' separation from their corresponding cloud service which results in latency increase and jitter ultimately. They have become a major issue as the requirements of end users are dynamic mostly, and the introduction of ubiquitous network architecture took place just due to this. Big data processing components include data mining, data acquisition, pre-processing, and pattern recognition using real-time applications such as healthcare monitoring and control, GPS navigation, Internet of Things (IoT) where low latency is highly desired to satisfy the end-user requirements.

Social networks, smart cities, smart grids, Internet of Everything (IoE) are the main areas of Industry 4.0 which benefited a lot from the cloud paradigm and face different challenges as well. Jitter and latency have been two major issues as the user requirements keep changing always. To deal with these situations, ubiquitous computing came into the picture and proved its efficiency in its concerned domain, sometimes by introducing a content delivery network (CDN) which is all about web content caching along with computation accompanying it. This work is concerned with several shortcomings found in the conventional centralized architecture of cloud computing platforms in Industry 4.0, which can be summarized as follows:

- Multi-source data processing: The requirements of multi-source data processing cannot be met by the linearly growing capability of the cloud at the network edge.

- Data transfer speed: The speed of data transfer and bandwidth suffer from a bottleneck as long-distance interactions result in latency and computing resource wastage.
- Privacy and security: During the transmission, privacy and security can become major issues for edge device users.

Motivation and Contribution

While scientists are working on overcoming these issues, there are several security challenges that remain unaddressed till now. In this paper, privacy preservation and data security have been attempted to be achieved. The authentication schemes have been implemented using an extended decentralized form contributing to the following factors.

- Addressing the gaps in conventional architectures: After reviewing recent research works and their loopholes in the concerned field, the blockchain mechanism has been integrated with edge computing and the expected challenges of this task have been evaluated also.
- Use of permission blockchain: Among two categories of blockchain, i.e., permissioned and permissionless, the principle of the first one has been followed during the deployment of the comprehensive decentralized scheme.

In brief, the proposed work contributes to the following outcomes in the field of edge computing, also shown in Figure 1.

- Track each transaction and entity selection through identification and authorization.
- Access control privacy mechanism while using data transfer in image format from the central cloud to the edges.
- Peer-to-peer transparency and tamper-proofing mechanism through blockchain help increase data transfer speed.
- Reduction of third-party involvement cut off the computational cost of multi-source data processing.

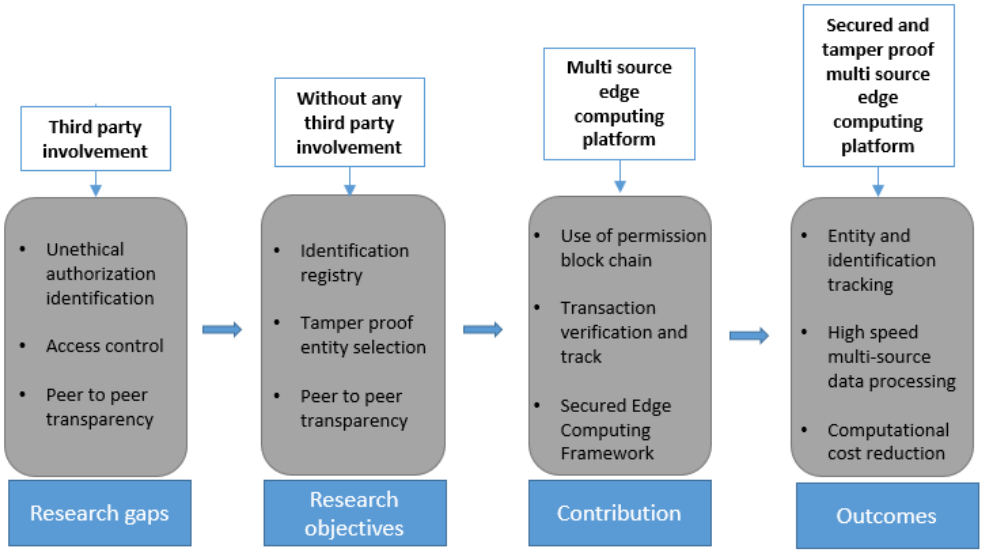

Figure 1. Contribution of the proposed system.

The next section analyzes different related studies along with their benefits and drawbacks. The motivation behind introducing this proposed method is explained which is actually overcoming the loopholes found in the existing literature and the major contributions are discussed also. This proposed methodology is presented in Section 3 and the experimental setup along with the performance analysis of the outcome is discussed in Section 4. The concluding statements and probable future directions of this research are discussed in Section 5.

2. Literature Survey

Mobile edge computing suffers from different security threats which differ from the cloud computing paradigm because of the additional features of edge computing namely heterogeneous architecture [1], location awareness, and mobility support. In each network tier, the threats should be addressed to protect the end user data. To prevent intruders from accessing, misusing, and abusing resources, an access control mechanism should be introduced where a unique identifier can be assigned to each and every entity in the network for better authentication in the system.

2.1. Challenges in Edge Platform

There are

different forms of edge computing, i.e., fog computing [2–5], mobile cloud computing [6–8], mobile edge computing [9,10], dew computing [11], and others; Table 1 has differentiated them by using some parameters, i.e., deploying infrastructure, ownership, deployment platform, purpose, and target users. However, all of them have the same aim to bring cloud services to the network edge so that the distance from the cloud infrastructure to the end user device gets more reduced than before for better service quality by improving response time.

If the network core is replicated at the edge of the network, computation offloading gets easier and network burden is reduced by preserving privacy and security also.

Table 1. Comparison among different edge paradigms.

	Dew Computing	Fog Computing	Mobile Edge Computing	Mobile Cloud Computing
Purpose	Without using the features of cloud, provide service to customers in on-premise devices	To satisfy the requirements of delay-aware services and IoT geographic distribution	To reduce the latency, shift the cloud storage and computation from network to network edge	To improve the quality of service delivery, a delegation of computation and storage to edge devices
Deployment platform	User device	Edge and closer to edge devices	Network edge	Edge devices
Deploying infrastructure	On premise devices	Heterogeneous servers, switches, access points, routers, etc.	Heterogeneous servers, Various base stations, Radio access points, etc.	Heterogeneous servers, User devices, Various base stations, Radio access points, etc.
Ownership	Private and single owners	Private and single owners	Tele-communication organizations	Private and single owners
Target users	Common Internet users including mobile data user	Mobile data users	Mobile data users	Mobile data users

2.2. Functionality of Blockchain

Among two types of the blockchain network, the permissionless blockchain cannot control the entry of users in the network; thus, every node in the network is bound to participate in a broad consensus for getting verified. The consensus protocols are stricter in this case for avoiding conspiracies in the network, e.g., bitcoin applies Proof of Work (PoW) consensus for this purpose. On the other hand, permissioned blockchain has a control over the entries in a network consisting of some limited users participating in the consensus and others as observers. All the nodes are detectable within this system and regulated using simpler algorithm and having lesser intensity than the public network.

2.3. Characteristics

Blockchain can be characterized by the following points based on some principles followed by it to get more adopted in various systems. The attributes include:

2.3.1. Peer-to-Peer

Trusted third party (TTP) needs to exist in the network during the transaction, otherwise every node gets the same privilege and right to connect with each other. The

network duties and controls are circulated among peers to increase the overall security in the system [12].

2.3.2. Transaction Verification

Integrity and authenticity of the ongoing transactions are verified and maintained by every node participating in the task as miner or observer which, in turn, leads to an increased trust in the network and a transaction gets approved only after getting permission from the maximum of the nodes.

2.3.3. Transparency

The initial transaction in the system is responsible for checking and auditing each and every transaction where all the users are allowed to access a particular ledger which, in turn, results in a difficulty for attacking and manipulating individual node's ledgers [13].

2.3.4. Trust

In a traditional centralized cloud and edge platform, trust and security are major issues to be addressed as before a transaction, nodes require trust between each other. In blockchain-based networks, cryptographic rules are the only ones upon which the nodes rely and transact [14].

2.3.5. Tamper Proofing

Blockchain is designed in a manner where to attack and modify a particular block content, the intruder needs to subsequently alter the next blocks also. As the corresponding hash values get changed, such an action requires consensus from the maximum nodes in the network. The role of different nodes in a blockchain can be of three categories:

- Light node: It is responsible for storing a particular part of information observed in the system.
- Full node: It is responsible for storing all the information of the recorded transmission in the blockchain network.
- Forging or mining node: It is responsible for processing transactions, compiling the allowed transactions, and adding the block in the blockchain for further broadcast in the entire system.

2.4. Working Mechanism

The working mechanism of blockchain can be described by the following steps.

Step 1 When two users initiate a transaction based on their cryptographic key pair, all the other nodes are asked in the network to check the transaction's validity based on the cryptographic public key and the corresponding signature which is a private key by nature.

Step 2 Upon confirmation received from neighbors, the transaction is forwarded to all the closely located nodes and is received after a certain period.

Step 3 Upon getting approval from the consensus, the transaction is added to the chain consisting of the other valid transactions and, in turn, gets ordered and packaged within a timestamp by the mining process.

Step 4 While the mining process is performed, the participating nodes have to undergo a computationally concentrated task depending on the protocol followed, such as Proof of Work. After completion of this protocol, the miner gets allowed to broadcast and add the newly created block to the chain.

Step 5 The blocks in the chain consist of mainly five parts, namely current block hash, previous block hash, timestamp, data and, index, where the index keeps track of the sequence of the blocks by assigning each block a particular number. The hash function is responsible for converting the block contents to exclusively allotted output, each having a static length [15]. The current block refers to the hash of the previous blocks always, which, in turn, invalidates all the next blocks when

some hash value gets changed. This feature of the blockchain helps to lessen the chance of getting tampered by unauthorized users.

A comparison of this proposed approach with four existing related techniques is presented in Table 2.

Table 2. Functionality comparison of the proposed block-edge approach with a few existing research works.

Ref. No.	TTP	User Trace Ability	User Privacy	Security in Mutual Authentication	Man in the Middle Attack Prevention	Re-Play Attack Prevention	Imper-Sonation Attack Prevention	Data Security
[16]	Yes	Yes	Yes	Yes	No	No	Yes	No
[17]	Yes	Yes	Yes	Yes	No	No	Yes	No
[18]	Yes	Yes	Yes	Yes	No	No	No	No
[19]	Yes	Yes	Yes	No	No	No	No	Yes
[20]	Yes	Yes	Yes	No	No	Yes	No	Yes
[21]	Yes	No	Yes	No	No	No	No	Yes
[22]	No	Yes	Yes	No	Yes	No	Yes	Yes
Block-Edge	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes

3. Proposed Methodology

The design principles and workflow of this proposed scheme is introduced in this section as shown in Figure 2. A progressively decentralized blockchain-edge integrated framework has been proposed to increase the security and effective delivery of service in the edge platform. A novel privacy-preserving authentication system, inspired from [22], has been illustrated where there is no requirement of any third-party interrogation in registration or mutual authentication of the nodes involved in it.

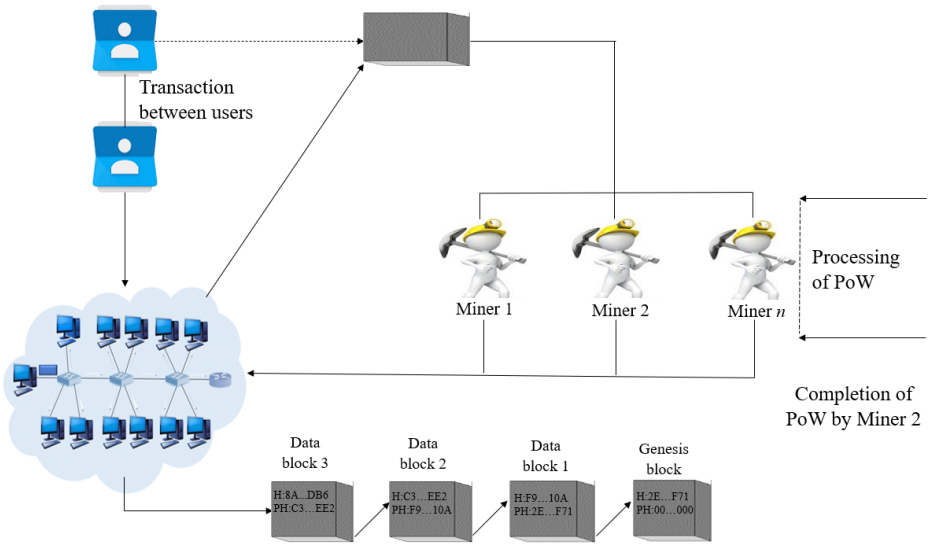

Figure 2. Workflow in the proposed system.

3.1. Proposed Architecture

The architecture of this proposed model includes different components, i.e., edge servers, edge devices, and cloud servers having wired as well as wireless connections between them as shown in Figure 3. Access allowance of a user to an exact server depends on the validation result of all the service providers by consensus in that particular network. When a service provider gets recognized as an element of the decentralized blockchain platform, the related activities get accommodated by a layer created externally around

it, which adapts the distributed ledger. This distributed ledger's significant consequence helps track every validated user identity, predominantly using their public keys. This layer also tracks the sub-transactions between the edge device and service provider to have better transparency between all the network elements.

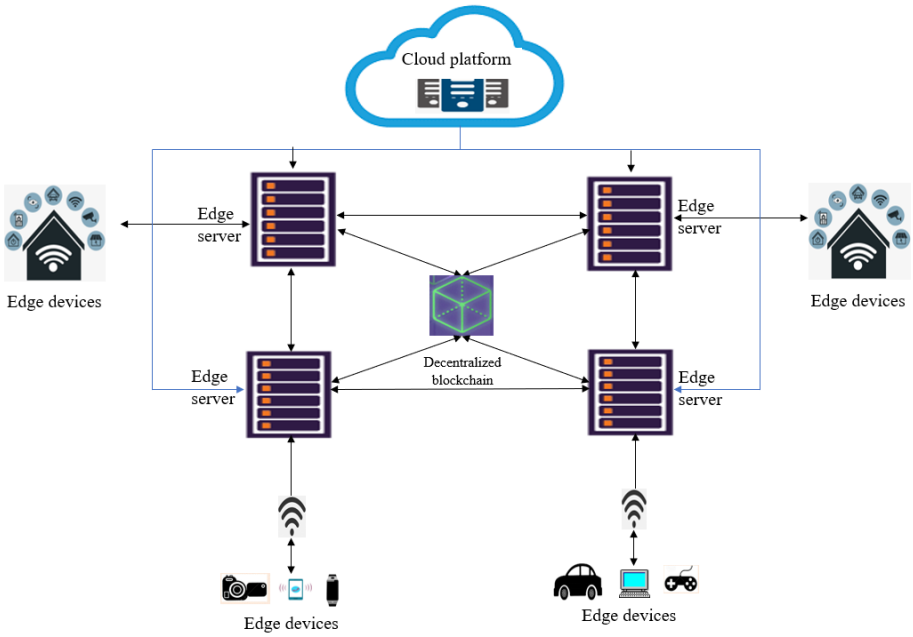

Figure 3. Architecture of this proposed model.

3.2. Registration, Login, and Authentication of Node

Each and every node having a decentralized blockchain scheme installed in it gets registered either as an edge device or as a service provider. After satisfying all the predetermined conditions, only if the node is recognized and hence is a genuine service provider, it involves a genesis node to trigger trailing nodes, i.e., edge devices for performing a successful transaction as shown in Figure 3. The functionality can be formulated as follows. Let D be a generator of an additive cyclic group G_1 and G_2, a group of cyclic multiplication. Let the bilinear map $e : G_1 \times G_1 \rightarrow G_2$ satisfy the following features, while d is the prime order of $G1$, $G2$ and $g = e(D, D) \in G_2$.

Bi-linearity: $\forall N, M \in G_1$ and $\forall$ random numbers $b, a \in Z_p, e(bN, aM) = e(N, M)bA$. Computability: There exists an algorithm for computing $e(M, N) \forall M, N \in G1$. Non-degeneracy: If 1 represents the identity element of $G1$, there exists $M, N \in G1$ where $e(U, V) \neq 1$. Three hash functions represented by $H_i : \{0,1\}^* \rightarrow \{0,1\}^n$ get (2) Please check if the comma should be the decimal point, like 0.1, if yes, please modify. generated where i = 1, 2, 3. A fuzzy extractor bio-cryptosystem has been implemented in this system to secure edge devices in this network which generates an input pair denoted by $\{\alpha, \beta\}$ for a user U_i and the probabilistic generator function $Gen(.)$. The public parameters created and stored at the service provider's end to be forwarded to all the users demanding access include $\{H_1, H_2, H_3, D, d, e, G_1, G_2, Gen(.), Rep(.), \gamma\}$ and follow the steps described below.

Step 1: Upon installing the decentralized client, every user U_i selects an identity ID_i and forwards it to the corresponding service provider SP_j.

Step 2: Upon receiving the id, the service provider shares that with all the service providers to validate it. After completing the validation process by most of the nodes, the public parameters get shared with the user U_i by the service provider.

Step 3: Whenever the user $_i$ receives pre-defined parameters, it selects $a \in Z_p$ and creates the private key as $S_i = a.ga.modd$ and the public key as $PK_i = S_iP$ and forwards them to the service provider immediately.

Step 4: The service provider adds the identity and public key to its chronology, also known as a record, and dispatches the narrative to all the connections and participating service providers to add it to the trail of the distributed ledger [22]. All the corresponding public keys and identities of the network $[(SP_1, PK_1), (SP_2, PK_2), \ldots, (SP_n, PK_n)]$ get shared with all the service providers in the network.

Step 5: The user U_i generates $(\alpha_1, \beta_1) = Gen(f_i)$ where f_i is the biometric of user i and the following parameters get encrypted as

$$Q_i = E_{H_2}(ID_i||PW_i)[(SP_1, PK_1), (SP_2, PK_2), (SP_3, PK_3), \ldots, (SP_J, PK_J)] \quad (1)$$

$$q_i = E_{H_1}(\alpha_i)(ID_i||PW_i||S_i) \quad (2)$$

For encrypting the identities, public keys get generated using a user identity and private password known only to the user. So, the service provider record and public key cannot be extracted by someone without knowing $(ID_i||PW_i||f_i)$, and malicious attacks [17,23] in the system are prevented in this way.

3.3. Data Transmission between Service Providers and Nodes

After successfully validating nodes, they get added to the already connected nodes with the service provider. By creating a digital signature, they start to exchange and transmit information with the direct service provider or the service providers with which they are connected via their own service providers. Once a user is validated, it will not have to get registered again in the network even if they connect via some other service-delivering entity.

3.3.1. Creation of Digital Signature

To authenticate and maintain integrity in user identity and data, a digital signature has been created in this proposed scheme using elliptic-curve cryptosystem-computing the transmitted message content between various nodes in the network. The process mainly includes generation and verification where generation takes ith user detail input in the form of $U_i : ID_i, PK_i$, and m. Upon selecting anyone among the random numbers $a, b, c \in Z_p$, the computation takes place with the following:

$$f = (H_1(m) \cdot ID_i \cdot P) \quad (3)$$

$$d = (f \cdot PK_i \cdot a) \in Z_p \quad (4)$$

The created signature for the message m is denoted by $Sign_i = (f, d)$, the device of the recipient must undergo a verification stage where after receipt of the file, the receiver calculates f . Using the bilinearity rule, the signature validity gets tested and the transaction gets approved when it satisfies $e(Bd, aG) = e(f \cdot PK_i, G)ba$.

3.3.2. Initiation and Exchange of Transaction

The initiation and exchange of transaction can be described as follows.

Step 1. Upon receiving a request from a user for initiating a transaction by creating a file called f_t, the service provider instructs the user U_i to choose a random number $v \in Z_p$ and calculates the exchanging transaction as

$$H_2(b||PK_i||Sig_i||EPK_j(F_t)||PK_j||H) \quad (5)$$

Step 2. Upon receiving a transaction from the ith user U_i, the service provider checks its previous hash value and compares it with the last transaction performed between those particular two entities. If it is not matched, then the file is immediately driven out of the system.

Step 3. The transaction-requesting source gets verified by the public key. If it does not either correspond to digital signatures from both the end or the identities provided prior, then the transaction request immediately gets eliminated from the queue.

Step 4. The destination gets verified by the service provider to check whether the recipient is intended, or it is just forwarding the request. Based on it, the service provider confirms the signature's authenticity in the first case, and the transaction gets decrypted with the public key. If the same was encrypted, then the file gets rejected in the second case, as it fails to satisfy the requirements of the system to get processed. When a validated user U_i tries to connect with some service provider with which it is not directly connected, it connects through its connected servers as shown in Figure 4. Black double arrows indicate service exchange between user and service provider, here one user via three service providers.

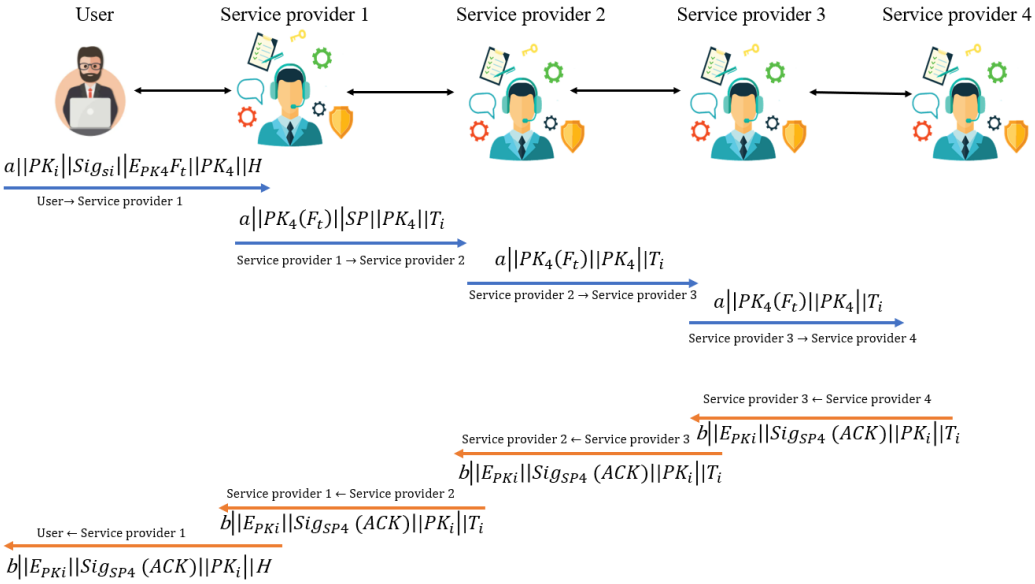

Figure 4. Transaction.

Step 5. After taking delivery of a transaction, if a service provider realizes that the recipient is not the desired one, in that case, it packs the transaction again newly by deleting the hash value and the source. A random number gets generated after that for computation and the signature of a private key

$$H_1(b||Sig_SP1||EP_K4(F_t)||PK_4||T_i) \quad (6)$$

The newly packed transaction gets forwarded to all its neighboring nodes.

Step 6. Upon receiving a transaction, the identity of the sender gets verified at first. If the sender's signature is not genuine, then the transaction gets dropped, and if it is valid, then the next step taken is to verify the recipient's identity. If the receiver is not the desired one, then the transaction gets transmitted to all the neighboring nodes and all of the service providers receive it within a short, stipulated time.

Step 7. The already forwarded transaction ids get eliminated from the network by service providers for avoiding network flood. The transactions which successfully get decrypted and the user validity gets verified, their ids get discarded, but if some transaction fails to do so, the transaction record is fetched again and tries to get authenticated for resource access. However, there is a time limit up to which a transaction is allowed to retry, otherwise, access gets revoked.

Step 8. Upon completion of reading the message which was carried by the transaction from the sender to the receiver, the recipient adds up the transaction to a dedicated block.

Step 9. Upon successful delivery of the transaction, the receiver sends an acknowledgment to the sender as

$$H_2(b||EPK_i(Sig_SK4(ACK)||PK_i)) \quad (7)$$

Before installing the BlockEdge client, the users are grouped using a clustering algorithm. Among n number of users, first, random k number of users are selected and the rest $n - k$ number of users are represented by medoids based on the distance from the medoids. The clustering method looks as follows (Algorithm 1).

Algorithm 1 Clustering

Input: K,C
Output: C_{final}
1: function CLUSTERING(K,C)
/*declare the center of the cluster*/
2: RandomCenter[K]=V
/*Initiate assigning clients*/
3: for $C_1 \in C$ and $V_j \in V$ do
4: if $cos\theta(C_i, V_j) gt Cos\theta(max)$ then
5: Add C_i to the cluster of V_j
6: end if
7: end for
/*Modify the center of the cluster*/
8: $C_{final} = C_i$
9: Go to the next iteration
10: Return C_{final}
11: End function

For every point, the criterion function is computed and the point which corresponds to the minimum function value is selected as the updated medoid. The process repeats until the medoid points stop changing and reached the maximum predefined number of iteration. The workflow in initiation process of nodes is shown in Figure 5.

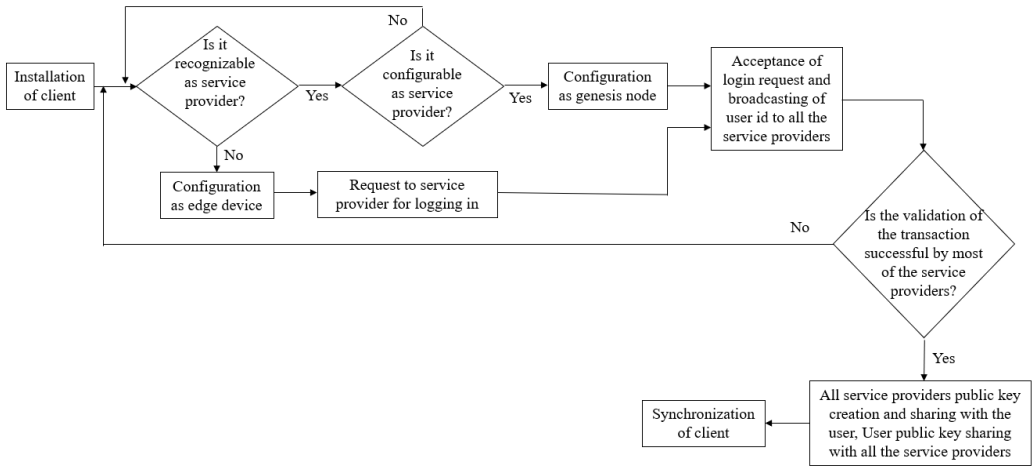

Figure 5. Work flow diagram of node initiation in the proposed approach.

3.4. Cryptoanalysis

The proposed method's capability of preventing the probable attacks originating from external as well as internal attackers can be analyzed by the following points:

i. The polynomial adversary monitors and controls unreliable transactions established on malicious links and, in turn, is able to modify, delete, replay, and reroute the transactions.
ii. The private key of any network object at any random time has increased entropy, so it cannot be predicted within a polynomial time.
iii. Every public network parameter is known to the polynomial adversary. It has the right to alter the hash values of ongoing transactions as well as the transactions to be added in the chain.

iv. The adversary polynomial is not capable of decrypting every ongoing or complete transaction of the network, it can only capture the leakage in information.

3.4.1. User Confidentiality and Impersonation Attack

User record protection during a loss or theft is taken into account in this phase. Any kind of impersonation attack is attempted to be prevented in this proposed system. Suppose, the adversary intercepts every exchanged transaction in the system, as the hash value $E_{H_1}(\alpha_i)(UID_i, P_i, S_i)$ is encrypted with biometric ith user details such as secret key K_i, user id UID_i, password P_i, thus accessing the credentials would be a difficult job for it. For the retrieval of the service provider's public key and identity details such as (UID_i, P_i, S_i), at first, a biometric key α should be created using a reproductive deterministic function $repr(.)$ consisting of β and f_i as input. Thus, it is hard for the adversary to retrieve user details and impersonate the user in order to initiate an attack in this proposed system.

3.4.2. User Non-Traceability and Secrecy

Fresh users in all the sessions only participate in transaction exchange for a random time being. That is why transaction linking is not possible within different sessions which, in turn, leads to non-traceable identity generation for users to prevent adversaries from tracking them in spite of a successful interception. Thus, this proposed design is secure enough for users interested to transact within the scheme.

3.4.3. Impersonation Attack to Service Provider

To maintain confidentiality from both sides, the service provider's identity is kept secret also, and it is transmitted only upon valid encryption. If the adversary captures a transaction between the server and some entity, it is still difficult for it to impersonate the service provider to do that, as it needs the digital signature designed by only the service provider's private key. Additionally, the hash function referencing also prevents it from interacting or discussing with the nodes residing in the network.

3.4.4. Man in the Middle and Replay Attack Scenario

Suppose some adversary tries to capture a transaction between two service providers or a service provider and a node. In that case, the system needs the hash referencing of the previously conducted transactions among the group; so, any initiatives with wrong hash values get automatically identified and treated accordingly. The timestamp also lets the servers know when a transaction replay attack is attempted. So, this proposed scheme is highly secure from these adverse scenarios.

4. Experimental Setup and Result Discussion

The experimental setup and the result is discussed in this section along with the performance analysis of the proposed approach with different parameters.

4.1. AVISPA-Based Cryptoanalysis

For security protocol-based model analysis, Automated Validation of Internet Security Protocols (AVISPA) is a useful tool where the protocols are mainly designed using High Level Protocol Specification Language (HLSPL). hlpsl2if is the translator which converts this high-level language to intermediate format (IF) having four backends, called Tree Automata-Based Protocol Analyzer (TABPA), Sat-based Model Checker (SATMC), CL-based ATtack SEarcher (CLATSE), On the Fly Model Checker (OFMC), which are able to read the high-level specifications in the IF. Among these four backends, OFMC is efficient to detect attacks and verify the protocols. Thus, it has been selected by us and the considered simulation goals have determined the simulation result.

Through the Dolev-yao channel the proposed scheme has been modeled in AVISPA, which includes an intruder to test the scheme's maliciousness when a transaction is tra-

versed in the established group. Any transactions and data dispatched to a genuine user(s) can be modified by intruder *i*. The simulation targets two major outcomes are:

i. Testing the Dolev-yao model.
ii. Testing the replay attack scenario.
iii. Execution capability checking of the non-trivial high-level protocol specification language.

To check the Dolev-yao model, OFMC at first checks the probability of initiating a man-in-the-middle attack by the intruder. The experimental outcome has shown that the intruder has failed to initiate that attack, so the proposed scheme guarantees robustness as desired. In both stages, the intruder has been provided with the transaction information between different users in the network in both stages. The obtained outcome has proved this proposed approach to be an effective one against the active attack and passive attack as well. The simulation results for stage 1 and stage 2 are shown in Figures 6 and 7, respectively.

```
% OFMC
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/home/span/span/testsuite/results/stage1.IF
GOAL
AsSpecified
BACKEND
  OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime:0.06s
visitedNodes:26 nodes
Depth:4 plies
```

Figure 6. Simulation outcome in stage 1.

```
% OFMC
SUMMARY
SAFE
DETAILS
BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
/home/span/span/testsuite/results/stage2.IF
GOAL
AsSpecified
BACKEND
  OFMC
COMMENTS
STATISTICS
parseTime: 0.00s
searchTime:0.12s
visitedNodes:56 nodes
Depth:7 plies
```

Figure 7. Simulation outcome in stage 2.

4.2. NS2-Based Practical Simulation

The applicability of this proposed approach in the practical world has been tested using the NS2 2.35 simulator. Network protocols are simulated widely using event-driven simulation tools such as NS2. Three different situations in two different stages have been taken into account to check the applicability here. The impact of the system on different resources of the service providers each having a uniform number of mobile users in all three situations has been focused on, where the impact of computational resources on mobile users was neglected, as the considered metric values in all the simulation stages were virtually the same in that case.

The hash function has kept in length as 160 bits, randomly created number size as 128 bits, and the duration of authentication [24], digital signatures, and transaction ids as 32 bits long. Mobile users were consistently distributed among service providers in both stages in all three situations. The first stage is concerned with connecting a user to its corresponding service providers for computing and sending the transaction file and acknowledgment from the service providers to the user. The second stage is concerned with the exchange of transactions between the user and the intended service provider via the service providers to which the user is connected to. In the first stage of this experimental simulation, two transactions took place between *i*th user and *j*th service provider, where the first one was sending the user biometric to the service provider, and the second one was the acknowledgment from the service provider to the user. The transaction composition parameters were set as 554 and 364 bits, respectively. As Service provider 4 is not directly connected to the user, the intermediate Service Provider 1, Service Provider 2, and Service Provider 3 helped it to reach the destination by forwarding the transaction between the entities. The simulation parameters are discussed in Table 3.

Table 3. Factors considered for simulation in NS2.

Factor	Description	
Operating system	Ubuntu 20.04 LTS	
Simulation stage	Stage 1	Stage 2
Number of service providers	11 for situation 1, 2, 3	5 for situation 1, 2, 3
Number of service providers	11 for situation 1, 2, 3	5 for situation 1, 2, 3
Number of customers	Situation 1: 15 Situation 2: 25 Situation 3: 30	Situation 1: 12 Situation 2: 17 Situation 3: 33
User movement	6 m	
Primary energy of every service provider	600 J	
Primary energy of every user	700 J	
Simulation duration	1700 s	

The factors which were analyzed in two deployment platforms, i.e., service providers and mobile devices are energy consumption in mW, load in bps, and throughput in bps. The analysis has been performed depending on the calculation of the average of total packets sent, total packets received, packet size in bits, starting energy of *j*th service provider, starting energy of *i*th user, simulation time, and residual energy.

4.2.1. Impact on Network Load

The average load in the network consisting of service providers and mobile nodes has been analyzed and calculated by

$$Load = \frac{(Packets\ sent + Packets\ received) \times Packet\ size}{Simulation\ time} \quad (8)$$

In the stage 1 simulation, the service provider's average load values in three different situations were 80.63 bps, 170.87 bps, and 225.96 bps, respectively. Since the network load increased with an increasing number of users on the service provider side with time, the second stage required three more service providers to meet the requirements of the user and the average load in the three situations became 47.98 bps, 69.56 bps, and 102.34 bps, respectively. As the source node had to involve three other service providers to forward the transaction to service provider 4, the network had to deal with a slight load increase but it was never a burden to the service providers at all, as they have large capacities and efficiency to perform the task.

4.2.2. Impact on Network Throughput

The network throughput defined as transmitted bit per unit time has been calculated by

$$Network\ throughput\ = \frac{Packets\ received\ \times\ Packet\ size}{Simulation\ time} \quad (9)$$

In the first stage, the average throughput has been obtained as 90.97 bps, 190.46 bps, and 280.42 bps, respectively, in situation 1, situation 2, and situation 3. The rise in mobile device users has resulted in an increased number of transactions in the system which, in turn, has obtained increased throughput in stage 2 as 145.41 bps, 223.98 bps, and 245.26 bps in the three situations, respectively, as displayed in Figures 8 and 9.

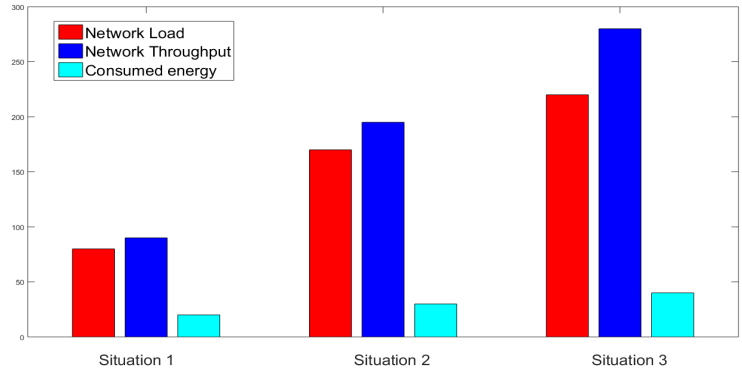

Figure 8. NS2 Simulation in Stage 1.

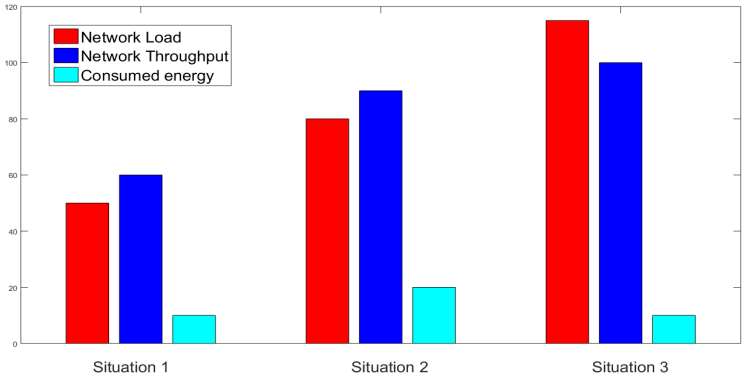

Figure 9. NS2 Simulation in Stage 2.

4.2.3. Impact on Consumed Energy

The impact of the proposed scheme on node energy consumption has been analyzed since many edge devices suffer from energy and resource restrictions. The consumed energy has been calculated by

$$Consumed\ energy\ = \frac{Initial\ energy - Residual\ energy}{Simulation\ time} \quad (10)$$

The average energy consumption of service providers in stage 1 was 20.45 mW, 27.63 mW, and 37.84 mW, respectively, in the three situations. Later, the increase in network load led to increased energy consumption and thus, in the second stage, the consumed energy is 3.98 mW, 18.90 mW, and 11.14 mW, respectively, in situation 1, situation 2, and situation 3.

4.3. Performance Analysis

The performance and efficiency of the proposed method, in this article, have been evaluated by comparing them with [16,17] in terms of computational resources. Execution

timing calculation has been done following [25]. Depending on PBC library [26], the base field in both the stages of the proposed approach are of order 512 bits in length and execution timings are of 150 bits in length. The time needed to execute and perform a multiplication of elliptic curve on G_1 denoted by T_{mul} while T_{par} denotes the execution time for an asymmetric bi-linear pairing operation, just like in [16]. The execution time for creating and authenticating nodes in the network has been analyzed by varying the number of nodes from 100 to 400 in Figure 10 where for each group among four groups having 100, 200, 300, and 400 nodes the minimum, average and maximum time have been recorded.

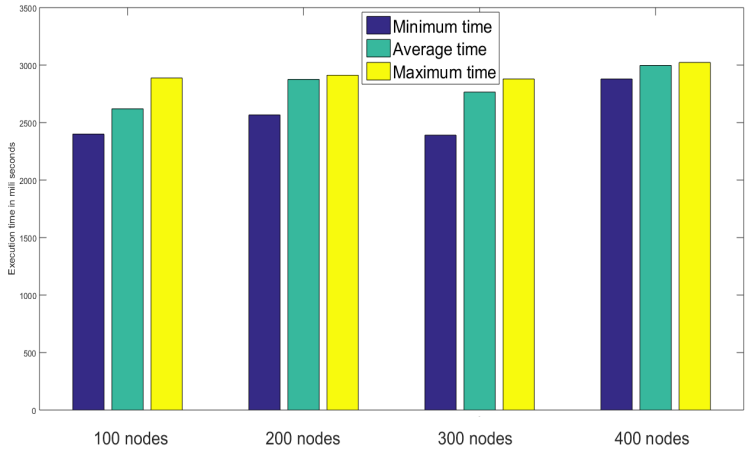

Figure 10. Performance analysis based on node creation and authentication time.

The sensor record reading is a vital task in IoT platforms as it's accuracy defines the whole system's efficiency and effectivity as well. The execution time for recording and storing sensor data has been analyzed in terms of minimum, average, and maximum time like the last scenario in Figure 11, where the graph was steady and the overall capability of the system could be better evaluated if there was no network congestion.

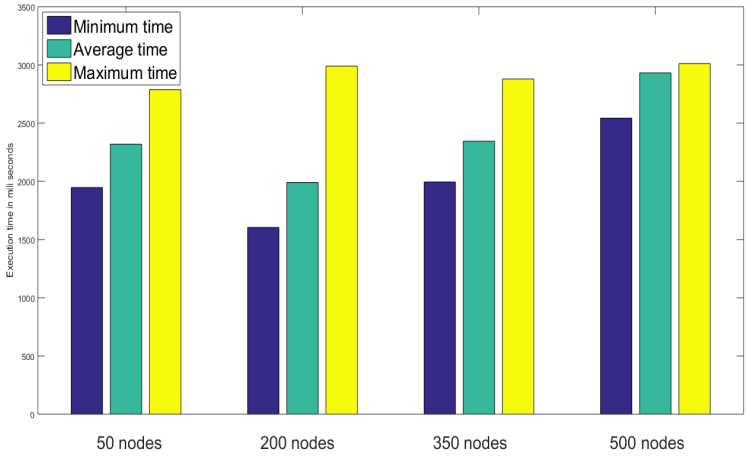

Figure 11. Performance analysis based on sensor record reading time.

During fetching the sensor records from the distributed ledger, the execution time has been analyzed for 500 to 5000 number of queries. It is clear from Figure 12 that number of records has a large impact on the network latency. The worst case was observed for 5000 queries as it took 541, 556, and 567 ms as a minimum, average, and minimum time to execute the operation.

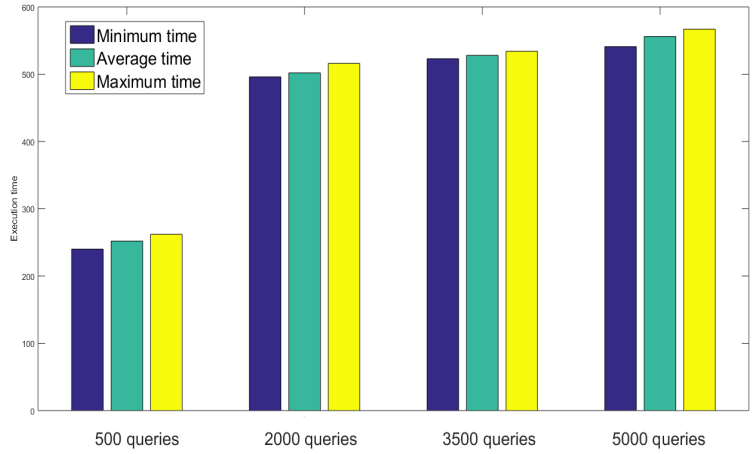

Figure 12. Performance analysis based on sensor query execution time.

The processing overhead of this proposed approach has been compared with [1,4] to evaluate the performance of BlockEdge, shown in Figure 13; where the blockchain network consisting of 60 blocks has been simulated for 70 s and 1030 transactions have been performed. This proposed approach has reduced the processing time overall by 34% which is really a novel contribution in the related domain.

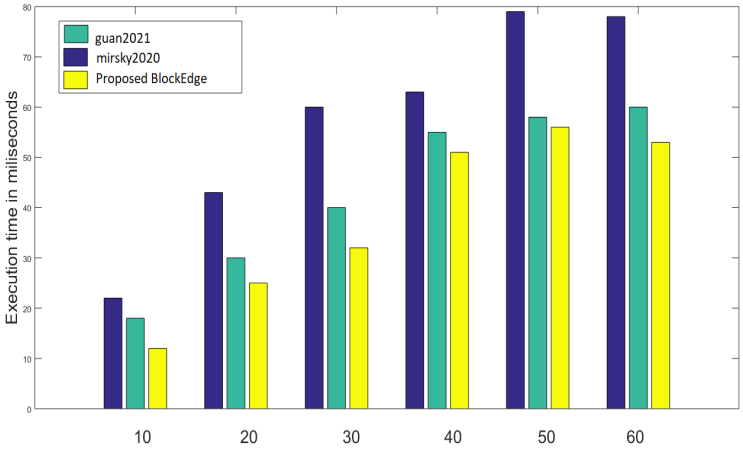

Figure 13. Performance analysis based on processing overhead [1,4].

Light computations: Bitwise XOR operation and cryptographic hash function have been omitted to deliver a comparison on equivalent parameters. The calculation of computational cost in the proposed approach for the mobile user as well as the service provider were performed analyzing the iPhone 6s and i5–4200M cost, respectively, as seen in Table 4. The computational cost of user and service provider in the proposed approach is denoted by $4T_{mul} \approx 27.3$ ms and $4T_{mul} + 2T_{par} \approx 4.8$ ms, respectively. The computational costs in Odelu et al.'s approach [16] have been denoted by $6T_{mul} \approx 39.6$ ms and $5T_{mul} + 4T_{par} \approx 7.8$ ms for user and service provider, respectively. The computational cost of Amin et al.'s [17] approach has been denoted by $5T_{mul} \approx 36.8$ ms and $3T_{mul} + 3T_{par} \approx 8.8$ ms for user and service provider, respectively. This proposed method's computational cost has been reduced due to the absence of a TTP unlike [16,17]. Besides that, the approach offered the same features of security incurring a lesser charge for the involved elements to execute the approach successfully as shown in Table 5. Security features of this proposed scheme were also analyzed and compared with a few similar approaches discussed in [16]. The approach in [16] consists of three entities, i.e., mobile user, service provider, and Smart Card Generator (SCG). This proposed approach integrated

blockchain which, in turn, resulted in not involving some trusted third party in the scheme. Mobile users and service providers generated their public and secret keys.

Table 4. Computational terms in cryptographic operation in two frameworks.

	iPhone 6S	**i5-4200M**
T_{mul}	8.6 ms	0.9 ms
T_{par}	54.5 ms	2.1 ms

Table 5. Comparison in terms of computing cost.

Approach	**Proposed in [17]**	**Proposed in [16]**	**Proposed in This Article**
User	$5T_{mul} \approx 36.8$ ms	$6T_{mul} \approx 39.6$ ms	$4T_{mul} \approx 27.3$ ms
Service Provider	$3T_{mul} + 3T_{par} \approx 8.8$ ms	$5T_{mul} + 4T_{par} \approx 7.8$ ms	$4T_{mul} + 2T_{par} \approx 4.8$ ms

5. Conclusions and Future Scope

The traditional edge platform authorization principles face a lot of challenges [27,28] including a single point of failure, scalability, etc. The introduction of blockchain in this framework reduces these issues for its decentralized and distributed nature, but it still suffers from some new structural challenges because of the various design principles [29]. In this paper , the challenges have been identified and discussed at first, and then, a novel scheme has been proposed incorporating the benefits from both operating principles. A decentralized design has been proposed for better privacy and security to be applied in different edge computing platforms for Industrial IoT and Industry 4.0/5.0. The robustness and cryptographic security have been tested using the AVISPA simulation tool for replay attacks and a man-in-the-middle attack scenario. Cost relative to computational resources has been used as the efficiency measuring tool of this proposed scheme in a practical simulation also, which, in turn, has proved the superior performance of the scheme compared with the existing ones in the concerned domain. The proposed method is well-suited for any user and serves its purpose efficiently. The system's applicability can be tested for some other attack scenarios, such as brute force and Coppersmith's attack in the future.

Funding: This research was funded by Vellore Institute of Technology, Vellore.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: I am thankful to Vellore Institute of Technology, Vellore, for supporting me throughout the completion of this research work.

Conflicts of Interest: The author declares no conflict of interest.

References

1. Guan, Z.; Lu, X.; Yang, W.; Wu, L.; Wang, N.; Zhang, Z. Achieving efficient and Privacy-preserving energy trading based on blockchain and ABE in smart grid. *J. Parallel Distrib. Comput.* **2018**, *147*, 34–45. [CrossRef]
2. Zhang, J.; Wang, Z.; Shang, L.; Lu, D.; Ma, J. BTNC: A blockchain based trusted network connection protocol in IoT. *J. Parallel Distrib. Comput.* **2020**, *143*, 1–16. [CrossRef]
3. Huang, B.; Zhang, R.; Lu, Z.; Zhang, Y.; Wu, J.; Zhan, L.; Hung, P. BPS: A reliable and efficient pub/sub communication model with blockchain-enhanced paradigm in multi-tenant edge cloud. *J. Parallel Distrib. Comput.* **2020**, *143*, 167–178. [CrossRef]
4. Mirsky, Y.; Golomb, T.; Elovici, Y. Lightweight collaborative anomaly detection for the IoT using blockchain. *J. Parallel Distrib. Comput.* **2020**, *145*, 75–97. [CrossRef]
5. Odelu, V.; Das, A.K.; Kumari, S.; Huang, X.; Wazid, M. Provably secure authenticated key agreement scheme for distributed mobile cloud computing services. *Future Gener. Comput. Syst.* **2017**, *68*, 74–88. [CrossRef]

6. Amin, R.; Islam, S.H.; Biswas, G.P.; Giri, D.; Khan, M.K.; Kumar, N. A more secure and privacy-aware anonymous user authentication scheme for distributed mobile cloud computing environments. *Secur. Commun. Netw.* **2016**, *9*, 4650–4666. [CrossRef]
7. Zhang, J.; Chen, B.; Zhao, Y.; Cheng, X.; Hu, F. Data security and privacy-preserving in edge computing paradigm: Survey and open issues. *IEEE Access* **2018**, *6*, 18209–18237. [CrossRef]
8. Maroufi, M.; Abdolee, R.; Tazekand, B.M. On the convergence of blockchain and Internet of Things (IoT) technologies. *arXiv* **2019**, arXiv:1904.01936.
9. Kumari, S.; Khan, M.K.; Atiquzzaman, M. User authentication schemes for wireless sensor networks: A review. *JAD Hoc. Netw.* **2015**, *27*, 159–194. [CrossRef]
10. Reyna, A.; Martín, C.; Chen, J.; Soler, E.; Díaz, M. On blockchain and its integration with IoT. Challenges and opportunities. *Future Gener. Comput. Syst.* **2018**, *88*, 173–190. [CrossRef]
11. Xiao, Y.; Zhang, N.; Lou, W.; Hou, Y.T. A survey of distributed consensus protocols for blockchain networks. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 1432–1465. [CrossRef]
12. Jo, H.J.; Paik, J.H.; Lee, D.H. Efficient privacy-preserving authentication in wireless mobile networks. *IEEE Trans. Mob. Comput.* **2013**, *13*, 1469–1481. [CrossRef]
13. Flores, H.; Srirama, S.N.; Paniagua, C. Towards mobile cloud applications: Offloading Resource-Intensive Tasks To Hybrid Clouds. *Int. J. Pervasive Comput. Commun.* **2012**, *8*, 344–367. [CrossRef]
14. Shahryari, S.; Tashtarian, F.; Hosseini-Seno, S.-A. CoPaM: Cost-aware VM Placement and Migration for Mobile services in Multi-Cloudlet environment: An SDN-based approach. *Comput. Commun.* **2022**, *191*, 257–273. [CrossRef]
15. Roy, D.G.; Das, P.; De, D.; Buyya, R. QoS-aware secure transaction framework for internet of things using blockchain mechanism. *J. Netw. Comput. Appl.* **2019**, *144*, 59–78.
16. Alakberov, R.G. Clustering Method of Mobile Cloud Computing According to Technical Characteristics of Cloudlets. *Int. J. Comput. Netw. Inf. Secur.* **2022**, *14*, 75–87. [CrossRef]
17. Roy, D.G.; Mahato, B.; Ghosh, A.; De, D. Service aware resource management into cloudlets for data offloading towards IoT. *Microsyst. Technol.* **2019**, *28*, 517–531.
18. Vaquero, L.M.; Rodero-Merino, L. Finding your way in the fog: Towards a comprehensive definition of fog computing. *ACM Sigcomm Comput. Commun. Rev.* **2014**, *44*, 27–32. [CrossRef]
19. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S.L. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 17 August 2012; pp. 13–16.
20. Beck, M.T.; Maier, M.L. Mobile edge computing: Challenges for future virtual network embedding algorithms. In Proceedings of the Eighth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP 2014), Rome, Italy, 24–28 August 2014; 2014; p. 3.
21. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. Mobile edge computing—A key technology towards 5G. *Etsi White Pap.* **2015**, *11*, 1–16.
22. Bonnah, E.; Shiguang, J. DecChain: A decentralized security approach in Edge Computing based on Blockchain. *Future Gener. Comput. Syst.* **2020**, *113*, 363–379. [CrossRef]
23. Ray, P.P. An introduction to dew computing: Definition, concept and implications. *IEEE Access* **2017**, *6*, 723–737. [CrossRef]
24. Bhargav-Spantzel, A.; Squicciarini, A.C.; Modi, S.; Young, M.; Bertino, E.; Elliott, S.J. Privacy preserving multi-factor authentication with biometrics. *J. Comput. Secur.* **2007**, *15*, 529–560. [CrossRef]
25. Lynn, B. *PBC Library Manual 0.5. 11*; Stanford University: Stanford, CA, USA, 2006.
26. Hang, L.; Kim, D.H. Design and implementation of an integrated iot blockchain platform for sensing data integrity. *Sensors Multidiscip. Digit. Publ. Inst.* **2019**, *19*, 2228. [CrossRef] [PubMed]
27. Dorri, A.; Kanhere, S.S.; Jurdak, R. Towards an optimized blockchain for iot. In Proceedings of the 2017 IEEE/ACM Second International Conference on Internet-of-Things Design and Implementation (IoTDI), Pittsburgh, PA, USA, 18–21 April 2007; pp. 173–178.
28. Chang, C.; Srirama, S.N.; Buyya, R. Indie fog: An efficient fog-computing infrastructure for the internet of things. *Comput. IEEE* **2017**, *9*, 92–98. [CrossRef]
29. Luo, E.; Bhuiyan, M.Z.A.; Wang, G.; Rahman, M.A.; Wu, J.; Atiquzzaman, M. Privacyprotector: Privacy-protected patient data collection in IoT-based healthcare systems. *IEEE Commun. Mag.* **2018**, *56*, 163–168. [CrossRef]