electronics

MDPI

Article

A Lightweight Blockchain Based Framework for Underwater IoT

Md Ashraf Uddin, Andrew Stranieri, Iqbal Gondal and Venki Balasurbramanian

Internet Commerce Security Laboratory, Federation University Australia, Mount Helen VIC 3350, Australia
a.stranieri@federation.edu.au (A.S.); iqbal.gondal@federation.edu.au (I.G.);
v.balasubramanian@federation.edu.au (V.B.)
* Correspondence: mdashrafuddin@students.federation.edu.au

Received: 18 November 2019; Accepted: 12 December 2019; Published: 16 December 2019

Abstract: The Internet of Things (IoT) has facilitated services without human intervention for a wide range of applications, including underwater monitoring, where sensors are located at various depths, and data must be transmitted to surface base stations for storage and processing. Ensuring that data transmitted across hierarchical sensor networks are kept secure and private without high computational cost remains a challenge. In this paper, we propose a multilevel sensor monitoring architecture. Our proposal includes a layer-based architecture consisting of Fog and Cloud elements to process and store and process the Internet of Underwater Things (IoUT) data securely with customized Blockchain technology. The secure routing of IoUT data through the hierarchical topology ensures the legitimacy of data sources. A security and performance analysis was performed to show that the architecture can collect data from IoUT devices in the monitoring region efficiently and securely.

Keywords: Internet of Things; Blockchain; hierarchical topology; underwater IoT monitoring; Fog layer; Cloud; IoUT routing; consensus mechanism; indexing

1. Introduction

Internet of Underwater Things (IoUT) sensor networks [1] enable agencies to monitor underwater spaces to explore resources including gas and gold; measure water temperature; observe fish and oil or gas pipelines; and convey information pertaining to a tsunami, water contamination or other natural disasters [2].

IoUT typically transmit data as sound variations [3]. However, acoustic signals (1500 ms^{-1}) have long propagation delays [3] and high bit error rates. Consequently, underwater communication channels result in low-quality transmission of data. In addition, IoUT devices are deployed in hostile environments and remain unattended, making underwater communication vulnerable to various malicious attacks [4]. Other challenges arising from underwater deployment include dealing with variable water currents, batteries that are difficult to recharge or replace, limited memory, and low bandwidth of devices. These factors in IoUT networks result in an extra barrier to develop a secured routing protocol to collect data from underwater IoT sensors [5].

Various routing architectures and protocols including hierarchical (vertical) [6,7] flat (horizontal) [8], location based [9], multipath routing [10], query based [11], and context aware [12] have been deployed in underwater IoT networks to transmit sensed data to a base station on the surface [13].

The network topology in hierarchical routing is divided into several layers resulting in a compact routing table enabling scalability required for large scale underwater IoT monitoring. Typically, a node with higher energy is nominated as a cluster head (CH) and other nodes with lower energy sense data. The CH's role is to aggregate data and forward the data to the next level. The CH is changed over time due to battery depletion caused by the computational load of aggregating and transmitting data. Further, privacy within a cluster is at risk if the cluster head is compromised by malicious attacks. Therefore, the cluster head needs a mechanism to protect and maintain member nodes' security and privacy.

A standard protocol in sensor networks known as proactive routing follows a static route but is unsuitable for IoUT devices that change their location with ocean currents [14]. In contrast, reactive routing finds a path on demand by flooding the network with route request messages. Cluster-based reactive routing can be contemplated for large scale mobile underwater sensor networks to address the scalability issue but has high power consumption due to the need to infer the path before forwarding data.

To address IoUT routing issues, we present a lightweight, reactive protocol for hierarchical IoUT networks that require fewer control messages to infer routing the packet forwarding path than other reactive protocols. In this routing mechanism, the cluster head uses a Bloom filter to preserve the privacy of the member node. A Bloom filter contains multiple pseudo-identifiers generated from a node's real identifier using different mathematical hash operations. As a result, the real identity stored in the Bloom filter is hidden from malicious nodes. In a routing protocol, control packets play significant roles in ensuring a node's security and privacy. Nodes from the same manufacturers can utilize symmetric key cryptography to save energy while exchanging control packets. Symmetric key encryption uses the same key to encrypt the plaintext and decrypt the ciphertext. Nodes from different manufacturers need a key exchanging algorithm [15] to communicate between them. The cryptographic hash function that maps data of arbitrary size to a fixed size of code is applied to perform authentication between nodes.

However, the next stage, after collecting IoUT data via a lightweight routing mechanism, is to store and process IoUT data securely. For this purpose, most IoUT architectures are comprised of interconnected underwater IoT devices that transmit data through perception, network, and application layers [16] on the surface, to a remote central, often Cloud-based, server to analyze and store data generated by IoUT devices [12,17–19]. However, a centralized IoT architecture can be paralyzed by a Denial of Service (DoS) or Ransomware attack because the central server represents a single point of failure and performance bottleneck. To address this issue, a distributed peer-to-peer wireless sensor network has been suggested to store and process IoUT data [19,37]. However, security and privacy are challenged with a distributed peer-to-peer network while processing and sharing IoUT data.

Recently, Blockchain (BC) leveraged distributed, decentralized peer-to-peer networks [8] have emerged to enable underwater IoT data to be stored securely and inexpensively without relying on any intermediary trusted authorities [28] while ensuring privacy. Entities in the Blockchain network participate in processing and validating IoUT data prior to confirming the inclusion of IoUT data to the Blockchain. This process, called a consensus mechanism, substitutes the needs of third-party involvement to process IoUT data. The Consensus mechanism in the Blockchain prevents fraudulent activities and ensure data immutability, transparency, and operational resilience of the Blockchain ledger. Blockchain leverages public key infrastructure (PKI) to authenticate, authorize entities, and encrypt records in peer-to-peer networks. Two entities on a Blockchain network can independently communicate without the aid of intermediaries. The basic data unit of the Blockchain is called a transaction, and several transactions are organized into a Block. Every timestamped Block's header contains a hash code of its immediate, previously confirmed Block. This creates a sequential linkage between data Blocks on the Blockchain, which confirms the irreversibility and ensures that data is tamper-proof.

From the above point of view, Blockchains can facilitate decentralized storage for recording IoUT data and secure processing and sharing IoUT data with different entities. Blockchain technologies have been promoted in IoUT applications that require decentralized access control, storage, and distributed trust [19]. For instance, decentralized Blockchain-based IoUT architecture can be applied for the storage of data generated by many heterogeneous underwater IoT devices deployed at different places, and authenticating software update confirmation for a wide range of marine IoT devices. Further, the consumer or customer often needs to purchase different services such as software update, antivirus, anomaly detection software from a single third party in an IoUT network. In these kinds of applications, the customer needs to trust a single party's solution blindly but the QoS (quality of service) for customers cannot always be guaranteed with a centralized IoUT architecture. The Blockchain technology can be adopted to enhance QoS for these kinds of IoUT applications. However, current Blockchain technology is not computationally efficient for handling IoUT Big data. Multiple nodes in the Blockchain contain a replica of the complete ledger. This distributed feature of the Blockchain demands high storage, and the consensus mechanism, which is the core component of the Blockchain, cannot process transactions faster than a traditional centralized IoUT architecture [19]. The pros and cons of conventional IoUT architecture and Blockchain IoUT architecture are presented in Table 1.

Table 1. The pros and cons of conventional IoUT architecture and Blockchain IoUT.

Parameter	**Conventional IoUT**	**Blockchain IoUT**
CIA = Confidentiality, Integrity, Availability	Potential risk of unauthorized access to IoUT data. Risk of hidden data alternation or modification by third party [20]. Interruption of service due to outrages [21,22]. Poor fault tolerance and occurring of bottleneck to handle many service requests.	Preservation of confidentiality because of user-driven record management. Facilitating cross-sharing records ensuring data integrity. Data integrity can be ensured by the replication of the ledger amongst multiple entities replacing third parties' involvement while processing data. Multiple entities can access a record in the Blockchain enabling a system to respond to many service requests
Privacy	Exposing user's identifier to Cloud administrator.	Storage of IoUT record anonymously. Privacy is violated if correlating data to the source [23] is successful.
Freshness	Manipulation of timestamped record by the host. Risk of updating timestamp	Assurance of data freshness using global timestamp.
Cyber Attack	Vulnerable to DoS, ransom, spoofing, and single point of failure due to centralized architecture. Risks of leakage of personal information during data dissemination [24]	Withstand against DoS, ransomware, and single point of failure. Susceptible to long-range attack, nothing at stake, dropping, eclipse, mining attack, and 51% attack.
Cost	High deployment, maintenance, and administrative costs [25]. The integration of fragmented records is expensive	The public can contribute resources. Alleviation of many service costs, including employee wages, legal expenses, and data center rentals.
Interoperability	Poor interoperability due to diverse legal requirements and security methods followed by a different institution. The extra barrier to the cross-border IoUT data sharing [26].	High interoperability because of a universal set of rules and regulations followed by every entity. Support cross-broader IoUT data sharing
Standardization	Standardization already established [27]	Lack of high level standardization [20]

In this article, we merge a customized lightweight Blockchain with an underwater IoT network to provide interested stakeholders with various secure marine services, including secure storage, sharing, and trading platform for IoUT monitoring data. The architecture advanced here implements the Blockchain at the Cloud to accommodate the high computing and storage required for this technology.

The framework facilitates aggregating continuous data, indexing, and handling vast amounts of IoUT data while maintaining privacy and security. We devised a lightweight consensus mechanism that is utilized to validate the processing and analysis of IoUT data. A smart Gateway Agent is envisaged to receive IoUT data from underwater monitoring sensors and insert them into Blockchain.

Our contribution includes the following design elements:

1. The hierarchical IoUT monitoring topology consists of underwater IoT devices at different levels, Gateway Agent in a Fog layer, and Blockchain in the Cloud layer.
2. The secure and lightweight routing protocol transfers the underwater IoT data to a lightweight Blockchain in the Cloud.
3. A lightweight consensus mechanism is proposed to process transactions in the Blockchain. The consensus mechanism selects a group of Miner nodes using the TOPSIS multi-criteria analysis method. Two use cases called IoUT Data Block Inclusion and IoUT Outcome Block Inclusion are investigated using the consensus mechanism.

The related research is described in Section 2 before advancing solutions in Section 3. The performance of the proposed approach is discussed in Section 4 before the concluding remark in Section 5.

2. Related Work

In recent IoT-based applications, IoT devices collect and forward data to a coordinator node. The coordinator node transmits data to a remote, often Cloud-based server. Some recent Blockchain-based IoT monitoring [29] systems consider a single level of IoT devices. However, some applications such as disaster monitoring, water/air pollution monitoring, and wildlife monitoring require that IoT devices are placed at different levels. In such a topology, an adversary can eavesdrop on the communication and can overhear critical information if secure routing among IoT devices is not ensured. Similar to ground-based IoT, IoUT devices are vulnerable to hole attacks, reconnaissance, Sybil, spoofing, eavesdropping, neighbor discovery, man-in-the-middle, rogue devices, and fragmentation attacks due to large scale, sparse, and unattended networks [4]. In this section, we present a ground-based IoT framework before presenting a comparative analysis of the underwater IoT framework.

2.1. Blockchain Based Ground IoT Framework

Some efforts to integrate IoT and Blockchain have been made in recent years. Ali [29] proposed an architecture to incorporate Blockchain and IoT for monitoring smart homes. The IoT data from the smart home is stored, indexed, and shared among consumers using Blockchain. IoT devices have limited processing power, and memory constraints so computationally expensive cryptographic techniques are not appropriate. IoT data can plausibly be generated faster than any consensus protocol can confirm the Block in the Blockchain network [30]. Therefore, Ali replaced the proof of work by a distributed trusted consensus method. However, Ali did not consider securing the routing for the hierarchical topology of IoT applications.

Zyskind [31] proposed a Blockchain-based user protocol for the installation of Apps. In that protocol, the user can set policies and terms, unlike traditional App installation, where the App forces users to agree to some terms and conditions. The same procedures can be applied to use IoUT data from the Blockchain. Lei [32] suggested a Blockchain-based infrastructure for exchanging keys in a vehicular communication system. The Blockchain's transactions are used to transfer critical information containing a key among heterogeneous infrastructure in a new region whenever the vehicle moves there. Lei's proposal might help to exchange keys in IoT devices in a vehicular network. Tian [33] presented an agricultural food supply chain traceability system by integrating IoT devices such as RFID with Blockchain technology. Tian identified benefits in using RFID and Blockchain technology in a food supply chain to ensure transparency and availability of data generated along the supply chain. Samaniego [34] incorporated Cloud services as a third layer with traditional IoT monitoring to store data streamed from IoT devices.

Recent proposals [35,36] integrated Fog and Cloud layer to process IoT data. Aazam [35] proposed a smart gateway to integrate IoT with Cloud. The smart gateway can coordinate, pre-process, and manage encryption keys before putting data into the Cloud. Sharma [36] proposed a layer-based architecture for capturing data from IoT devices. The architecture includes IoT devices, a Software Defined Network (SDN) controller in the Fog layer at the edge of the network and a distributed Blockchain in Cloud. The SDN controllers in the Fog layer form a distributed network like Blockchain. The Cloud service providers also constitute a distributed Blockchain. Proof of services that combines the proof of stake and proof of work is proposed as a consensus protocol. We extended the architecture with hierarchical IoT monitoring to efficiently store IoT stream data in distributed Blockchains in the Cloud.

Our architecture differs from these approaches; we designed our architecture for monitoring the underwater environment and presented two use cases of a modified Proof of Stake (PoS) consensus mechanism. We include an energy-efficient cluster-based routing protocol where the node's level is used to forward the data packet to the destination. The cluster head maintains its member nodes' privacy using a Bloom filter. Uddin [37] presented Blockchain-based IoT smart monitoring architecture that includes a network manager to manage keys for the IoT devices. A software agent executing on the Gateway Agent selects a group of Miners by considering their performance to run the Proof of Work. However, routing mechanisms for IoT devices were not discussed in [37].

2.2. Cluster Based Routing in Ground IoT

Different routing mechanisms have also been designed for IoT devices [38–40]. Tian [38] proposed an ad-hoc on-demand multipath distance vector routing protocol for IoT. The protocol maintained an internet connecting table (ICT) to discover new neighbors and a routing table for every node. However, the exchange of many control packets is required to discover neighboring nodes which consume much energy and should be avoided for lossy and power-constrained IoT devices.

Chze and Leung [39] advanced a secure, multihop routing protocol where a legitimate service provider inserted encrypted registration information including owner applications, and the network address into the IoT devices before forming a routing mechanism using a "Hello" control message. However, the reliance on a single service provider for registration causes bottleneck problems. The adoption of Blockchain instead of the specific service provider can improve the secure routing mechanism. However, that protocol is not scalable because IoT devices require large storage with a growing number of IoT devices in the network. Iova [40] proposed a destination-oriented directed acyclic graph using some objective functions such as link stability and lifetime. However, the protocol does not support multipath routing, and selections of the neighbor node based on objective functions require the exchanging of a control message.

LEACH [41] and HEED [42] are two fundamental cluster-based routing protocols, and their variations have recently been proposed in wireless sensor network (WSN). LEACH [41] routing for wireless sensors changes the cluster head after elapsing a designated period of time. LEACH follows a randomized rotation to select a cluster head. The randomization technique creates an opportunity for every node to become a cluster head after a certain period. HEED [42] encompasses the original structure of LEACH utilizing residual energy and node degree or density as a metric for cluster selection to attain power balancing. In HEED, the cluster head is intermittently nominated by two parameters. The probability of a sensor node to become a cluster head is calculated on the first parameter called residual energy, and the second parameter is a function of cluster density, or node degree is used to assess the inter-cluster communication cost. A hierarchical routing mechanism can lead to efficient and scalable IoT monitoring. However, the formation of a cluster head in hierarchical routing consumes a great deal of energy for memory-constrained IoT devices. Therefore, a lightweight cluster formation technique is needed to facilitate scalability.

2.3. Underwater Sensor Networks (UWSN)

Multi-hop-based routing protocols that exchange control messages to discover forwarding path [43–47] have been developed for the transmission of underwater data to the surface. However, these protocols drain more power of nodes near the sink.

The depth-based store and forward routing method are considered the most efficient protocol for UWSN. DBR [14] is the first designed depth-based underwater routing protocol that guides packet to the destination solely depending on the water depth of a node. The forwarding node waits for a specified period, which is proportional to its depth to avoid the involvement of multiple nodes to forward the same packet. Later, many depth-based power-efficient protocols [3,5,48,49] have been proposed to address the challenges of the underwater sensor network (UWSN). However, extra battery power is required to estimate the depth of a node in such routing methods. The protocols, as mentioned above, are the subsystems of an underwater monitoring architecture. In this article, we design an end-to-end secure underwater monitoring architecture, including a lightweight level-based routing protocol for UWSN. The Gateway in the Fog layer bridges Blockchain network with the power and memory limited IoUT devices. The Gateway Agent nominates a small set of Miners for executing Proof of Stake consensus mechanism to store and process IoUT data in the Cloud Blockchain. The Cloud can facilitate decentralized and distributed storage, and processing which can tackle privacy and security issues existed in current peer-to-peer distributed underwater wireless sensor networks. A comparative analysis of other existing underwater IoT monitoring frameworks and the proposed IoUT framework is presented in Table 2.

Table 2. The comparative analysis of IoUT architecture.

Comparison Criteria	TDSUAC [12]-Secure underwater network adopting security at physical layer, and combining software-defined cognitive and context-aware networks	ALSN [17]-AUV (Automatic Underwater Vehicle) and sensors constitutes networks to monitor and protect underwater pipelines carrying gas or water	HT-SHE [7]-Smart home monitoring with sensors labeled as high, medium and low capabilities	Blockchain IoT-SHM [8]-Blockchain-based architecture for underground structural health monitoring	Proposed Multi-Level IoUT
Fault tolerance	Medium	Low	Low	The centralized Gateway is vulnerable to cyber attacks, thus overall fault tolerance is medium	Multiple Gateway Agents run Blockchain protocols so high fault tolerance capability is achieved
Confidentiality, Integrity and Availability	Preserve confidentiality but cannot guarantee integrity, and high availability	No security standard is defined	Security is defined for the transmission of data but not for data storage, and availability is reduced	Integrity and availability are guaranteed for metadata in the Blockchain	Security, privacy and availability are ensured in both network and Blockchain data management layer
Cyber Attacks	Vulnerable to Ransomware, and DoS attacks	Susceptible to major cyberattacks	Vulnerable to Ransomware, DoS and man in the middle attack	Vulnerable to Ransomware and DoS as data are stored in a single database	Withstand Ransomware and DoS as multiple Cloud servers store the Blockchain ledger
Data Immutability	No	No	No	Single database is used to store data; therefore, there is a risk of data being tampered by an attacker	Yes
Computational Cost	Medium	low	low	PoW consensus mechanism consumes higher power	Lightweight security protocol for routing and PoS, energy and time-efficient consensus mechanism is applied
Secure Generic Communication	Yes	No	Yes	Routing mechanism is not discussed	Multi-level hierarchical routing
Interoperability	No	No	No	Yes	Yes
Scalability	No	No	Yes	Yes	Routing Protocol is scalable but Blockchain-based storage challenges Big IoUT data
Service Reliability	No	No	No	Smart contracts ensure service reliability	All user's services are verified by the consensus mechanism

2.4. Key Management in IoT

The problem of key management for IoT has also been explored in the current literature. Malina [50] examined the performance and memory requirements for IoT devices such as microcontrollers, smart cards, and mobile devices. The symmetric key and hash function are suitable for IoT devices with limited memory and processing power. Jayaraman [51] proposed a modified OpenIoT architecture that used homomorphic encryption techniques. However, homomorphic encryption that requires bilinear pairing, modular exponential, and point multiplication is not appropriate for IoT devices because of the high computational costs involved [50]. Shafagh [52] presented a proximity authentication technique for IoT devices, but proximity authentication is not applicable for mobile and unattended IoUT devices. Lie [53] proposed wireless monitoring and control systems for the smart grid and smart home using a one-time dynamic password for authentication. Ngo [54] described the advantages and disadvantages of maintaining security in wireless networks by using a dynamic key.

We utilized a secure routing mechanism in our designed topology consisting of IoUT devices by maintaining hybrid key management. The secure routing mechanism is not focused on the Blockchain-based hierarchical structure of IoUT monitoring applications. Existing IoUT monitoring systems with a routing mechanism ignore the secure storage of IoUT data. Therefore, there is an opportunity to design a generic hierarchical IoUT monitoring topology with the inclusion of Blockchain for the safe storage of IoUT data.

3. Hierarchical Architecture for Underwater IoUT Monitoring

We present an end-to-end framework that handles the collection, processing, and storage of IoUT data. The architecture, as illustrated in Figure 1 for monitoring generic IoUT applications, consists of three layers: Internet of Things layer, Fog layer (comprising one or more levels), and Cloud layer. The functional flow diagram of the framework is depicted in Figure 2. The top, middle, and bottom parts of Figure 2 describe the operation of the Internet of Things, Gateway Agent in the Fog, and Blockchain in the Cloud, respectively. The smart Gateway node is deployed in the Fog layer. The Cloud layer contains a distributed Blockchain that consists of varying Cloud service providers. In this section, we describe the hierarchical routing mechanism that forwards IoUT data to the surface nodes. The IoUT devices are deployed at different levels in the underwater environment.

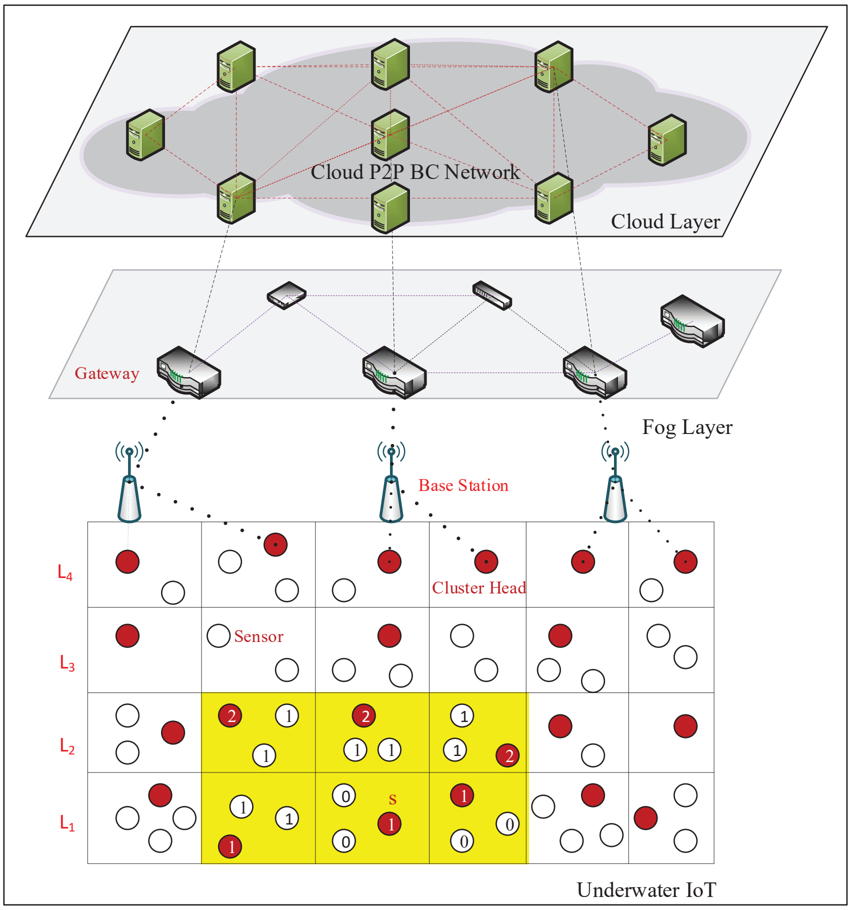


Figure 1. The multilevel architecture for IoUT monitoring.

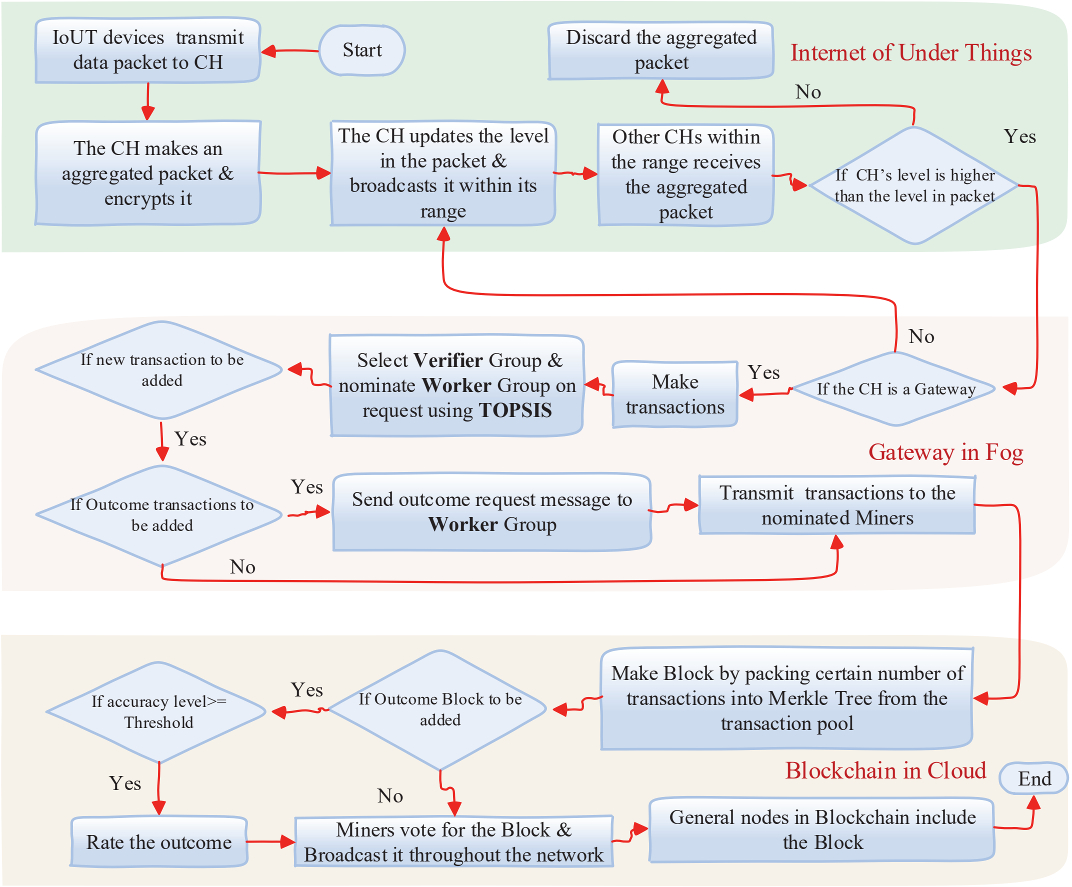

IoUT devices transmit data packet to CH
Start
Discard the aggregated packet
Internet of Under Things
No
Yes
The CH makes an aggregated packet & encrypts it
The CH updates the level in the packet & broadcasts it within its range
Other CHs within the range receives the aggregated packet
If CH's level is higher than the level in packet
No
If new transaction to be added
Select **Verifier** Group & nominate **Worker** Group on request using **TOPSIS**
Make transactions
Yes
If the CH is a Gateway
Yes
Gateway in Fog
Yes
If Outcome transactions to be added
Send outcome request message to **Worker** Group
Transmit transactions to the nominated Miners
No
Make Block by packing certain number of transactions into Merkle Tree from the transaction pool
Yes
If accuracy level>= Threshold
If Outcome Block to be added
Blockchain in Cloud
End
Yes
No
Rate the outcome
Miners vote for the Block & Broadcast it throughout the network
General nodes in Blockchain include the Block

Figure 2. The flow diagram for the framework.

3.1. Internet of Things Layer

In this layer, IoUT devices are organized into different levels of the underwater monitoring region, forming a grid network. Each block of the gird contains a cluster head, and members of that cluster are sensors broadcasting data within a specific range. The IoUT device with higher processing power, memory, and stability is normally nominated as the cluster head. The following assumptions are made to devise the secure routing in this layer:

Cluster head coverage: A cluster head's range can cover its immediate top, bottom, left, and right rectangular area marked in the yellow shaded cells at the lower end of Figure 1. A member node's range is limited to its cluster. The monitoring region is vertically divided into different levels (e.g., the level close to the ground (or base) is 1, the next one is 2, and so on). The relative position of each IoUT device (represented as a dark shaded circle in Figure 1) is determined by the level number (L) of its associated cluster head (represented by a red shaded circle in Figure 1). The cluster head's level is higher than its member. If the cluster head level is L, the member nodes' level is $L-1$ (for instance, if cluster head's level is 4, every member node's level under this cluster head is 3).

Key management: In this protocol, every node uses a common symmetric key, and an asymmetric key (public and private key pairs) for the communication. Every IoUT device obtains the common symmetric key (csk) and the asymmetric key (private key ($privateKey_m$) and public key ($publicKey_m$)) inserted into the node's hardware by their manufacturers. A manufacturer inserts a similar symmetric key and asymmetric key for all its IoUT devices. The public key of each IoUT device is stored on the Cloud Blockchain. The IoUT devices from various manufacturers need to register to the smart Gateway Agent placed in the Fog layer to join in the underwater monitoring. The Gateway Agent verifies the registered IoUT devices by retrieving their public keys from the Blockchain. Later, the smart Gateway Agent regularly obtains the updated keys from the Blockchain and distributes those keys among IoUT nodes of the monitoring region. Here, every node's security key can be periodically updated through Blockchain, according to Lee [55].

Nodes' privacy: Cluster head and member IoUT devices are mobile and can change their levels. Nodes do not utilize their real identifier for performing communications with other nodes. Cluster head and member nodes use a shadow identity (*id*) with their level to preserve privacy. The cluster head uses a Bloom filter to keep track of its member IoUT devices. A Bloom filter is a data structure designed to locate or identify the presence of an element in a set rapidly and memory-efficiently. Several hash functions such as MurmurHash3 [56], cityhash [57], or fnvhash can be used to index the hash function in the Bloom filter. For example, the index function for a Bloom filter can be defined as $H(id)\%m$ where *id* is the shadow identifier of an IoUT device, $H()$ is either MurmurHash3 or cityhash function, and m is the size of the Bloom filter.

$$H_1(id) = id\%m$$

$$H_2(id) = (id + 1)\%m$$

A Bloom filter is depicted in Figure 3. Suppose the cluster head has two member nodes (identifier 10 and 20 ,respectively) to be recorded in its Bloom filter. The Fnvhash and MurmurHash of node *id* 10 and 20 are 13 and 14, and 10 and 9, respectively. For node identifier 10, indices 13 and 14 contain 1 (color-marked) and, for node identifier 20, indices 9 and 10 hold 1 (color-marked). The cluster head generates an index from a node *id* using FnvHash and MurmurHash. If the respective indexes hold 1 in the Bloom filter, the node is a member of that cluster. The benefit of using the Bloom filter is that even If an attacker compromises a cluster head, the identity of the member nodes is not disclosed to the attacker.

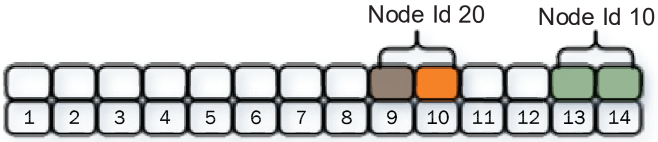


Figure 3. The Bloom filter for the cluster head.

3.1.1. Data Forwarding Phase

The first step of forwarding data from the source node to destination involves a node's participation in a cluster. A new node needs to send a request control message to a cluster head to join the cluster. The request control message contains the manufacturer's code, residual energy (*re*), and signature generated by the private key of the node. If the cluster head and the potential member node are produced from the same manufacturer, the cluster head sends an encrypted reply message using the common symmetric key (*csk*). Otherwise, the cluster head asks the Gateway to provide it with the public key of the member node's manufacturer. The cluster head verifies the member node's signature using this public key. The control message containing a reply holds a data encryption key, identifier(*id*), and level number (l) for the member node for further communication with the cluster head. This process is illustrated in flow diagram depicted in Figure 4.

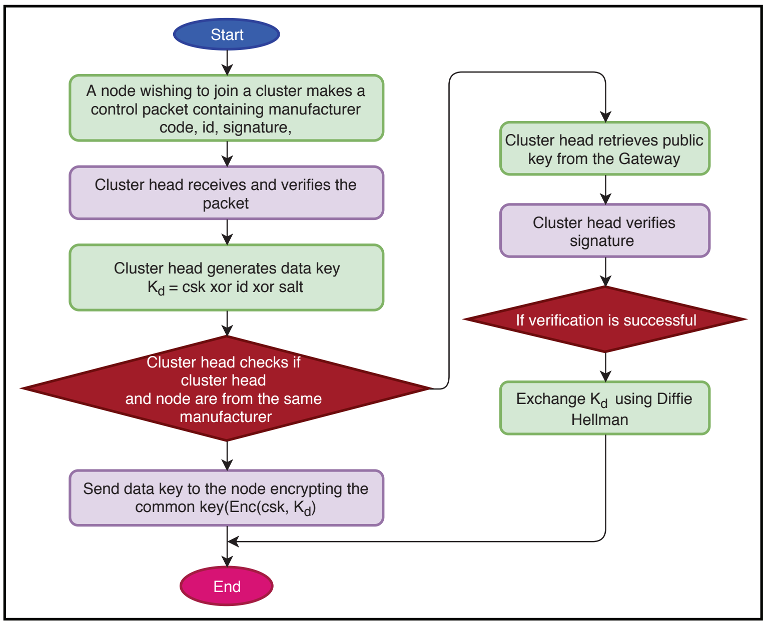

Start
A node wishing to join a cluster makes a control packet containing manufacturer code, id, signature,
Cluster head receives and verifies the packet
Cluster head generates data key K_d = csk xor id xor salt
Cluster head checks if cluster head and node are from the same manufacturer
Send data key to the node encrypting the common key(Enc(csk, K_d)
End
Cluster head retrieves public key from the Gateway
Cluster head verifies signature
If verification is successful
Exchange K_d using Diffie Hellman

Figure 4. The key management process

The cluster head generates data encryption key as follows: $K_d \leftarrow id \oplus csk \oplus salt$ If the cluster head and the member node are from the same manufacturers, K_d is encrypted using the common symmetric key $Enc(csk, K_d)$.

If the cluster head and the member node are from the different manufacturers, they need to use the Diffie-Hellman Key Exchange algorithm to exchange data encryption keys. Similarly, the cluster head gets a data encryption key from the associated smart Gateway Agent for a certain period of communication.

The packet forwarding method is described below. A node transmits its sensed data to the associated cluster header. The format of the data packet is illustrated in Table 3. The data packet has two parts, namely the header containing the timestamp (t), node's level number (l), identifier(id), residual energy (re), and $HMAC(K_d, t\|l\|id\|re\|H(ciphertext))$ as a signature; and body containing $ciphertext \leftarrow Enc(K_d, data)$. The cluster head produces $HMAC(K_d, t\|l\|id\|H(ciphertext))$ to verify the packet's header information such as level (l), identifier, and data contents. If the verification process is successful, the cluster head includes the data into its forwarding packet pool. The cluster head forms a big data packet by taking a certain amount of data from the forwarding packet pool. The cluster head encrypts data using a data encryption key (K_g) given by the Gateway Agent to which the cluster head is associated. The cluster head broadcasts this aggregated data packet throughout its range (marked by yellow rectangular shape in Figure 1). The member nodes within the range of this cluster head also receive the aggregated data packet. However, only the cluster heads with a level higher than that of the sending cluster head's level forward the aggregated data packet. Other nodes whose level is equal or lower than the sending node's level discard the packet. The new forwarding cluster head updates the level field by inserting its level before broadcasting the packet further throughout its range. In this forwarding process, only cluster heads (more than one cluster heads at the same level) participate in forwarding a data packet to the destination without the need to exchange any control messages between the sending and forwarding cluster head.

For example, the level of the cluster head at ground level is 1; the level number of its member nodes is 0. The cluster head of its immediate upper level is 2, and the member nodes' level of this cluster head is 1, which is equal to the level of the immediate lower level cluster head. Now, a cluster head at ground level broadcasts packets throughout the range. The cluster heads that exist at level 2 and to the right and left of the sender cluster head will receive the aggregated packet. The cluster head at level 2 will forward the packet as its level is greater than the sender cluster head (which is at ground level). Other cluster heads and normal nodes discard the packet as their level does not permit sending the data packet. All the cluster heads at level 2 wait for a period that is inversely proportionate to their energy after receiving the aggregated packets ($T = \frac{1}{re} \pm \delta$, δ is random number needed to make each node's waiting time different in case some of them have the same residual energy). The most competent cluster heads at level 2 forward the packets. Other cluster heads overhear the same aggregated packet and discard their copy. The forwarding cluster head will send an ACK control message to the sender cluster head. If the sender cluster head does not receive an ACK within a certain period, the sender will reduce the level of the packet and broadcast the packet again. As a result, other nodes might participate in forwarding the packet. This data gathering method and aggregated data packet forwarding are illustrated in Algorithms 1 and 2. Level-based routing mechanism ensures the involvement of small numbers of IoUT devices in forwarding a data packet without exchanging control messages. However, few control messages are periodically required to exchange security keys among the IoUT devices, the Gateway Agent, and Blockchain nodes.

Algorithm 1: Data Gathering by Cluster Head.

```
Data: data packet ($dp$), sender node identifier ($id$) the level ($l$) in data packet, cluster head level ($L_i$)
Result: Data gathering by each cluster head
initialize $dp = false$
while $dp$ is $true$ do
    if $L_i > l \wedge HMAC(K_d, t \| l \| id \| re \| ciphertext) = Tag$ then
        if $id$ is in Bloom Filter then
            Store the data packet into forwarding packet pool
        else
            discard the packet
        end
    else
        discard the packet
    end
end
```

Algorithm 2: Data Forwarding Algorithm.

```
Data: sender cluster head identifier (id), the latest level (l)in data packet, forwarder cluster head node's
      level $L_j$
Result: forwarding data packet or reject data packet
initialize $forwarding = false$
while $forwarding = true$ do
    Form aggregated data packet from forwarding packet pool and perform $Enc(K_g, data)$
    Insert cluster head's level into the packet
    Broadcast the data packet throughout cluster head's range
    for all nodes $j = 1$ to $m$ within the cluster head $i$ range do
        if $L_j > l$ then
            update the level of the packet
            Estimate $T = \frac{1}{re} \pm \delta$
            if overhear the same packet during $T$ then
                Discard the packet
            else
                Broadcast the packet
            end
        else
            Discard the packet
        end
    end
end
```

Table 3. The data packet format.

Sequence No	Timestamp	Level	Identifier	Residual Energy
$Tag \leftarrow HMAC(K_d, t\|l\|id\|re\|H(ciphertext))$				
$Enc(K_d, Data)$				

3.1.2. Cluster Head Selection

The member nodes under a cluster head include their residual energy every time they send a data packet to the cluster head. The cluster head needs to maintain a database to keep records for its members' residual energy and periodically updates its database. If the current cluster head's residual energy is below the threshold energy or if it leaves the current cluster, the current cluster head selects a member node as a new cluster head considering residual energy and link stability. The current cluster head informs the Gateway Agent about the next cluster head. The current cluster head shares the Bloom filter containing the list of the member nodes with the newly nominated cluster head (the next cluster head), and the newly nominated cluster head increases its level by one after receiving approval from the Gateway Agent.

3.2. The Gateway Agent on the Fog Layer

IoUT devices cannot support high processing and memory required for the Blockchain. Further, IoUT devices cannot be directly connected to the Blockchain because the current Blockchain technology cannot process IoUT data in real-time. The Gateway Agent service has been suggested to bridge IoUT devices and Blockchain. The Gateway Agent needs to be placed at the proximity of IoUT devices to sense, temporarily store, and process IoUT data before permanently adding the data into Blockchain. In this architecture, the smart Gateway Agent is placed at the Fog layer that is close to surface nodes of the underwater network. The reason is IoUT devices require a coordination node that is closer to them to manage the Blockchain on their behalf. The Fog has been called an extension of Cloud to facilitate the computing capabilities to the bottom/edge of the network to provide faster communication, storage, and software services to the lower end devices [58]. Some computing resources, including storage and network services, have already been available on network devices such as a router, switch, and base station that are involved in routing the data produced from end devices. Further computing resources might be augmented to these devices to facilitate computing closer to the user's devices. Fog improves the latency, quality of service, and quality of experience over Cloud because of its proximity to user's devices.

The smart Gateway Agent in the architecture connects IoUT devices with the Blockchain through Base Station, as depicted in Figure 1. A Software Agent (SA) executed on the Gateway Agent depicted in Figure 5 gathers data from different cluster heads and makes Blocks for the consumers/customers who are interested in purchasing the IoUT data. The Gateway Agent coordinates and manages encryption keys for the Blockchain and IoUT devices. The Gateway Agent in the Fog does not directly take part in selecting a cluster head. Instead, the Gateway Agent provides a cluster head with a data encryption key. The Gateway Agent performs the role of certifying and approving the cluster head through Blockchain. The elected cluster head certifies its members.

Further, the Gateway Agent executes the selective Miner consensus protocol to reduce energy consumption in the Blockchain network. A small set of Miners are randomly selected with Proof of Stake (PoS) to validate the Block. The votes in favor of a Block are collected from the Miners selected using TOPSIS described in Section 3.3.3. If the majority of Miners approve the Block, the Block is confirmed in the Blockchain. The small set of Miners equally shares the mining fee for verifying the Block in the Blockchain. In addition, the Gateway Agent also maintains a Bloom filter to record IoUT devices' identities and separates benign and malicious Miners.

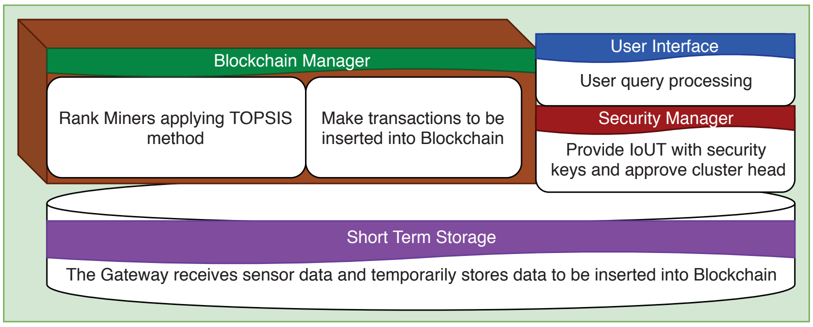

Blockchain Manager
Rank Miners applying TOPSIS method
Make transactions to be inserted into Blockchain
User Interface
User query processing
Security Manager
Provide IoUT with security keys and approve cluster head
Short Term Storage
The Gateway receives sensor data and temporarily stores data to be inserted into Blockchain

Figure 5. The functional block of the Gateway.

3.3. Blockchain on the Cloud Layer

In this section, we first discuss the motivation of using Cloud Blockchain to tackle IoUT data. Later, a customized Blockchain to be executed on the Cloud layer is described. The Proof of Stake (PoS) [59] is one of the more efficient consensus mechanisms. With PoS, any Miner can participate in processing Blocks by locking a certain amount of digital coin to the system. We modified this approach by nominating a set of healthy Miners using the TOPSIS method. Further, we present two use cases where the modified PoS can be applied to process IoUT in the Cloud Blockchain efficiently.

IoUT devices come with various properties such as different manufacturers, security protocols, and power and memory capacities. The integration of heterogeneous IoUT data and the maintenance of updated security protocols for IoUT devices are challenging in the conventional IoUT infrastructure [60]. Further, data generated from different IoUT devices are required to share with multiple stakeholders who are being operated by different legal rules and regulations. The cross border sharing of IoUT data poses the risk of malicious attacks and private information from being compromised. The current centralized architectures have proven not to be so efficient to protect data and disseminate data securely [61] among different stakeholders.

The Blockchain structure can ensure integrity, confidentiality, availability, trust, anonymity, authentication, authorization, user control, non-repudiation, and privacy for the user's record [62].

To be more precise, every transaction in Blockchain contains the user's signature, and Blockchain technology allows others to verify the legitimacy of the user's record. Hence, data provenance is strictly maintained in the Blockchain. Multiple nodes store the complete ledger in a decentralized fashion, which ensures a reliable and tamper-proof storage system. These features of Blockchain technologies have motivated researchers to utilize it to form a secure connection between heterogeneous IoUT devices.

However, the adoption of Blockchain in IoUT needs to address the massive power and memory cost, poor scalability, and legislative compliance issue pertaining to this technology. Blockchain provides IoUT with higher security compromising computation and storage. IoUT devices are power and memory constraint and cannot adopt the Blockchain [63].

The Cloud has been utilized to undertake the high processing and storage requirements for IoUT devices. However, the users need to trust a third-party Cloud service provider that cannot provide them with guaranteed accountability and traceability of their data [64]. This problem can be solved if the Cloud service providers formed a peer-to-peer network and adopt Blockchain technology on that network.

In the framework, we assume that the Cloud service providers support the considerable storage and processing power required for adopting the customized Blockchain. Unlike traditional Cloud services, the customer does not need to trust the Blockchain-based Cloud service provider. Further, public nodes such as computers or smartphones can also participate in validating Blocks.

The components of a standard Blockchain (used in Bitcoin) are described in Figure 6 before explaining the customized Blockchain in the next section.

1. The Blockchain operates on a peer-to-peer network depicted in Figure 6a consisting of three kinds of nodes: half node, general node, and Miner nodes.
2. The half node and general node produce transactions formatted as in Figure 6b and broadcast throughout the peer-to-peer network
3. The Miner nodes collect transactions and pack them into a Merkle tree to form a Block. Each Miner repeatedly inputs the Block into a cryptographic hash function incrementing the nonce field by one until it comes up with a target hash code for the Block. This process depicted in Figure 6c is called Proof of Work. Only one Miner that can first publish the Proof of Work for the Block receives rewards.
4. Finally, all nodes except the half-nodes add the Block to the end of the existing ledger, as depicted in Figure 6d, by executing the verification process.

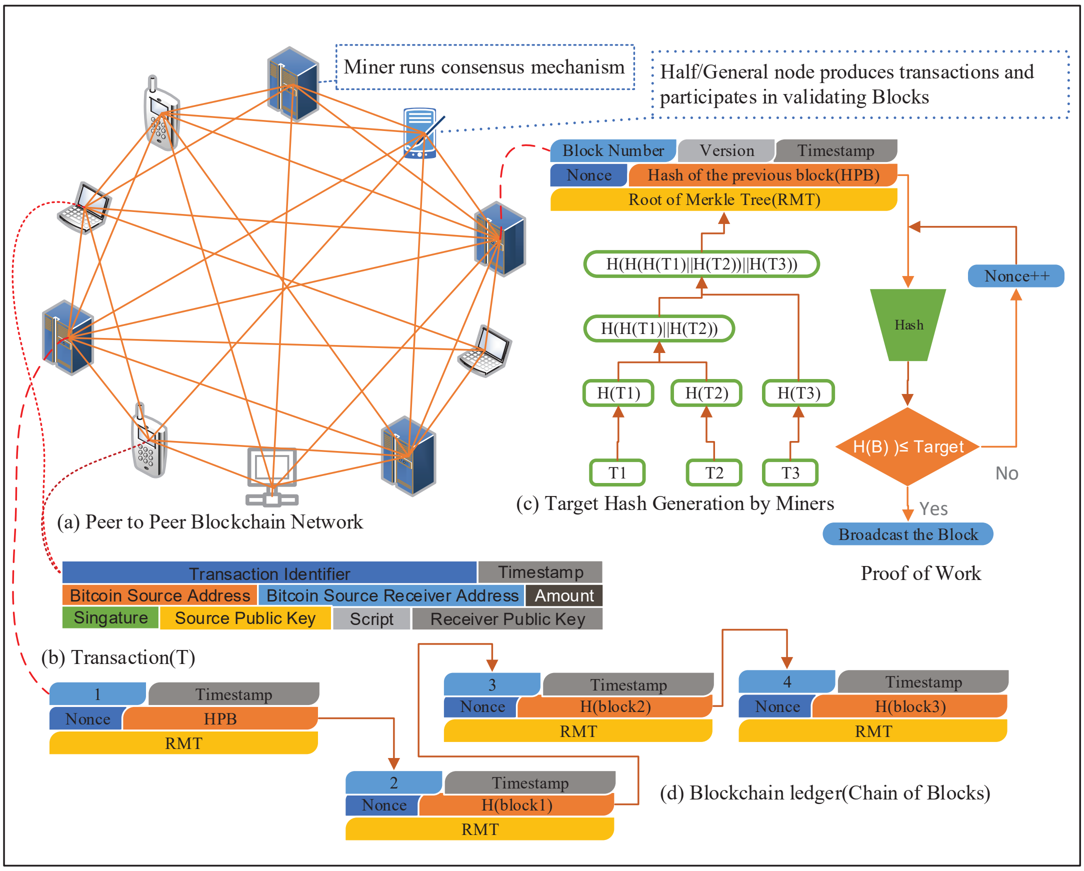


Figure 6. The Bitcoin Blockchain.

3.3.1. Transaction

A Blockchain transaction is made by the Gateway Agent when it receives a data packet from IoUT devices or other devices. A transaction format is represented in Table 4. The Gateway Agent makes different kinds of transactions such as IoUT data transaction, solution or outcome transaction, and financial transaction. A transaction may involve a large volume of data. Thus, the transaction on the Blockchain contains a pointer to the raw data. The sender indicates the Gateway Agent and the consumer indicates persons or organizations who purchase IoUT data. Public/private key is used to generate the signature, and the script describes the procedure for evaluating the transaction (e.g., verification of signature and the legitimacy of sender and receiver).

Table 4. The general format of transaction.

Transaction Identifier			
Transaction Type			
IoUT data or Pointer to Data			
Sender Address		Consumer Address	
Signature	Sender pubKey	Script	Consumer pubKey
Block Number			

3.3.2. Data Block Structure

The structure of a data Block is presented in Table 5. The Block has two parts: header and Data. In the header, Block Type represents the data type on the Block. For instance, the data Block contains records monitored by IoUT devices, and the problem-solution Block holds the transactions of a probable solution or outcome. The outcome-based Block is elucidated below. The Rate Vector field of the Block contains the rates given by Miners based on the accuracy of each solution. All transactions of a Block are organized in a Merkle tree. The Merkle tree ensures the integrity of the data and facilitates Miner to check the integrity without downloading each transaction in the Block.

Table 5. The format of data Block.

Block Header of Blockchain	
Field	Description
Version	Block Version Number
Block Types	IoUT data, or Problem solution, outcome, financial blocks
Previous Block Hash	Link for connecting Block
Timestamp	Block Creation time
Merkle Tree Root	Holds the root of Merkle tree to preserve integrity of transactions
Vote	Miner verifies the Block and votes
Rate Vector	Rating for problem solution/outcome transaction
Gateway Agent Signature	Source signature
Index Record	Holds outer and inner index information

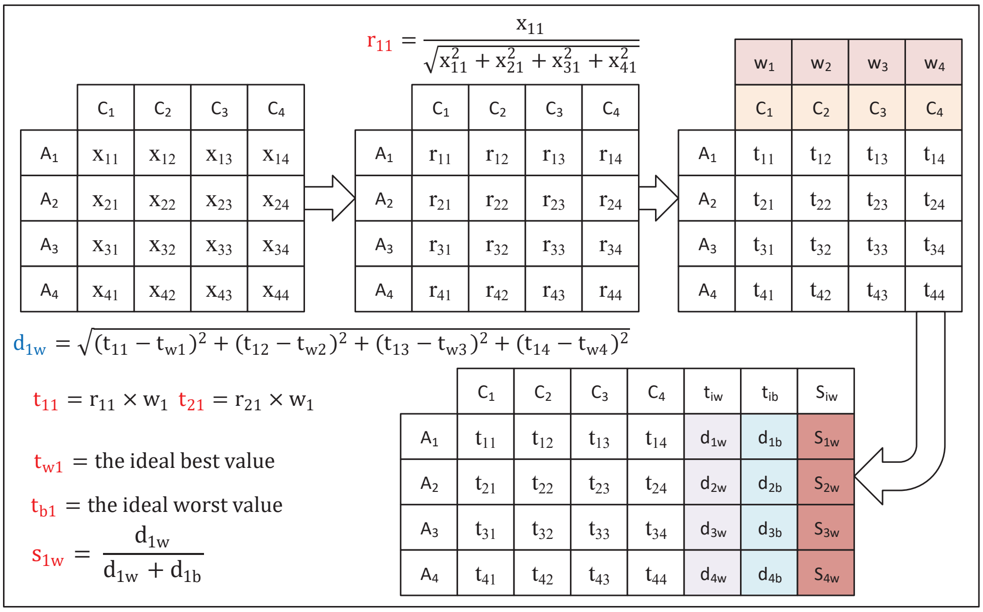
3.3.3. The Lightweight Consensus Mechanism

In this section, we first discuss the procedure for selecting Miners for the proposed lightweight consensus mechanism. Secondly, we present two use cases where the consensus mechanism can be applied.

The Gateway selects a group of healthy BC (Blockchain) nodes to make the consensus process faster and reliable. This group of nodes takes part in the mining process on the Cloud Blockchain. The Gateway intends to nominate a group of Miners based on multiple criteria of a BC node. The criteria might include reputation, attack vulnerability, amount of locked coin, processing power, storage capacity, availability, throughput, and mining cost; network capabilities such as bandwidth and computing; and other criteria. An efficient method to combine multiple criteria to discover a set of qualified BC nodes to perform mining for IoT data is necessary. In this article, we propose to use TOPSIS [65] (Technique for Order of Preference by Similarity to Ideal Solution) to rank BC nodes. TOPSIS, a multi-criteria decision analysis method, estimates the shortest geometric distance from the ideal best value and the longest geometric distance from the ideal worst value. Before applying TOPSIS, each selection criterion is weighted using AHP (Analytic Hierarchy Process) because each criterion mentioned above should not be considered as equally important. For example, reputation is two times more important than the amount of locked coin or stake; attack vulnerability is three times more desirable than storage capacity or availability; throughput is two times more expected than mining cost; and so on. AHP assigns a weight to each criterion using pairwise comparisons of the attributes or elements. TOPSIS-based Miner selection process is described below.

- Step 1: The BC peer-to-peer network is partitioned into several zones. Primarily, a Gateway randomly picks a certain number of nodes from each zone.
- Step 2: A rank evaluation matrix consisting of m randomly selected nodes from each zone as alternatives and n criteria is created. Each value in the matrix is identified as x_{ij}. The rank evaluation matrix is represented as $(x_{ij})_{m \times n}$.
- Step 3: Each value in the matrix is normalized according to the following formula: $r_{ij} = \frac{x_{ij}}{\sqrt{\sum\limits_{k=1}^{m} x_{kj}^2}}, \; i = 1, 2, \ldots, m, \; j = 1, 2, \ldots, n$

- Step 4: Each value of the matrix is normalized according to the following formula: $t_{ij} = r_{ij} \times w_j, \ i = 1, 2, \ldots, m \ j = 1, 2, \ldots, n$. Here, w_j, which indicates normalized weight for a criterion, is produced using AHP.
- Step 5: The worst alternative (A_w) and the best alternative (A_b) are chosen from each column (criterion) of the matrix. For example, for the criterion reputation, the best ideal value (alternative) is the maximum value of that column and the worst ideal value is the minimum value of that column. For criteria locked coin, processing power, storage capacity, availability, throughput, and network capability, the best ideal (alternative) value and worst ideal value are determined using the same formula. In contrast, for criteria attack vulnerability and mining cost, the best alternative is the minimum value of the respective columns and the worst alternative is the maximum value of the respective columns.
- Step 6: Geometric distance is estimated between the target alternative i and the worst condition A_w and A_b respectively as follows: $d_{iw} = \sqrt{\sum_{j=1}^{n} (t_{ij} - t_{wj})^2}, \ i = 1, 2, \ldots, m$ and $d_{ib} = \sqrt{\sum_{j=1}^{n} (t_{ij} - t_{bj})^2}, \ i = 1, 2, \ldots, m$.
- Step 7: Rank is estimated according to the following formula: $s_{iw} = \dfrac{d_{iw}}{d_{iw+d_{ib}}}, \ i = 1, 2, \ldots, m$
- Step 8: The Gateway selects a certain number of nodes following the ranking generated by using the TOPSIS. The TOPSIS process is illustrated in Figure 7.


$r_{11} = \dfrac{x_{11}}{\sqrt{x_{11}^2 + x_{21}^2 + x_{31}^2 + x_{41}^2}}$

	C₁	C₂	C₃	C₄
A₁	x_{11}	x_{12}	x_{13}	x_{14}
A₂	x_{21}	x_{22}	x_{23}	x_{24}
A₃	x_{31}	x_{32}	x_{33}	x_{34}
A₄	x_{41}	x_{42}	x_{43}	x_{44}

	C₁	C₂	C₃	C₄
A₁	r_{11}	r_{12}	r_{13}	r_{14}
A₂	r_{21}	r_{22}	r_{23}	r_{24}
A₃	r_{31}	r_{32}	r_{33}	r_{34}
A₄	r_{41}	r_{42}	r_{43}	r_{44}

	w₁	w₂	w₃	w₄
	C₁	C₂	C₃	C₄
A₁	t_{11}	t_{12}	t_{13}	t_{14}
A₂	t_{21}	t_{22}	t_{23}	t_{24}
A₃	t_{31}	t_{32}	t_{33}	t_{34}
A₄	t_{41}	t_{42}	t_{43}	t_{44}

$d_{1w} = \sqrt{(t_{11} - t_{w1})^2 + (t_{12} - t_{w2})^2 + (t_{13} - t_{w3})^2 + (t_{14} - t_{w4})^2}$

$t_{11} = r_{11} \times w_1$ $t_{21} = r_{21} \times w_1$

t_{w1} = the ideal best value

t_{b1} = the ideal worst value

$S_{1w} = \dfrac{d_{1w}}{d_{1w} + d_{1b}}$

	C₁	C₂	C₃	C₄	t_{iw}	t_{ib}	S_{iw}
A₁	t_{11}	t_{12}	t_{13}	t_{14}	d_{1w}	d_{1b}	S_{1w}
A₂	t_{21}	t_{22}	t_{23}	t_{24}	d_{2w}	d_{2b}	S_{2w}
A₃	t_{31}	t_{32}	t_{33}	t_{34}	d_{3w}	d_{3b}	S_{3w}
A₄	t_{41}	t_{42}	t_{43}	t_{44}	d_{4w}	d_{4b}	S_{4w}

Figure 7. The TOPSIS to select a group of Miners.

IoUT Data Block Inclusion: The following consensus mechanism is executed by the Miners to add a new IoUT data Block in the Blockchain. The procedure depicted in Algorithm 3 is described below.

- First, a group of Miners is nominated by the Gateway Agent to verify the newly created Block using the TOPSIS selection method described above.
- Second, Miners mainly verify the hash value of the immediate previous Block, the integrity of all transactions packed in the Merkle Tree of the Block, and the Gateway Agent's signature.
- Third, The Block is broadcast throughout the Blockchain network if specific numbers of the nominated Miners approves the Block as a valid Block.

- Last, the general node adds the Block to the end of the current chain of Blocks.

IoUT Outcome Block Inclusion: The user can request different kinds of services related to IoUT data management from a Gateway Agent. The Gateway Agent solicits services from various entities that participate in managing the Blockchain network. The Gateway utilizes a distributed trust feature of the Blockchain to record the services in the Blockchain. The Gateway Agent initiates the following consensus mechanism when it needs to add a service outcome or a problem solution such as classification or clustering for anomaly detection. The use case depicted in Algorithm 4 is explained below.

- First, the Gateway Agent nominates two small groups of Miners using the TOPSIS method.
- Second, the first group called **workers** generates their respective outputs on a given problem or proposes a solution to solve a problem, and the second group of Miners called **verifiers** rate the results or answer.
- Third, the **verifiers** accept a result of a developed solution only if the outcome from the solution does not exceed a threshold level of accuracy. The **verifiers** rate the solution/outcome based on the accuracy level.
- Last, the **workers** that obtain higher ratings on the solutions of the problem receive rewards for this.

For instance, the Gateway Agent solicits an efficient method from a group of Blockchain nodes (**workers**) to classify IoUT data as malicious and non-malicious data. The solutions from different **workers** will be packed into a Block and transmitted to the verifier group (also nominated by the Gateway Agent). The **verifiers** test the solution using their test dataset. The **verifiers** measure the accuracy level for each solution and rate the solution according to the accuracy level. The **workers** that obtain the highest rate for its solution will be financially rewarded. In such an application, the Gateway Agent does not rely on a third-party or centralized server for confirming a solution. The centralized server system may suffer from higher latency in such an application. In contrast, the Blockchain executing on the distributed peer-to-peer network ensures quick response on the user's task avoiding a single point of failure.

Algorithm 3: Data Block Validation.

```
Data: Block (B)
Result: Confirmed Block, Winner Miner
initialize $sigStatus = false$, $dataStatus = false$, $prevBlockHash = false$, $voteCount = 0$,
  $indexStatus = false$;
The Gateway Agent selects a group of Miners(M) using the TOPSIS method;
for each Miner $i = 0$ to $M$ do
    while $newBlock = true$ do
        $sigStatus \leftarrow$ checkSignature ($B$);
        $dataStutus \leftarrow$ checkDataIntegrity ($B$);
        $prevBlockHash \leftarrow$ checkPrevHash ($B$);
        $indexStatus \leftarrow$checkIndexRecord ($B$);
    end
    if $sigStatus == true \wedge dataStatus == true \wedge prevBlockHash == true \wedge indexStatus == true$ then
        $voteCount + +$;
    end
end
for each node $j = 1$ to $N$ do
    if $voteCount \geq thresholdCount$ then
        Insert the Block into the current ledger;
        Broadcast throughout the network;
    else
        Block is rejected;
    end
end
```

Algorithm 4: Outcome Block Validation Algorithm.

```
Data: Block(B[t]), worker[n], verifier[m]
Result: Ratings of a Block, Winner Miner
The Gateway Agent makes Block with outcome Transactions collected from worker group
The Gateway Agent transmits the Block to the verifier group
for each verifier $i = 0$ to $m$ do
    for each Transaction $t = 1$ to $n$ do
        $accuracy \leftarrow$ testSolution ($B[t]$);
        if $accuracy \geq thresholdAccuracy$ then
            $rating[t] \leftarrow$ rating[t]+ rateEstimation($accuracy$);
        end
    end
end
```

3.3.4. Multilevel Index on the Blockchain

The replication of the complete ledger is a significant reason for having higher security from Blockchain technology. Ledger replication enables an entity to validate IoT data without relying on the third-party. Further, data integrity and availability are guaranteed by replicating the ledger amongst multiple nodes. The distributed decentralized Blockchain storage facilitates multiple active data access point. The other benefit of the distributed ledger is that Blockchain can withstand the single point of failure, Ransomware, and Denial of Service attacks.

In general, transactions related to results or outcome generated from the analysis of raw IoT data, and the transactions related to software update and firmware are occasionally produced. These kinds of transaction can be replicated among multiple entities.

However, nodes that process data streamed from IoUT require a large volume of storage. Although the Cloud server supports massive storage for the Blockchain, volunteers or storage constrained devices might be reluctant to participate in the Blockchain [30].

Validators in IoUT Blockchain do not require the complete chain of Blocks as in digital cryptocurrencies (Bitcoin) to confirm a new Block. Further, the consumer or customer is most interested in the most recent IoUT because recent data are typically the most significant in real-time IoUT monitoring. In such a case, few Cloud servers are allowed to store the complete ledger, and all other validators cache the most recent IoUT data.

Multilevel index records based on time attribute can be maintained in the Blockchain to track the most recent Blocks. A multilevel Blockchain index is depicted in Figure 8. In Figure 8, the multilevel index comprises the outer index and inner index. Every index record has a search key and two pointers where one pointer holds the hash code of the previous index record of the same level, and another pointer contains the hash code of the next level index record. The link between index records can preserve record integrity and prevent index records from being tampered.

Outer and inner index use year, month, and day, respectively, as a search key. Block's content can be accessed through metadata representing the header's information of a Block. Every entity in the Blockchain stores the complete multilevel index into the main memory and the most recent chain of Blocks in permanent storage rather than store the whole ledger. Several Cloud servers store the entire chain of Blocks. Consequently, limited memory nodes such as smartphones or laptop machines can take part in performing the verification process.

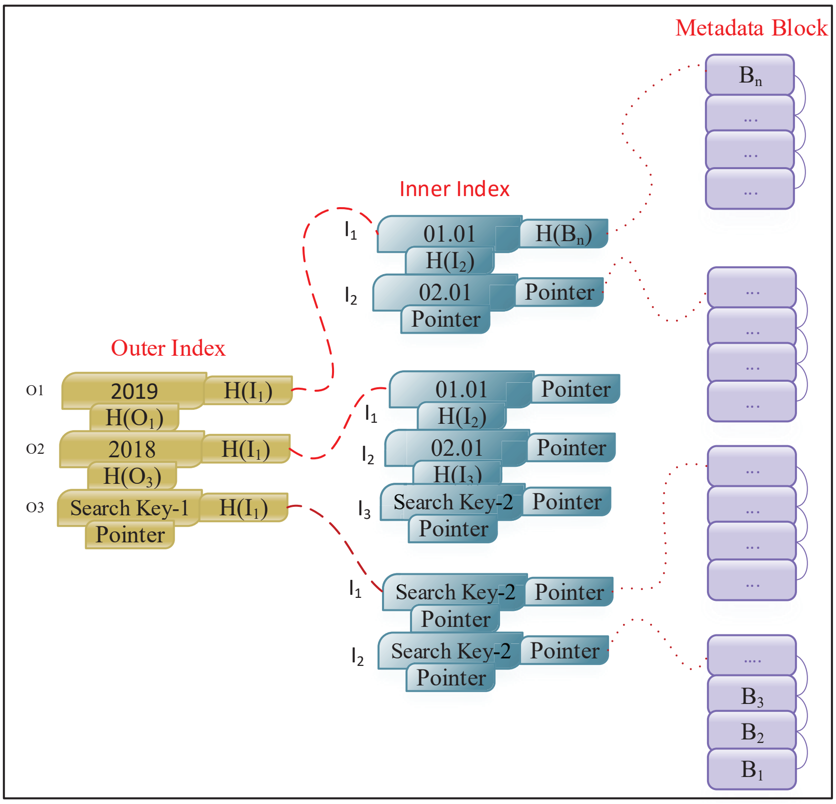


Figure 8. Time-based multilevel index record in IoUT Blockchain.

4. Performance Analysis

This section contains the performance analysis of the proposed architecture in terms of Block generation energy consumption, time, reliability, remaining energy, and standard security attack.

The architecture was simulated using Java Programming following the iFogSim [66]. The simulation parameter is illustrated in Table 6. IoUT devices with different power and memory capacity are considered in the simulation. A private Blockchain module is implemented using the idea introduced in [67]. To implement the second use case of the consensus mechanism that is used to confirm a solution for a particular problem in the Blockchain, we considered a problem of anomaly detection given a dataset. The worker Miners proposed six classifiers, namely C5, C45̇ [68], SVM, Naive Bayes (NB), Multilayer perception (MLP), Random Forest (RF) and other, as a probable solution of the problem. Each worker node proposed a classifier and put the classifier's link in the solution transactions. The Blockchain verifier Miner used KDD Cup 1999 Data [69] to measure the accuracy illustrated in Figure 9 of the classifiers by Weka [70]. The threshold accuracy level was assumed as 80%. The Miner accepted the solution, which had a skill above 80%. Here, the worker that proposed C5 with the highest accuracy obtained the highest rating from the verifier and was rewarded.

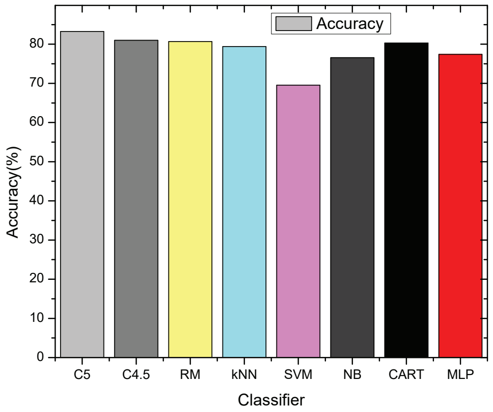


Figure 9. The Accuracy of different classifier to detect anomaly.

Table 6. The Simulation Parameters.

Simulated Network Area	1000 × 1000 m^2
Radio Range of a node	100 m
MIPS (Million instructions per second) of Gateway Agent at Fog layer	9900 M–83,000 M
Cloud Node MIPS	317,900 M–113,093 M
IoUT device MIPS	14,000 M
RAM capacity of each node	816 MB
Gateway Agent Bandwidth	600–300 Mbps
IoUT device Bandwidth	100–50 Mbps
Gateway Agent power consumption rate (per Hour)	140–95 W
Gateway Agent power consumption rate (per Hour) while transmitting data	10 W
IoUT device power consumption rate (per Hour)	25–20 W
IoUT device transmission power consumption rate (per Hour)	2
Blockchain Transaction Size	1024 bytes
Block Size of the Blockchain	10 × 1024 bytes
Instructions required to validate a Block	10 M

The simulation was executed in a 1000 × 1000 m^2 area. Different numbers of IoUT nodes (100, 200, 300, 400 and 500) were considered for different executions. We analyzed the performances of the architecture concerning the following parameters.

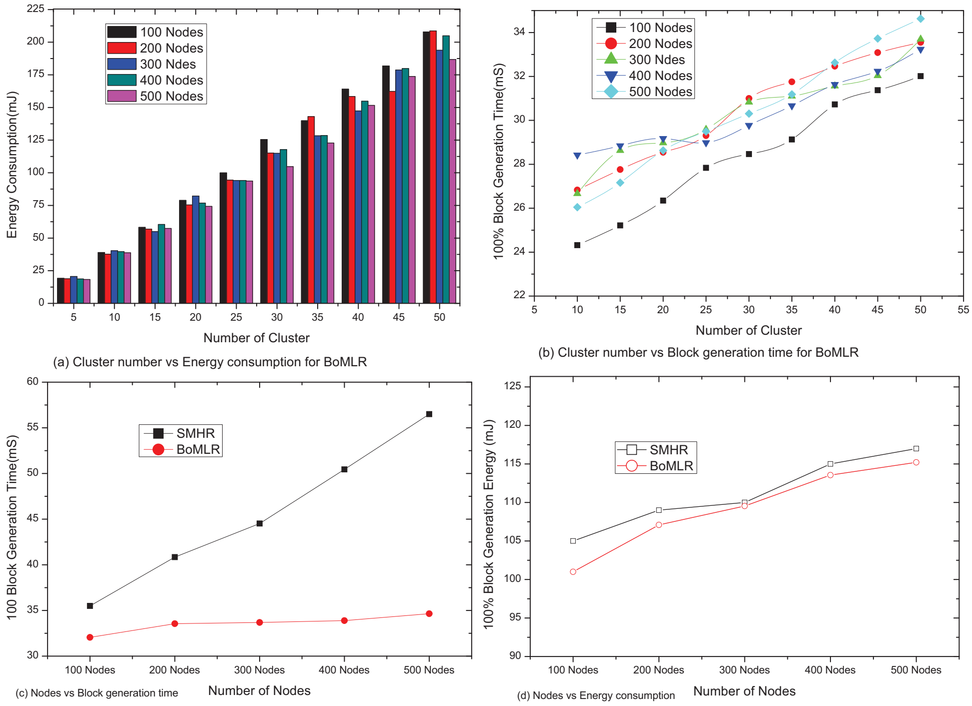
- Energy Consumption: The energy consumption required for the simulated network includes energy for transmitting, receiving transactions, and validation of a certain number of Blocks.
- Block Generation Time: The Block Generation time includes the time required for transmitting, making a certain number (100%) of Blocks, and validating the Blocks.

The simulation was run ten times, and the average value from 10 times simulation was used to represent the performance graph. The energy consumption to generate 100 Blocks using the proposed hierarchical routing under a different number of clusters (10, 15, 20, ..., 50) and nodes is illustrated in Figure 10a. The energy consumption increases with increasing cluster numbers as more nodes participate in sensing and forwarding data. However, for a particular cluster number with a variable number of nodes, the energy consumption does not significantly vary. The Block generation time in BoMLR (Blockchain Oriented Multilevel Routing) is illustrated in Figure 10b. Block generation time keeps increasing with higher numbers of nodes and clusters. The fixed number of Gateway Agents experiences some queue latency because of the higher number of associated cluster heads, which reduces the Block per second.

The comparison of Block generation energy consumption and time for the BoML (Blockchain oriented multilevel) and SMH (secure multi-hop) routing are depicted in Figures 10c,d, respectively. The SMHR shows higher Block generation energy consumption and time than BoMLR. In SMHR, authentication is performed between IoUT source node and the forwarding node before transferring data. The BoMLR does not need to perform authentication between the source node and the forwarding node before sending data as the node joins a cluster head through authentication. In BoMLR, IoUT devices do not need to generate a new key every time they transmit data to the Gateway. The IoUT devices do not need to exchange the control packet to establish a data forwarding route. Only nodes with higher levels participate in transferring the data packet to the Gateway Agent. Therefore, Block generation delay and energy consumption will be minimized.

Key management and level improve the packet delivery ratio of the proposed protocol. Bitcoin's Blockchain processes around 3–4, and Ethereum [71] processes around 20 transactions per second with Proof of Work that requires a high computational cost. In contrast, Proof of Stake used in our current setting can produce around 2500 Blocks per second. In the proposed architecture, some benign Miners validate the Block in Proof of Stake fashion unless forks are created. Forks on the Blockchain are resolved with Proof of Work and the longest chain rule. The hybrid Proof of Stake will process a higher number of transactions per second than Proof of Work. The PoW consensus mechanism used in the Bitcoin Blockchain demands high computational cost. The Proof of Stake used in the proposed Blockchain does not waste power unless forks occur. Further, Gateway Agent selects a small set of Miners to validate and confirm the Block in the Blockchain. The selection of a small set of Miners also improves the end-to-end energy consumption required for generating Blocks. Specifically, the consensus mechanism of the Blockchain can ensure better QoS for an individual.

The remaining energy comparison of the Internet of Things layer for BoMLR, SMHR [39], HEED [42], and LEACH [41] protocol is illustrated in Figure 11a. HEED and LEACH show higher remaining energy in comparison to BoMLR and SMHR because no security is employed in those routing approaches. However, our BoMLR's network lifetime is higher than SMHR. In BoMLR, cluster head selection does not require the exchange of a control message to find the new cluster head as the current cluster head selects the next cluster head. Further, the next cluster head does not need to rediscover its member nodes as the present cluster head shares the Bloom filter holding the member nodes' identifier with the next cluster head. The new cluster head does not need to send a control message containing cluster head information to its members. Member nodes just broadcast their data throughout the cluster, and the cluster head will receive the data as the cluster head's level is greater than its members. Lightweight HMAC verifies the legitimate source node of a data packet. The aggregated data from the general nodes are forwarded to the Gateway Agent by only the cluster heads. The level number associated with each IoUT device guides the node to forward the data packet to the destination. In the proposed framework, avoidance of control message to discover the cluster head and forwarding nodes, involvements of fewer IoUT devices to forward the data packets results in lower power consumption in the IoUT network layer.



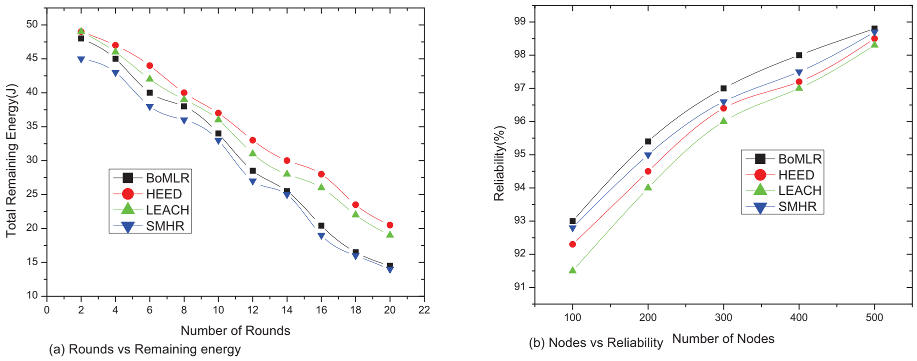
(a) Cluster number vs Energy consumption for BoMLR

(b) Cluster number vs Block generation time for BoMLR

(c) Nodes vs Block generation time

(d) Nodes vs Energy consumption

Figure 10. The comparison of Block generation energy and time.



(a) Rounds vs Remaining energy

(b) Nodes vs Reliability

Figure 11. The comparison of remaining energy and reliability.

The reliability graph is presented in Figure 11b. HEED and LEACH are vulnerable to different attacks because of the absence of a security approach while routing the data packet. SMHR always uses hardware inserted keys. In this approach, if a key is compromised, the packet is always captured by the attacker. On the other hand, in BoMLR, IoUT devices also use hardware inserted symmetric key to transmit data securely. However, IoUT devices obtain firmware updates for the keys from the Gateway Agent through the Blockchain. Thus, IoUT devices are not required to generate new encryption keys regularly. Further, key compromises do not impact heavily on the routing because of updating a firmware through Blockchain.

The IoUT framework may encounter diverse kinds of cyberattacks, and the mitigation of those attacks in light of our IoUT framework are presented in Tables 7 and 8.

Table 7. The Security Attack and Mitigation.

Attack Name	**Description**	**Mitigation**
Nothing at Stake attack [72]	Nothing at stake is a situation that occurred in Proof of Stake where a Miner loses nothing misbehaving but stands to be rewarded. In Proof of Stake, a malicious Miner adds a bad transaction in a fork of a Blockchain and all Miners keep validating on two forks in the Blockchain. The malicious Miner waits for the confirmation of the bad transaction and stops validating on the legitimate chain. Thus, the chain holding bad transactions is confirmed and all Miners validating both chains receive reward even one of the chain is discarded.	Proof of Work safeguards Blockchain from nothing at stake attack as a Miner can consume power for only one of the chains at the same time. The Miner can be forced to switch from PoS to PoW when a fork is created in the Blockchain. In our IoUT architecture, a Miner runs Proof of Work when forks occur in the Blockchain until the fork is resolved. Further, the Miner is penalized and enlisted as a malicious Miner by the Gateway if it validates more than one chain.
Long-Range Attack [73]	A long-range attack is a scenario where an adversary creates a branch on the Blockchain starting from the Genesis Block and overtakes the main chain. The reasons for long-range attacks are weak subjectivity, costless simulation of PoS. The weak subjectivity occurs when new nodes or nodes that are off-line for a long time can be fooled to recognize the wrong genesis Blocks or chain. Costless simulation of PoS refers to the ability of a Miner to validate multiple chains of Blocks without spending money or power).	The long-range attack can be mitigated in the following ways: longest rule chain where the Miners overtakes the longest chain, moving checkpoints where only the latest **X** number of Blocks of the chain can be reorganized, and context-aware transactions where every transaction has a field called Block Number that indicates to which Block the transaction belongs. Both Gateway and Blockchain use the above-outlined method to mitigate a long-range attack.
Sybil Attack	An attacker creates multiple false identifiers to control a peer network. In the proposed architecture, the Gateway Agent is vulnerable to Sybil attack because of choosing a group of Miner.	To prevent such an attack, the Gateway Agent might charge high costs from the Miners while creating a new identity and choose Miners based on their trust and reputation. Similarly, mobile IoUT devices require to register to the Gateway Agent to join the peer-to-peer network.
Eclipse Attack	A hacker controls a large number of IP addresses to make a distributed botnet to overwrite the controlled IP address on the tried Table of the victim nodes and wait until the victim node is restarted. The Gateway Agent is vulnerable to eclipse attack.	The Gateway Agent can verify the IP address registered in the Blockchain and can avoid the impact of eclipse attack
Selfish Mining	Selfish Miners attempt to increase their share by not broadcasting mined Blocks throughout the network for some period and then releases several Blocks at a time, making other Miners lose their Blocks.	The Gateway Agent mitigates such an attempt by selecting a group of Miners from different zones randomly, defining the maximum acceptable time of a Block generation and preparing Blocks with the most recent timestamp.
Replay attack	Replay attack occurred when the attacker intercepts streamed messages to be exchanged between two legitimate entities and delay or resend the stream to one or more of the bodies.	The session timestamp in the data packet prevents an attacker from replaying the same data transaction or control message.

Table 8. The Security Attack and Mitigation.

Attack Name	**Description**	**Mitigation**
Eavesdropping	Eavesdropping, known as sniffing or snooping attack, is an incursion where an attacker steals information transmitted over a network.	The cluster head sends the encrypted key to its members. The cluster head transmits ciphertext (produced by Gateway Agent's key) of the data to the destination Gateway Agent. The cluster head and the Gateway Agent update their key through Blockchain. All communications occur using a secure channel.
Spoofing Attack	Attackers might change the identity of the source IoUT device, which is called the spoofing attack.	The data transaction contains the hash message authentication code produced from the identifier and other information of the sender. Therefore, a spoofing attack is not possible in the proposed routing mechanism.
Denial of Service attack	The perpetrator floods or overwhelms a targeted machine by issuing requests and thus make the devices inaccessible to other intended entities. The Gateway Agent might be the target of the DoS attack and results in a single point of failure. The cluster heads associated with the affected Gateway Agent can request other Gateways to resume services for them.	Denial of Service (DoS) attack might occur in IoUT routing. Cluster head and the Gateway may be the target of the DoS attack. Bloom filter helps the cluster head or Gateway Agent to reject packets coming from non-listed nodes. DoS attack does not succeed in the Blockchain because multiple nodes contain the exact copy of the user record and fake Blocks require mining fee to be added into the Blockchain which discourages attacks from making fake Block [30].
Rouge Gateway agent	A rogue agent can compromise a Gateway, or an attacker can pretend to be a Gateway agent. The rogue agent can hold IoUT and decide not to make transactions and send them in the Blockchain network	The Gateway Agent requires to agree on Blockchain protocol suits to act as a broker between IoT devices and the Blockchain network. The consensus mechanism is designed on the standard Proof of Stake in which Miners and the Gateway must deposit digital coin to participate in processing Block in the Blockchain. Thus, a Gateway Agent or Miner discovered as a rouge agent will lose its stake.
Network Analysis	An attack can map a transaction to a particular profile using behavior-based clustering techniques.	Transactions are required to be encrypted so that attack cannot trace and analyze data patterns. Node's identity should be changed to prevent the attackers from tracing transfer history of a token in the Blockchain.
Spreading Illegal Content	Attackers can insert illegal contents into a transaction and broadcast it throughout the network	The smart Gateway provides an IoUT device with a data encryption key. Therefore the Gateway can check the contents of a transaction is before sending it to the Blockchain. However, if the Gateway is compromised and makes encrypted transactions, the Block containing illegal contents is committed in the Blockchain.

5. Conclusions

In this paper, we propose a multilayer hierarchical architecture for monitoring and managing IoUT data using Blockchain on the Cloud. The proposed solution organizes sensors into clusters and selects a cluster head based on the residual energy and a node's level to route the data to a higher layer. The cluster head uses a Bloom filter to track member nodes. The underwater IoT routing protocol uses a standard secret key if the sensors are from the same manufacturer. For communicating data with the Gateway, the cluster head uses another secret key, which is provided by the Gateway. Finally, the data are stored in the Cloud using a Blockchain. The smart Gateway in the Fog layer integrates the Blockchain network with the underwater IoT devices to solve the scalability issue, prepare transactions, and route them to Miners. The Block is added to the Blockchain by running a lightweight consensus mechanism. The Blockchain in the Cloud executes the consensus mechanism for inserting IoUT data into the Blockchain. Finally, a performance analysis and mitigation of security attacks analysis demonstrated the feasibility of the architecture to monitor IoUT data.

Author Contributions: M.A.U. designed the different components of the architecture and implemented the framework. A.S. contributed in writing, editing, and refining the idea. A.S., I.G., and V.B. reviewed the article.

Acknowledgments: The research was supported by Internet Commerce Security Laboratory, Federation University Australia. The authors are grateful to Centre for Informatics and Applied Optimization (CIAO) for their support.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Domingo, M.C. An overview of the internet of underwater things. *J. Netw. Comput. Appl.* **2012**, *35*, 1879–1890.
2. Akhter, A.; Uddin, M.A.; Abir, M.A.I.; Islam, M.M. Noise aware level based routing protocol for underwater sensor networks. In Proceedings of the 2016 International Workshop on Computational Intelligence (IWCI), Dhaka, Bangladesh, 12–13 December 2016; pp. 169–174.
3. Li, N.; Martínez, J.F.; Meneses Chaus, J.; Eckert, M. A survey on underwater acoustic sensor network routing protocols. *Sensors* **2016**, *16*, 414.
4. Han, G.; Jiang, J.; Sun, N.; Shu, L. Secure communication for underwater acoustic sensor networks. *IEEE Commun. Mag.* **2015**, *53*, 54–60.
5. Uddin, A.; Mamun-or-Rashid. Link expiration time-aware routing protocol for UWSNs. *J. Sens.* **2013**, *2013*, 1–9, doi:10.1155/2013/625274.
6. Faheem, M.; Tuna, G.; Gungor, V.C. LRP: Link quality-aware queue-based spectral clustering routing protocol for underwater acoustic sensor networks. *Int. J. Commun. Syst.* **2017**, *30*, e3257.
7. Kim, M.; Lim, K.S.; Song, J.; Jun, M.-S. An efficient secure scheme based on hierarchical topology in the smart home environment. *Symmetry* **2017**, *9*, 143.
8. Jo, B.; Khan, R.; Lee, Y.S. Hybrid Blockchain and internet-of-things network for underground structure health monitoring. *Sensors* **2018**, *18*, 4268.
9. Jiang, J.; Han, G.; Guo, H.; Shu, L.; Rodrigues, J.J. Geographic multipath routing based on geospatial division in duty-cycled underwater wireless sensor networks. *J. Netw. Comput. Appl.* **2016**, *59*, 4–13.
10. Al Salti, F.; Alzeidi, N.; Arafeh, B.R. EMGGR: An energy-efficient multipath grid-based geographic routing protocol for underwater wireless sensor networks. *Wirel. Netw.* **2017**, *23*, 1301–1314.
11. Wei, Z.; Song, M.; Yin, G.; Song, H.; Wang, H.; Ma, X.; Cheng, A. Data access based on a guide map of the underwater wireless sensor network. *Sensors* **2017**, *17*, 2374.
12. Lal, C.; Petroccia, R.; Pelekanakis, K.; Conti, M.; Alves, J. Toward the development of secure underwater acoustic networks. *IEEE J. Ocean. Eng.* **2017**, *42*, 1075–1087.
13. Dhumane, A.; Prasad, R.; Prasad, J. Routing issues in internet of things: A survey. In Proceedings of the International Multiconference of Engineers and Computer Scientists, Hong Kong, China, 16–18 March 2016; Volume 1, pp. 16–18.

14. Yan, H.; Shi, Z.J.; Cui, J.H. DBR: Depth-based routing for underwater sensor networks. In Proceedings of the International Conference on Research in Networking, Singapore, 5–8 May 2008; Springer: Hoboken, NJ, USA, 2008; pp. 72–86.
15. Faz-Hernández, A.; López, J.; Ochoa-Jiménez, E.; Rodríguez-Henríquez, F. A faster software implementation of the supersingular isogeny Diffie-Hellman key exchange protocol. *IEEE Trans. Comput.* **2017**, *67*, 1622–1636.
16. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* **2013**, *29*, 1645–1660.
17. Jawhar, I.; Mohamed, N.; Al-Jaroodi, J.; Zhang, S. An architecture for using autonomous underwater vehicles in wireless sensor networks for underwater pipeline monitoring. *IEEE Trans. Ind. Inform.* **2018**, *15*, 1329–1340.
18. Mohamed, N.; Jawhar, I.; Al-Jaroodi, J.; Zhang, L. Sensor network architectures for monitoring underwater pipelines. *Sensors* **2011**, *11*, 10738–10764.
19. Fernández-Caramés, T.M.; Fraga-Lamas, P. A Review on the Use of Blockchain for the Internet of Things. *IEEE Access* **2018**, *6*, 32979–33001.
20. Onik, M.M.H.; Aich, S.; Yang, J.; Kim, C.S.; Kim, H.C. Blockchain in Healthcare: Challenges and Solutions. In *Big Data Analytics for Intelligent Healthcare Management*; Elsevier: Amsterdam, The Netherlands, 2019; pp. 197–226.
21. Ali, O.; Shrestha, A.; Soar, J.; Wamba, S.F. Cloud computing-enabled healthcare opportunities, issues, and applications: A systematic review. *Int. J. Inf. Manag.* **2018**, *43*, 146–158.
22. Mesbahi, M.R.; Rahmani, A.M.; Hosseinzadeh, M. Reliability and high availability in cloud computing environments: A reference roadmap. *Hum.-Centric Comput. Inf. Sci.* **2018**, *8*, 20.
23. Kuo, T.T.; Kim, H.E.; Ohno-Machado, L. Blockchain distributed ledger technologies for biomedical and health care applications. *J. Am. Med. Inform. Assoc.* **2017**, *24*, 1211–1220.
24. Guo, R.; Shi, H.; Zhao, Q.; Zheng, D. Secure attribute-based signature scheme with multiple authorities for blockchain in electronic health records systems. *IEEE Access* **2018**, *6*, 11676–11686.
25. Yaqoob, S.; Khan, M.M.; Talib, R.; Butt, A.D.; Saleem, S.; Arif, F.; Nadeem, A. Use of Blockchain in Healthcare: A Systematic Literature Review. *Int. J. Adv. Comput. Sci. Appl.* **2019**, *10*, doi:10.14569/IJACSA.2019.0100581.
26. Zubaydi, H.D.; Chong, Y.W.; Ko, K.; Hanshi, S.M.; Karuppayah, S. A Review on the Role of Blockchain Technology in the Healthcare Domain. *Electronics* **2019**, *8*, 679.
27. Faramondi, L.; Oliva, G.; Setola, R.; Vollero, L. IIoT in the Hospital Scenario: Hospital 4.0, Blockchain and Robust Data Management. In *Security and Privacy Trends in the Industrial Internet of Things*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 271–285.
28. Uddin, M.A.; Stranieri, A.; Gondal, I.; Balasubramanian, V. A Patient Agent to Manage Blockchains for Remote Patient Monitoring. *Stud. Health Technol. Inform.* **2018**, *254*, 105–115.
29. 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 2017; ACM: New York, NY, USA, 2017; pp. 173–178.
30. Uddin, M.A.; Stranieri, A.; Gondal, I.; Balasubramanian, V. Continuous Patient Monitoring with a Patient Centric Agent: A Block Architecture. *IEEE Access* **2018**, *6*, 32700–32726.
31. Zyskind, G.; Nathan, O.; Pentland, A. Decentralizing privacy: Using blockchain to protect personal data. In Proceedings of the 2015 IEEE Security and Privacy Workshops (SPW), San Jose, CA, USA, 21–22 May 2015; pp. 180–184.
32. Lei, A.; Cruickshank, H.; Cao, Y.; Asuquo, P.; Ogah, C.P.A.; Sun, Z. Blockchain-Based Dynamic Key Management for Heterogeneous Intelligent Transportation Systems. *IEEE Internet Things J.* **2017**, *4*, 1832–1843.
33. Tian, F. An agri-food supply chain traceability system for China based on RFID & blockchain technology. In Proceedings of the 2016 13th International Conference on Service Systems and Service Management (ICSSSM), Kunming, China, 24–26 June 2016; pp. 1–6.
34. Samaniego, M.; Deters, R. Using Blockchain to push Software-Defined IoT Components onto Edge Hosts. In Proceedings of the International Conference on Big Data and Advanced Wireless Technologies, Blagoevgrad, Bulgaria, 10–11 November 2016; ACM: New York, NY, USA, 2016; p. 58.
35. Aazam, M.; Hung, P.P.; Huh, E.N. Smart gateway based communication for cloud of things. In Proceedings of the 2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP), Singapore, 21–24 April 2014; pp. 1–6.

36. Sharma, P.K.; Chen, M.Y.; Park, J.H. A software defined fog node based distributed blockchain cloud architecture for IoT. *IEEE Access* **2018**, *6*, 115–124.
37. Uddin, M.A.; Stranieri, A.; Gondal, I.; Balasubramanian, V. An Efficient Selective Miner Consensus Protocol in Blockchain Oriented IoT Smart Monitoring. In Proceedings of the 2019 IEEE International Conference on Industrial Technology (ICIT), Melbourne, Australia, 13–15 February 2019; pp. 1135–1142, doi:10.1109/ICIT.2019.8754936.
38. Tian, Y.; Hou, R. An improved AOMDV routing protocol for internet of things. In Proceedings of the 2010 International Conference on Computational Intelligence and Software Engineering (CiSE), Wuhan, China, 10–12 December 2010; pp. 1–4.
39. Chze, P.L.R.; Leong, K.S. A secure multi-hop routing for IoT communication. In Proceedings of the 2014 IEEE World Forum on Internet of Things (WF-IoT), Seoul, Korea, 6–8 March 2014; pp. 428–432.
40. Iova, O.; Theoleyre, F.; Noel, T. Using multiparent routing in RPL to increase the stability and the lifetime of the network. *Ad Hoc Netw.* **2015**, *29*, 45–62.
41. Heinzelman, W.B.; Chandrakasan, A.P.; Balakrishnan, H. An application-specific protocol architecture for wireless microsensor networks. *IEEE Trans. Wirel. Commun.* **2002**, *1*, 660–670.
42. Younis, O.; Fahmy, S. HEED: A hybrid, energy-efficient, distributed clustering approach for ad hoc sensor networks. *IEEE Trans. Mob. Comput.* **2004**, *3*, 366–379.
43. Liu, G.; Wei, C. A new multi-path routing protocol based on cluster for underwater acoustic sensor networks. In Proceedings of the 2011 International Conference on Multimedia Technology, Hangzhou, China, 26–28 July 2011; pp. 91–94.
44. Guangzhong, L.; Zhibin, L. Depth-based multi-hop routing protocol for underwater sensor network. In Proceedings of the 2nd International Conference on Industrial Mechatronics and Automation, Wuhan, China, 30–31 May 2010; Volume 2, pp. 268–270.
45. Wahid, A.; Lee, S.; Kim, D. A reliable and energy-efficient routing protocol for underwater wireless sensor networks. *Int. J. Commun. Syst.* **2014**, *27*, 2048–2062.
46. Domingo, M.C. A distributed energy-aware routing protocol for underwater wireless sensor networks. *Wirel. Pers. Commun.* **2011**, *57*, 607–627.
47. Ayaz, M.; Abdullah, A.; Jung, L.T. Temporary cluster based routing for underwater wireless sensor networks. In Proceedings of the 2010 International Symposium on Information Technology, Kuala Lumpur, Malaysia, 15–17 June 2010; Volume 2, pp. 1009–1014.
48. Diao, B.; Xu, Y.; An, Z.; Wang, F.; Li, C. Improving both energy and time efficiency of depth-based routing for underwater sensor networks. *Int. J. Distrib. Sens. Netw.* **2015**, *11*, 781932.
49. Latif, K.; Javaid, N.; Ahmad, A.; Khan, Z.A.; Alrajeh, N.; Khan, M.I. On energy hole and coverage hole avoidance in underwater wireless sensor networks. *IEEE Sens. J.* **2016**, *16*, 4431–4442.
50. Malina, L.; Hajny, J.; Fujdiak, R.; Hosek, J. On perspective of security and privacy-preserving solutions in the internet of things. *Comput. Netw.* **2016**, *102*, 83–95.
51. Jayaraman, P.P.; Yang, X.; Yavari, A.; Georgakopoulos, D.; Yi, X. Privacy preserving Internet of Things: From privacy techniques to a blueprint architecture and efficient implementation. *Future Gener. Comput. Syst.* **2017**, *76*, 540–549.
52. Shafagh, H.; Hithnawi, A. Poster: Come closer: Proximity-based authentication for the internet of things. In Proceedings of the 20th Annual International Conference on Mobile Computing and Networking, Maui, HI, USA, 7–11 September 2014; ACM: New York, NY, USA, 2014; pp. 421–424.
53. Li, T.; Ren, J.; Tang, X. Secure wireless monitoring and control systems for smart grid and smart home. *IEEE Wirel. Commun.* **2012**, *19*, 66–73.
54. Ngo, H.H.; Wu, X.; Le, P.D.; Wilson, C.; Srinivasan, B. Dynamic Key Cryptography and Applications. *IJ Netw. Secur.* **2010**, *10*, 161–174.
55. Lee, B.; Lee, J.H. Blockchain-based secure firmware update for embedded devices in an Internet of Things environment. *J. Supercomput.* **2017**, *73*, 1152–1167.
56. Smhasher. Available online: https://github.com/aappleby/smhasher (accessed on 22 July 2018).
57. Cityhash. Available online: https://github.com/google/cityhash (accessed on 22 July 2018).
58. Farahani, B.; Firouzi, F.; Chang, V.; Badaroglu, M.; Constant, N.; Mankodiya, K. Towards fog-driven IoT eHealth: Promises and challenges of IoT in medicine and healthcare. *Future Gener. Comput. Syst.* **2018**, *78*, 659–676.

59. Tosh, D.; Shetty, S.; Foytik, P.; Kamhoua, C.; Njilla, L. CloudPoS: A proof-of-stake consensus design for blockchain integrated cloud. In Proceedings of the 2018 IEEE 11th International Conference on Cloud Computing (CLOUD), San Francisco, CA, USA, 2–7 July 2018; pp. 302–309.
60. Hassan, M.U.; Rehmani, M.H.; Chen, J. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. *Future Gener. Comput. Syst.* **2019**, *97*, 512–529.
61. Moin, S.; Karim, A.; Safdar, Z.; Safdar, K.; Ahmed, E.; Imran, M. Securing IoTs in distributed blockchain: Analysis, requirements and open issues. *Future Gener. Comput. Syst.* **2019**, *100*, 325–343.
62. Khan, M.A.; Salah, K. IoT security: Review, blockchain solutions, and open challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411.
63. 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.
64. Ali, M.S.; Vecchio, M.; Pincheira, M.; Dolui, K.; Antonelli, F.; Rehmani, M.H. Applications of blockchains in the internet of things: A comprehensive survey. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 1676–1717.
65. Balioti, V.; Tzimopoulos, C.; Evangelides, C. Multi-Criteria Decision Making Using TOPSIS Method Under Fuzzy Environment. Application in Spillway Selection. *Proceedings* **2018**, *2*, 637.
66. Gupta, H.; Vahid Dastjerdi, A.; Ghosh, S.K.; Buyya, R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw. Pract. Exp.* **2017**, *47*, 1275–1296.
67. Kass. Programming Blockchain. Available online: https://medium.com/programmers-blockchain (accessed on 19 April 2018).
68. Quinlan, R. Data Mining Tools See5 and C5.0. 2004. Available online: https://topepo.github.io/C5.0/reference/C5.0.html (accessed on 21 April 2018).
69. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA 2009), Ottawa, ON, Canada, 8–10 July 2009; pp. 1–6.
70. Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I.H. The WEKA data mining software: An update. *ACM SIGKDD Explor. Newsl.* **2009**, *11*, 10–18.
71. Buterin, V. A next-generation smart contract and decentralized application platform. *White Pap.* **2014**, *3*, 37.
72. Gaži, P.; Kiayias, A.; Russell, A. Stake-bleeding attacks on proof-of-stake blockchains. In Proceedings of the 2018 Crypto Valley Conference on Blockchain Technology (CVCBT), Zug, Switzerland, 20–22 June 2018; pp. 85–92.
73. Wei, P.; Yuan, Q.; Zheng, Y. Security of the Blockchain Against Long Delay Attack. In Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, Australia, 2–6 December 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 250–275.