




Article

Distributed Secure Edge Computing Architecture Based on Blockchain for Real-Time Data Integrity in IoT Environments

Rongxu Xu ¹, Lei Hang ², Wenquan Jin ³ and Dohyeun Kim ^{1,*}¹ Department of Computer Engineering, Jeju National University, Jeju 63243, Korea; rongxu@jejunu.ac.kr² Tianhua College, Shanghai Normal University, Shanghai 201815, China; hanglei@jejunu.ac.kr³ Big Data Research Center, Jeju National University, Jeju 63243, Korea; wenquan.jin@jejunu.ac.kr

* Correspondence: kimdh@jejunu.ac.kr

Abstract: The traditional cloud-based Internet of Things (IoT) architecture places extremely high demands on computers and storage on cloud servers. At the same time, the strong dependence on centralized servers causes major trust problems. Blockchain provides immutability, transparency, and data encryption based on safety to solve these problems of the IoT. In this paper, we present a distributed secure edge computing architecture using multiple data storages and blockchain agents for the real-time context data integrity in the IoT environment. The proposed distributed secure edge computing architecture provides reliable access and an unlimited repository for scalable and secure transactions. The architecture eliminates traditional centralized servers using an edge computing framework that represents cloud computing for computer and security issues. Also, blockchain-based edge computing-compatible IoT design is supported to achieve the level of security and scalability required for data integrity. Furthermore, we present the blockchain agent to provide internetworking between blockchain networks and edge computing. For experimenting with the proposed architecture in the IoT environment, we implement and perform a concrete IoT environment based on the EdgeX framework and Hyperledger Fabric. The evaluation results are collected by measuring the performance of the edge computing and blockchain platform based on service execution time to verify the proposed architecture in the IoT environment.

Keywords: Internet of Things; blockchain; edge computing; data integrity; EdgeX; hyperledger fabric



Citation: Xu, R.; Hang, L.; Jin, W.; Kim, D. Distributed Secure Edge Computing Architecture Based on Blockchain for Real-Time Data Integrity in IoT Environments. *Actuators* **2021**, *10*, 197. <https://doi.org/10.3390/act10080197>

Academic Editor: Kash Khorasani

Received: 30 June 2021

Accepted: 11 August 2021

Published: 13 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The IoT is an ever-present network of intelligent and network-available objects (called “things”) and people. Every “thing” can be connected and communicated with via the IoT, transforming the physical world into the virtual world which is a huge information system. Various technologies such as machine learning, cloud computing, and data analysis quickly become part of the IoT architecture [1]. As IoT devices and their visibility on the Internet continue to increase, security (that is, access to resources by legitimate users) has become a major issue. On the one hand, the omnipresence of the IoT promotes creative applications for end-users, on the other hand, the vulnerability of safety measures can lead to serious problems such as personal injury and burglary. Security has another aspect, the ‘privacy problem’, companies that manage confidential user data centrally can use it illegally, leading to data protection violations [2]. Current IoT security efforts are aimed at ensuring the security of point-to-point communication and security issues cannot be resolved during the data life cycle (for example, secure sharing and verifiable access control). The cloud-based model in the IoT ignores the location of data and requires centralization by reliable third parties [3]. Most current IoT architectures are centralized architecture that connect with a cloud server to provide efficient resources.

As IoT systems become more and more complex, this solution offers powerful functions for elastic computing and data management. However, there are still various security

problems, such as a single point of failure caused by a rapidly growing IoT-based infrastructure that could jeopardize the effectiveness of the entire data center. For a large number of IoT devices, tamper-proof environments and fault-tolerant networks must be implemented [4]. The IoT and blockchain are two technologies that have become popular since their inception. Blockchain has become a popular technology that uses community authentication to synchronize the ledger content between multiple users [5]. It works as a decentralized ledger to check and save transaction records. The performance of the blockchain is better than the corresponding method, which is based on a central digital ledger. In the near future, the IoT will affect almost all the things we use daily. With the increasing use of this technology, the risk of abuse increases. The existing technology is not sufficient to solve this problem. That is why blockchain is an effective technology to solve the problem of security issues related to IoT [6]. The integrated cryptocurrency function in the basic blockchain technology enables the implementation of an autonomous, self-sufficient, decentralized storage ecosystem in which storage nodes are rewarded for the delivery of storage and bandwidth and, above all, following the protocol correctly [7]. With this automatic transaction exchange in the blockchain, an intermediary's intervention is not required, so that the transition from central management to distributed management is possible.

Edge computing is an extension of the cloud that is introduced at the edge of the network to allocate resources and services [8]. Subscribers will receive a multi-access environment that allows them to use cloud-like functions to improve computer, application, and storage services [9]. Edge computing enables real-time computing and communication through the use of nearby Edge servers. Many companies such as Intel, Amazon, and Cisco have developed advanced services to facilitate the development of the IoT. An edge device can be any computer source that is located between a data source and the cloud [10].

IoT devices have limited resources, and the demand for data safety is increasing day by day with the development of the IoT. Cloud computing can be a suitable solution to the problem of resource scarcity of IoT devices. However, due to the centralized structure and dependence on network resources, the response to time-limited services is insufficient. Also, a centralized structure is not a solution to the problem of data reliability [11]. In contrast, edge computing implements the cloud from a remote data center to an IoT network terminal, replenishing IoT resources as well as solving network latency [12]. Blockchain is a technology that stores and manages ledger data in multiple computers connected to the network rather than a central server through a p2p network. Blockchain guarantees immutability and security of stored data using distributed processing and encryption technology [13]. The integration of edge computing and blockchain in a system has become a natural trend [14]. By integrating the blockchain network into the edge computing environment, the system could present reliable control and access over the network, repository, and computing via distributed edge devices. As a result, the security of the network, integrity of data, and the computing power of the application can be significantly advanced. Moreover, the combination of blockchain and edge computing enables the distribution of a large number of computing and repository resources at the edge computing environment, effectively reducing blockchain storage and mining computing load on performance-related devices. External computers can perform scalable storage and calculations in the blockchain.

In this paper, we propose a distributed secure edge computing architecture using multiple data repositories that combine blockchain platforms and edge computing frameworks for ensuring the safety and integrity of IoT data. In the proposed IoT environment, reliable access and an unlimited repository are proposed to handle scalable and secure transactions. Edge computing brings cloud computing close to the environment, overcoming the lack of delays experienced with traditional centralized servers. Moreover, based on the blockchain, edge computing achieves the level of security and scalability required for data integration in IoT environments. For implementing the proposed IoT architecture, the EdgeX framework and Hyperledger Fabric are integrated to present a concrete IoT

environment and performance based on collecting the service execution time. The EdgeX framework provides microservices using standalone servers to enable loose-coupling based on minimizing the dependencies between the implementation of functions. Hyperledger Fabric is an open-source framework to build a blockchain network that interacting with EdgeX-based edge computing to provide secure data integrity in the IoT environment.

We provide the following contributions:

1. We designed a secure edge computing architecture comprising a blockchain network and edge computing technology.
2. We developed a blockchain agent to integrate edge computing and blockchain technology into the IoT environment.
3. We ensured data integrity by recording data to the distributed blockchain ledger.
4. We overcome the problem of delays by using edge computing.

The rest of the paper is structured as follows. Section 2 reviews the existing approaches for the integration of blockchain and edge computing, and related technologies of the proposed edge computing architecture. Section 3 presents the proposed blockchain-based distributed secure edge computing architecture. Section 4 presents the functional details and service scenarios of the proposed edge computing architecture. Section 5 presents the implementation details of the real-time data integrity scenario in the IoT environment using proposed blockchain-based distributed secure edge computing. Section 6 evaluates the performance of the proposed edge computing architecture. Finally, we conclude this paper and introduce our future directions in Section 7.

2. Related Works

Blockchain was originally used to solve the dual spending problem in Bitcoin [15] but has grown over time to support other solutions like smart grids, delivery networks, healthcare, and logistics systems. However, there is a drawback in that it has limited ability to scale and process frequency-intensive tasks [16]. In the blockchain, transactions are recorded as blocks and their relationships are logically built up as linked lists of transaction blocks [17]. There is a consensus scheme that upgrades in data blocks affect the whole blockchain network. It forms a fraud-proof platform for recording and sharing data [18]. The authors of [19] and [20] exploited blockchain as a secure access point and repository of the virtual resources of managed devices. The authors proposed a blockchain-based architecture for IoT virtual resources on fog nodes. To manage the configuration of a broad and diverse set of device information, the virtualized IoT components' metadata is stored in the blockchain blocks in an encrypted manner. At the same time, different customers registered in the authorization-based blockchain can define and supply their own virtual systems and read or write blocks. Similarly, the author of [21] introduces a blockchain-based design for the IoT that makes distributed connection control and data governance possible. The design is tailored to IoT data traffic and makes secure data exchange possible. The blockchain is used as a verifiable and distributed connection management layer of the repository layer to achieve safe and flexible access management.

Blockchain technology can be divided into two types: public and permissioned networks [22]. Bitcoin [23] is a public blockchain with decentralized architecture, any participant can access the data stored on it to operate read and write actions. To keep all participants in a trusted state even malicious, it uses a memory-heavy consensus algorithm. Ethereum [24] provides not only the monetary function of Bitcoin but also the ability to program on the blockchain by supporting smart contracts. The consensus mechanism is POW (Proof of Work) where all the connected nodes involve to validate a ledger. There are some studies that utilize the public blockchain platform network as a distributed secure storage. The authors of [25] proposed a distributed data storage system with blockchain and certificate-free cryptography. This solution eliminates traditional centralized services by exploiting blockchain miners who perform "transactional" verification and record audits using non-encrypted cryptography. For example, in IoT applications such as implantable medical systems, data can be recorded in a distributed hash table (DHT) and references

to DHT memory addresses in the blockchain. When an entity requests to get information from the DHT, the blockchain concludes whether to allow the connection. The applicant identity verification is conducted by a blockchain miner instead of a centralized server, which provides the following benefits such as the absence of a central trusted server, decentralized repository, and traceability. The authors of [26] proposed a framework for data integrity services based on blockchain. This framework provides data owners and data consumers with a more reliable check of data integrity without relying on external auditors (TPAs). To ensure that the result of the system proposed in [27] is credible and will not be tampered with, the blockchain network was employed.

Hyperledger Fabric is a permissioned blockchain network platform. As a permissioned blockchain, only authorized participants can participate in the network. Since only some nodes run smart contracts, multiple transactions can be quickly processed in parallel, and enable interchangeable consensus protocols to accelerate the processing of the blockchain [28]. Some studies attempt to integrate permissioned blockchain networks with the IoT in a different domain to use the features of the blockchain. Blockchain provides data with immutability and security [29]. In vehicular networks, the author of [30] proposes a reputation-based data exchange scheme to ensure high-quality data exchange between vehicles and uses the consortium blockchain to set up a secure and distributed vehicle blockchain for data management. To achieve safe and efficient data storage on the roadside unit (RSU) and data exchange between vehicles, the author implemented an intelligent contract for the vehicle blockchain. Additionally, there is a study that utilizes the blockchain as a distributed control system. In a smart home, the author of [31] adds blockchain to a smart home to control the responsibilities of the transactions, conveying a high sense of responsibility and not missing transactions. To improve the performance of challenge-based collaborative intrusion detection networks (CIDNs), the authors of [32] and [33] chose blockchain technology as a solution. CIDNs were applied to the IoT environment to secure the IoT network [34]. The immutability of blockchain data allowed it to be used to verify the shared signatures to deal with insider attacks. Permissioned blockchains were responsible for network safety by providing an authentication service. The authors of [35] proposed a distributed control system based on blockchain for edge computing. The proposed system architecture consisted of an IoT devices layer, an edge computing layer, and a cloud service layer. The edge devices were prepared with Docker containers of Hyperledger Fabric that operate smart contracts to provide security and validation abilities to transactions.

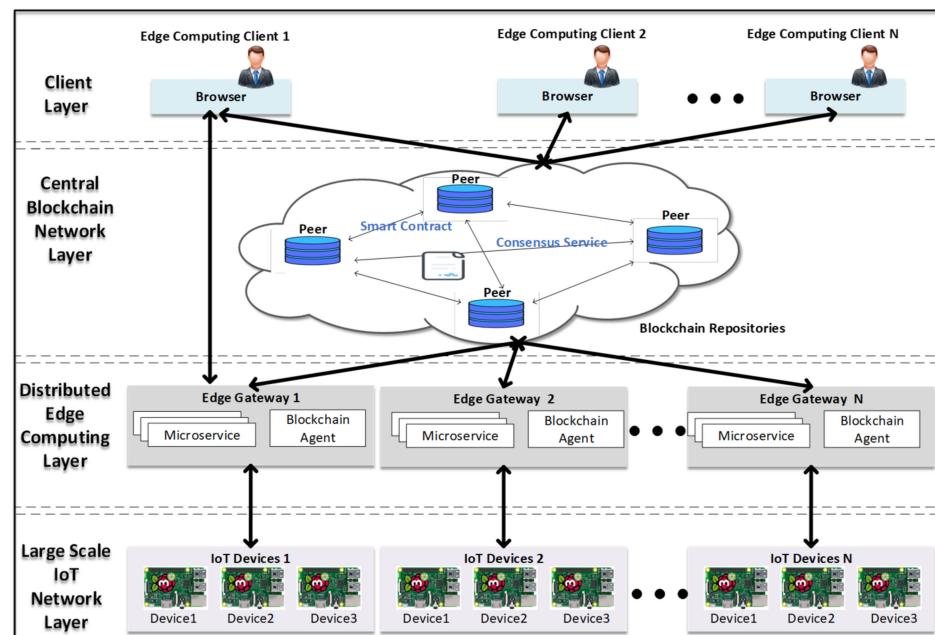
Table 1 provides a comparison of blockchain platforms based on technical factors for Implementing distributed applications. Ethereum and Bitcoin enable access to any user in the network without permission. Conversely, Hyperledger Fabric only allows authorized participants to connect to the network to improve the security of data. While Bitcoin is specialized in virtual currency functions, Ethereum and Hyperledger Fabric can implement business logic through smart contracts. While Ethereum uses Solidity programming, Hyperledger Fabric supports several high-level languages to developing smart contracts. Looking at the consensus algorithm, Hyperledger Fabric supports several algorithms to make them pluggable, but the other two platforms use pow, which is dependent on computing resources. With the features of Hyperledger Fabric that is suitable to provide security and integrity to the IoT environment.

Table 1. Blockchain platforms Comparison.

Features	Hyperledger Fabric	Bitcoin	Ethereum
Network	Permissioned or private	Permissionless and public	Permissionless, public or private
Consensus	Pluggable	PoW	PoW
Smart Contract	Support with higher-level programming language	No	Support with Solidity programming
Transaction Cost	Low	High	High

3. Blockchain-Based Distributed Secure Edge Computing Architecture

The proposed blockchain-based distributed secure edge computing is comprised of clients, blockchain networks, edge computing, and IoT devices. Through the interaction between the entities, real-time data integrity is provided in the IoT environment. Figure 1 presents the layered IoT architecture including a layer of clients, central blockchain network, distributed edge computing, and large-scale IoT devices.

**Figure 1.** Proposed IoT architecture based on distributed secure edge computing.

In the client layer, the clients can be any terminal devices with web browser features, through which clients could read from or write transactions to the blockchain network. The client application provides the user interfaces for interacting with users to access the proposed system. In the blockchain network, the transactions are generated by requests from the client.

The central blockchain repository layer provides a sensing data repository. The blockchain network consists of peers, smart contracts. Each peer is the network component where copies of the blockchain ledger are hosted. The ledger is the storage of the blockchain network. To support consistent information updates and enable ledger functions (transactions, inquiries, etc.), the blockchain network uses smart contracts to provide controlled access to the ledger.

The distributed edge computing layer includes various fine-meshed and independent microservices with independent functions based on EdgeX Foundry. Microservices can be independently developed using separate technologies and implemented in containers that provide portability, interchangeability, scalability, and simple execution environments.

The advanced computer framework breaks down centralized cloud computing services into decentralized software modules with minimal functionality and standalone services such as microservices for basic services, device connections, data repositories, and other application microservices to provide storage and information. The microservices communicate with each other via well-defined messaging interfaces such as Representational State Transfer (REST) Application Programming Interface (API). From the perspective of a service consumer, an edge computing framework can be seen as a collection of APIs.

The large-scale IoT devices layer consists of various small single-board computers, acquisition devices, and actuators. IoT devices supplied in the IoT infrastructure can collect data from the physical world and execute actuators to operate electrical IoT devices. In general, actuators and sensors have no network functions. That is why they make a direct connection to IoT devices such as single-board computers through native interfaces. IoT devices can provide wireless or wired connectivity for communication with edge computing frameworks via IoT data transmission protocols.

Figure 2 presents the proposed distributed secure edge. The client role can be defined as general client or administrator. Once the administrator configures the device and device manager using the blockchain platform, the general client can access the secure IoT data. The central blockchain network layer includes the blockchain network and the REST server that exposes the blockchain network functions to the Internet. The transactions and wallet are represented by the REST APIs that provide the functions to manage and access the transactions. Using the REST APIs, the participants, assets, and transactions are handled in the blockchain network.

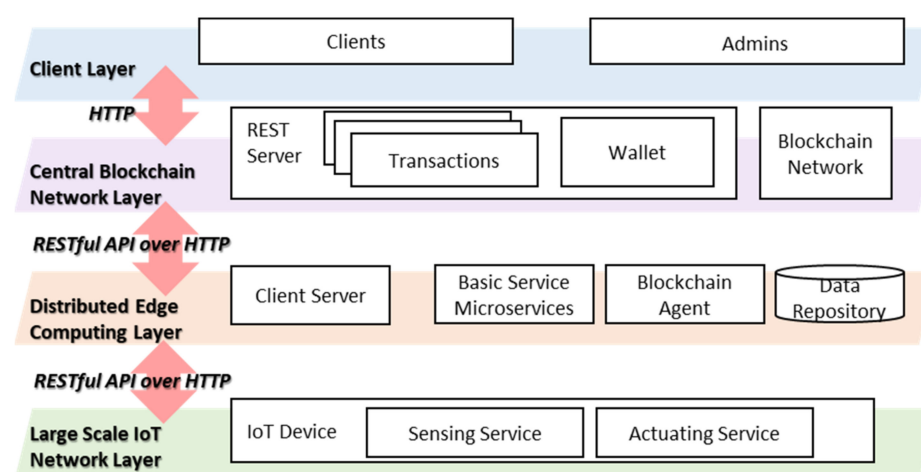


Figure 2. Distributed secure edge computing based on blockchain.

The distributed edge computing layer consists of various independent microservices. The client server is used for forwarding the request to the edge computing. The basic service microservices provide the function to manage devices and data in the edge computing. For integration into the blockchain network, the blockchain agent is designed as the microservice provider. Blockchain agent receives orders from the edge computing services and sends transactions to the blockchain network. In the IoT network, the IoT device includes sensors and actuators to provide services for collecting environmental data and changing environmental parameters. The data collection is performed in the IoT network layer where the IoT devices deliver the sensing data and actuator statuses to the edge computing platform. The data is stored in the data repository and the blockchain agent sends the data to the blockchain network for enabling data integrity in the IoT environment.

4. Proposed Distributed Secure Edge Computing for Real-Time Data Integrity

Data integrity is important in the data collection and storage in an IoT environment to protect the privacy of IoT devices. The proposed distributed secure edge computing enables real-time data integrity based on the blockchain network layer between clients and

edge computing layer. Through the interactions between the edge computing platform and blockchain platform, the IoT data is protected throughout its lifetime. To achieve the proposed real-time data integrity in the IoT environment, the device and data are managed on both sides of the edge computing and blockchain platforms.

Figure 3 presents the functional architecture of the proposed blockchain-based distributed secure edge computing. The edge computing platform includes a blockchain agent based on microservices to provide a secure connection from the blockchain network. The device connectors provide connections with IoT devices. The blockchain agent uses the APIs that are provided by the blockchain platform based on the REST server. The REST server provides APIs for registering device manager, device, and saving sensing data.

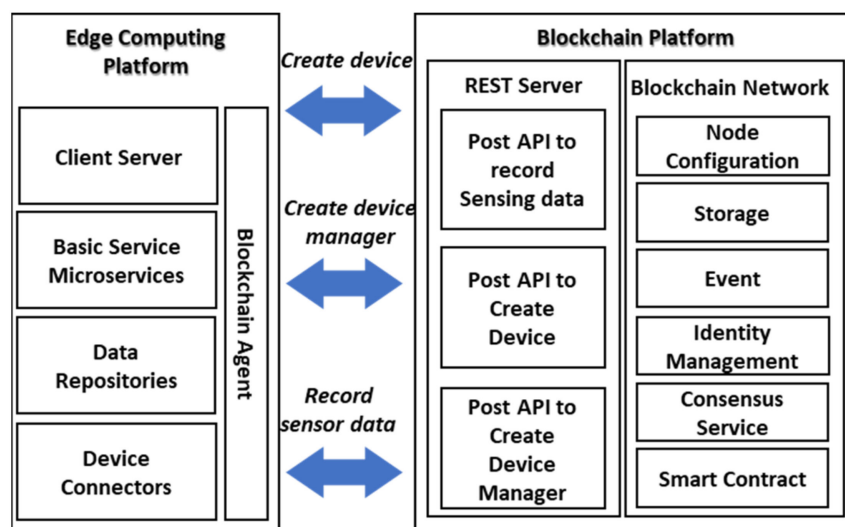


Figure 3. Blockchain-based distributed secure edge computing functional architecture.

Figure 4 shows the configuration process for the device and device manager in the distributed secure edge computing. Before starting the proposed system, client information and device information must be registered. To register clients, the administrator requests client information from the client-server that responds to client management and authentication. The response creates a transaction that registers client participants on the blockchain network using a blockchain agent. Similarly, the device registration first collects registered information from the core metadata of EdgeX. Finally, a related transaction is created through the blockchain agent, then it is registered on the blockchain network.

Figure 5 shows the data collection process based on the proposed real-time data integrity in an IoT environment. For sensing data from the IoT device, the client requests the EdgeX framework to get the sensing data. Within the EdgeX framework, the core command microservice exposes commands to each device sensors. When the client selects a command and sends a request to the core command, then the core command sends the command to the device service on behalf of the client, and finally, the device service will request the corresponding IoT device to send sensing data. When the IoT device responds with sensing values the device service requests the blockchain agent to generate a transaction to save the sensing values in related assets, and to send back the value to the client.

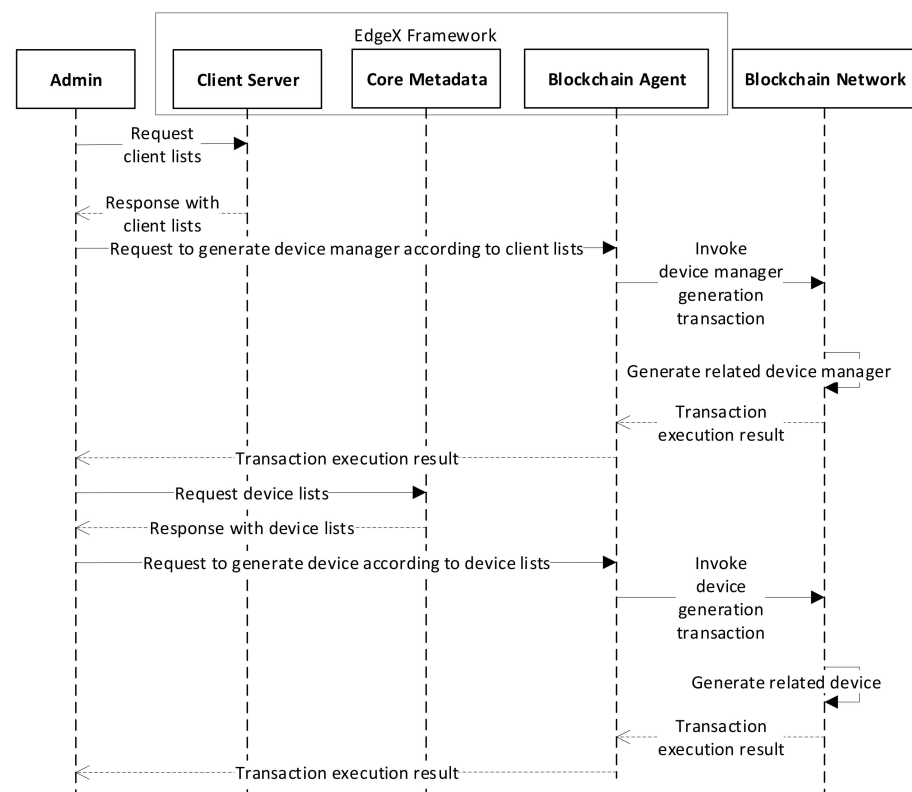


Figure 4. Configuration process for distributed secure edge computing based on blockchain.

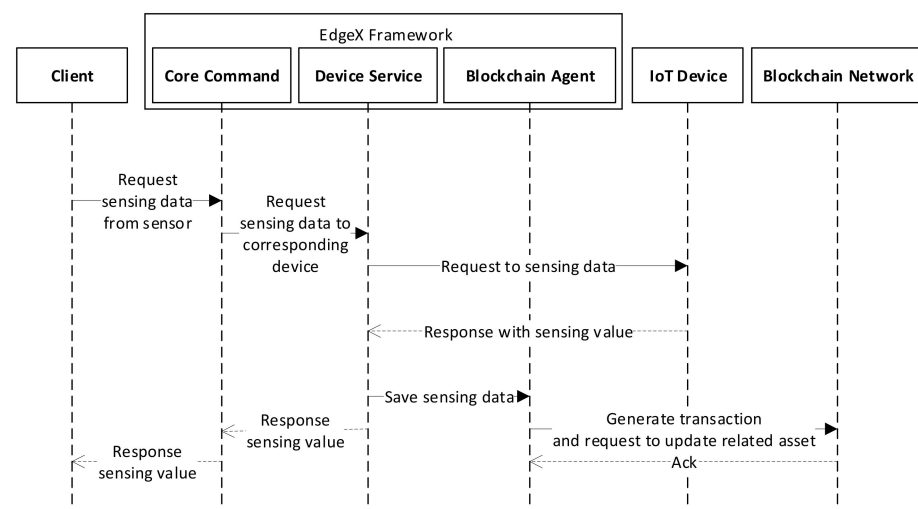


Figure 5. Data collection process using distributed secure edge computing.

5. Implementation Details

The proposed distributed secure edge computing architecture comprises an edge computing platform, blockchain platform, and device platform. Table 2 presents the development environment of the proposed architecture that is used for provide the experimental results and performance.

Table 2. Development environment.

Entity	Hardware	Operating System	Library and Framework
Edge Computing Platform	PC (Quarter core of AMD 64)	Ubuntu 18.04 Desktop	EdgeX Framework
Blockchain Platform	PC (Quarter core of AMD 64)	Ubuntu 18.04 Desktop	Spring Boot
Device Platform	Raspberry Pi 3 Model B (Quad-core 64-bit ARM 1.2 GHz)	Ubuntu Meta 16	Hyperledger Fabric

The edge computing platform is developed based on the EdgeX framework that provides microservices in a Dockerized manner that packages all the code and dependencies of the application in a standard format allowing it to run quickly and reliably across computing environments. We arranged an executing environment with the following hardware: desktop with Ubuntu 18.04 operating system, a quarter core of an AMD 64, 4 GB memory, and 100 GB hard disk. To operate EdgeX framework services the tools Docker and Docker Compose were installed. The blockchain network was hosted on an Ubuntu 18.04 operating system. The operating system was operated on hardware including a quarter core of an AMD 64 CPU, 4 GB memory, and 100 GB hard disk. In software, we used the open-source framework Hyperledger Fabric to build the blockchain network. The sensing data was provided by the BMP280 humidity and temperature sensor. The sensor directly connects with a Raspberry Pi including an Ubuntu MATE operating system, 1 gigabyte of memory, 16 gigabytes of micro-SD card disk, and software of JDK 8 and Spring Boot framework. To control and manage the data received from the sensor an HTTP server is executed on this IoT device.

Figure 6 presents the source code of the blockchain agent to connect the blockchain network with EdgeX microservices. The blockchain agent provides several APIs for connecting with the blockchain network. Based on the agent, the device manager is generated that is the participant of the blockchain network and role of the EdgeX client. Also, the device is generated that is the asset of the device manager to save sensing data through the blockchain transaction.

```

1  @RequestMapping(value="/api/generate.new.devicemanager", method={RequestMethod.POST})
2  public boolean generateNewDeviceManager(@RequestBody NewDeviceManager deviceService) {
3      log.info("request body is "+deviceService.toString());
4      log.info("invoke devicemanager generation function");
5      return handler.generateNewDeviceManager(deviceService);
6  }
7  @RequestMapping(value="/api/generate.new.sensordevice", method={RequestMethod.POST})
8  public boolean generateNewSensorDevice(@RequestBody NewSensorDevice device) {
9      return handler.generateNewSensorDevice(device);
10 }
11 @RequestMapping(value="/api/save.sensingdata", method={RequestMethod.POST})
12 public boolean saveSensingData(@RequestBody SaveSensingData data) {
13     return handler.saveSensingData(data);
14 }
15 @RequestMapping(value="/api/save.token", method={RequestMethod.POST})
16 public String saveToken(@RequestBody String token) {
17     return repo.saveToken(token).getStartTime().toString();
18 }
19 @RequestMapping(value="/api/secure.token", method={RequestMethod.POST})
20 public String secureWithToken(@RequestBody NewBlockchainToken token) {
21     bcToken.setSecureWithToken(token.isEnabled());
22     bcToken.setToken(token.getToken());
23     return bcToken.toString();
24 }

```

Figure 6. Source code of blockchain agent.

Figure 7 presents the source code of data collection based on the blockchain. Once a client sends a command to the IoT device through the EdgeX microservices with sensing data, the data will be saved to the blockchain network of assets managed by the client. The

command comprises the device id which is the id of an asset of a device in the blockchain network, user id which is the id of the device manager, sensing time, and sensing value.

```

1 public String sendCommand(String deviceId, String commandId, String method,
2   String parameters, String token, String user) {
3   String responseBody = "";
4   String uri = "http://" + MyUtil.URI_SERVER + ":8000/command/api/v1/device/"
5   + deviceId + "/command/" + commandId + "?jwt=" + token;
6   if (method.equals("get")) {
7     List<NameValuePair> params = new ArrayList<NameValuePair>();
8     responseBody = new HTTPClient().request(uri, method, params);
9     JSONObject jsonObject = new JSONObject(responseBody);
10    String value = jsonObject.getString("tmp").isEmpty() == true ? "0" : jsonObject.getString("tmp");
11    Date date = new Date();
12    String time = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss").format(date);
13    String sensingData = "{ \"value\": \"\" + value + \"\", \"sensingtime\": \"\" + time
14    + \"\", \"sensor\": \"\" + deviceId + \"\", \"deviceManager\": \"\" + user + \"\" }";
15    String result = new HTTPClient().requestPostWithJson(blockchainAgent, sensingData);
16  } else if (method.equals("put")) {
17    CloseableHttpClient client = HttpClients.createDefault();
18    HttpPut httpPut = new HttpPut(uri);
19    StringEntity entity = new StringEntity(parameters);
20    httpPut.setEntity(entity);
21    httpPut.setHeader("Accept", "application/json");
22    httpPut.setHeader("Content-type", "application/json");
23    CloseableHttpResponse response = client.execute(httpPut);
24    BufferedReader bufferedReader
25    = new BufferedReader(new InputStreamReader(response.getEntity().getContent()));
26    StringBuffer result = new StringBuffer();
27    String line = "";
28    while ((line = bufferedReader.readLine()) != null) {
29      result.append(line);
30    }
31    responseBody = result.toString();
32  }
33 }

```

Figure 7. Source code of data collection based on blockchain.

Figure 8 presents the source code of the blockchain transaction operating function that configures the transaction logic for the smart contract. The function changes the information of the sensor asset with sensing data. Moreover, the function updates the information of the sensor asset with the data presented by the related physical device, then updates the sensor asset in the repository of the blockchain network.

```

1 function SensorReading(tx) {
2   let id = tx.sensor.deviceId;
3   let factory = getFactory();
4   let assetRegistry = await getAssetRegistry('org.mcl.edgecomputing.Sensor');
5   let sensor = await assetRegistry.get(id);
6   if (!sensor.data) {
7     sensor.data = factory.newConcept('org.mcl.edgecomputing', 'Reading');
8     sensor.data[0].value = tx.value;
9     sensor.data[0].timestamp = tx.timestamp;
10  } else {
11    newData = factory.newConcept('org.mcl.edgecomputing', 'Reading');
12    newData.value = tx.value;
13    newData.timestamp = tx.timestamp;
14    sensor.data.push(newData);
15  }
16  await assetRegistry.update(sensor);
17  var event = factory.newEvent('org.mcl.edgecomputing', 'SensorEvent');
18  event.sensorId = id;
19  event.managerId = tx.deviceManager.managerId;
20  event.msg = 'The sensor(' + id + ') ' + 'sensing the value: '
21  + tx.value.toString() + ' ' + sensor.unit;
22  emit(event);
23 }

```

Figure 8. Source code of blockchain transaction operating function.

Table 3 presents the configuration details of the device asset and participant for the smart contracts in blockchain networks. Based on the Hyperledger Composer, a comprehensive and open implementation tool framework is supported to promote the development of blockchain applications for designing and implementing the smart contract. The participants are members of our proposed blockchain network, which allows us to generate assets and publish transactions. In the proposed distributed secure edge computing, the device management is provided to contact and manage IoT devices. The assets can be the goods, services, or properties stored in registries. The asset and participant are generated with the identifier and include any other properties. The asset of a device is the representation of the IoT device sensor that contains general properties such as the id of the device, name manager of the realistic device in the edge computing environment.

Table 3. Device asset and participant configuration in smart contract.

Model Type	Category	Component	Type
Asset	Sensor	sensorId	String
		name	String
		deviceManager	Object
		timestamp	DateTime
Participant	Device Manager	value	Double
		managerId	String
		name	String

Table 4 shows the transaction configuration in the smart contract. To interact with assets, we define some transactions that are part of the smart contract. It also can interact with participants, each of which has an identity across multiple blockchain networks. There are several user management operations such as create, read, update, and delete and all the operation is implemented by the conditions in the blockchain network. According to the table below, the device manager of the device is permitted to operate the update action on the instance of the device asset.

Table 4. Transaction configuration in smart contract.

Component	Type	Participant	Condition
Sensor reading	Transaction	Device Manager	Asset = Device Manager
Device creating	Transaction	Device Manager	Participant = Device Manager

Table 5 presents the details of REST APIs that are provided by the blockchain agent for connecting the EdgeX and the blockchain network. Each REST API has a media type that specifies state transition data elements, a base URI, and a method such as GET, POST, PUT, DELETE. Where the URI is the path of a specific datum, and the desired action in the identified resource along with the request is represented by the method. For example, a request “/API/sensor” with method GET would return sensor information as a list, while a request with method POST would command the server to accept the entity included in the request.

Table 5. REST APIs for data collection from distributed secure edge computing.

URI	Method	Media Type	Action
/api/sensor	ALL	Application/json	Device Manager Management
/api/sensorReading	ALL	Application/json	Device Manager Management

Figure 9 shows the implementation result of the transaction log history in the blockchain network by a time sequence. Once a transaction is submitted, it will enter the log along with an inconvertible blockchain ledger record time in which the entry type presents the transaction type submitted by the client. The admin participant adds a participant and an asset to the network, then collecting sensing data from IoT devices to the asset of the blockchain network.

Date, Time	Entry Type	Participant	
2020-02-12, 22:52:58	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:52:54	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:52:50	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:52:45	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:52:41	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:49:22	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:49:15	SensorReading	admin (NetworkAdmin)	view record
2020-02-12, 22:44:41	AddAsset	admin (NetworkAdmin)	view record
2020-02-12, 22:42:35	AddParticipant	admin (NetworkAdmin)	view record

Figure 9. Implementation result of transaction list.

Figure 10 shows the implementation result of device information that is registered through the EdgeX framework. The device information is comprised of multiple properties such as id, name, description, address, profile name, and service name, where the id is an identifier used to distinguish the device from other devices registered in the EdgeX framework and the service name is the name of the device service microservice of the EdgeX framework which is used to connect with IoT devices.

Device General Info
[Go Device Resource](#)

ID	Name	Description
5d67af869f8fc200014ef4af	device-v1	http example
Address	Profile Name	Service Name
JNU ENG 4	device.service.http.v1.yaml	device-http-test

Figure 10. Implementation result of device information.

Figure 11 shows the implementation result of collected IoT data that is an asset of the device in the blockchain network. The id and name are the same as the id of the device registered in the EdgeX framework. The data will be saved in the data filed with a timestamp, and value. As can be seen from the above figure, user02 has a sensor named device-v1 and reads data with a value of 0.

ID	Data
5d67af869f8fc200014ef4af	<pre> { "\$class": "org.mcl.edgecomputing.Sensor", "data": [{ "\$class": "org.mcl.edgecomputing.Reading", "timestamp": "2020-02-11T12:50:49.334Z", "value": 0 }], "Type": "SENSOR", "deviceId": "5d67af869f8fc200014ef4af", "name": "device-v1", "deviceManager": "resource:org.mcl.edgecomputing.DeviceManager#user02" } </pre>

Figure 11. Implementation result of collected IoT data.

6. Performance Evaluation

To experiment with the proposed system, we carried out several experimental tests. We measured the execution time that it took for a transaction from the submission by the client until it was processed and written on the ledger. In the first experiment we measured the device manager registration latency time and the results are recorded in Figure 12. For this evaluation, three groups of 100, 200, and 300 tests were executed. The execution times of the proposed system were collected with minimum, average, and maximum times recorded.

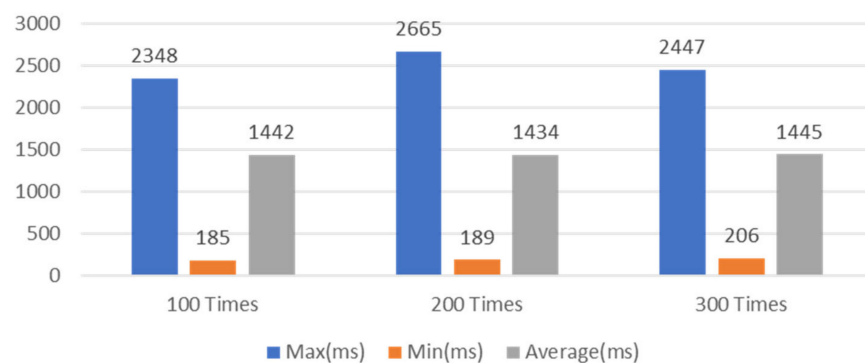


Figure 12. Data manager registration latency.

For the 100-times dataset, the minimum latency time was 163 ms, averaging at 1410 ms, and the maximum time was 2509 ms. For the 200-times set, the minimum time was 165 ms, the average was 1360 ms, and the maximum time was 2289 ms. For the 300-times set, the minimum time was 153 ms, the average was 1367 ms, and the maximum time was 2336 ms. According to the latency times for device manager registration, we can see that the latency time was not affected much by the number of tests.

The second study was the IoT device registration latency time and the results are shown in Figure 13. For the 100-times set, the minimum time was 185 ms, averaging at 1442 ms, and the maximum time was 2348 ms. For the 200-times set, the minimum time was 189 ms, averaging at 1434 ms, and the maximum time was 2665 ms. For the 300-times set, the minimum time was 206 ms, averaging at 1445 ms, and the maximum time was 2447 ms. From these results it can be seen that the latency of IoT device registration is stable regardless of the number of tests.

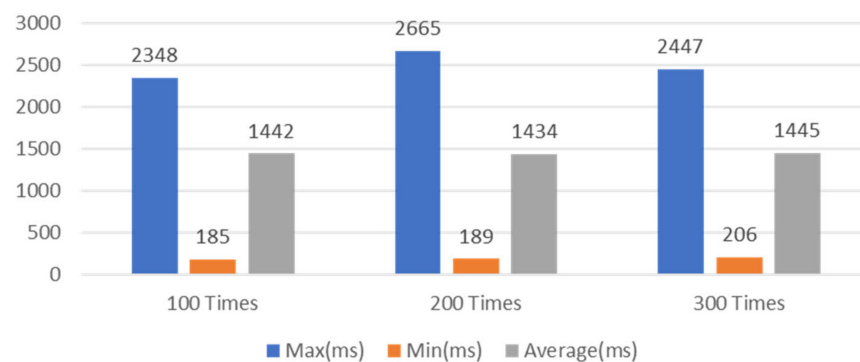


Figure 13. Device registration latency.

A test of the service execution time on sensing data storage was also conducted and the results are illustrated in Figure 14. For the 100-times set, the minimum time was 2134 ms, averaging at 2226 ms, and the maximum time was 2336 ms. For the 200-times set, the minimum time was 2294 ms, averaging at 2178 ms, and the maximum time was 2454 ms. For the 300-times set, the minimum time was 2232 ms, averaging at 2390 ms, and the maximum time was 2447 ms. From these results it can be seen that the delay time did not change significantly even when the number of transactions was increased to 300 times.

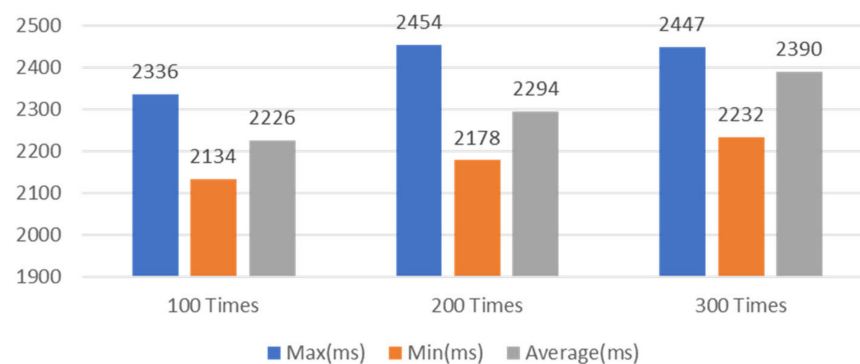


Figure 14. Data collection latency.

Table 6 provides resource utilization data of our implemented blockchain network. The measured results include memory usage and central processing unit (CPU) utilization rate through the tested blockchain network. The nodes hosting the peer used 261.6 and 273.4 MB of memory respectively and showed 5.64% and 3.51% CPU usage. On the other hand, in the node where DB was running, the memory usage was less than 100 MB, but the CPU was used a lot at 24.27%. The rest of the nodes showed significantly lower usage. Looking at the results, the CPU and memory usage was low, which will help the service to run safely.

Table 6. Analysis of Resource utilization for Blockchain based Edge Computing Network.

Type	Name	Memory	CPU
Docker	Peer0.org1	261.6 MB	5.64%
Docker	Peer0.org2	273.4 MB	3.51%
Docker	Couchdb	96.7 MB	24.27%
Docker	Ca1	6.0 MB	0.00%
Docker	Ca2	5.8 MB	0.00%
Docker	Orderer	9.3 MB	0.50%

7. Conclusions and Future Directions

We proposed a distributed secure edge computing architecture using multiple data repositories that combines blockchain platforms and edge computing frameworks to ensure

the safety and integrity of IoT data through the real-time data integrity. Using the proposed distributed secure edge computing architecture, the IoT data is protected throughout its lifetime based on reliable access and unlimited repository. The blockchain agent provides the internetworking function between blockchain networks and edge computing to enable the IoT data stored in the edge computing platform as well as the blockchain platform. For experimental purposes the proposed distributed secure edge computing, EdgeX framework and Hyperledger Fabric were integrated to present a concrete IoT environment, and performance was measured based on collecting service execution times. The integration of the EdgeX-based edge computing and Hyperledger Fabric-based blockchain network provides loose-coupling edge computing service scenarios and secure data integrity in the IoT environment. However, the data collection latency takes more than 2 s which needs to be reduced for processing real-time data immediately.

In the future we will apply an intelligent approach based on fuzzy- or machine learning to improve the performance of the blockchain network to reduce the data collection latency. This approach is expected to automatically control the transaction data flow in real-time based on network performance indices for solving transaction traffic congestion. Moreover, the proposed device manager will regulate the transaction traffic flow according to throughput and latency in the blockchain network.

Author Contributions: Writing—Original draft preparation, R.X., L.H., W.J. and D.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Energy Cloud R&D Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT (2019M3F2A1073387), and this work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2021-0-00188, Open source development and standardization for AI enabled IoT platforms and interworking), any correspondence related to this paper should be addressed to Dohyeun Kim.

Data Availability Statement: Data sharing not applicable.

Acknowledgments: This research was supported by Energy Cloud R&D Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT (2019M3F2A1073387), and this work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (2021-0-00188, Open source development and standardization for AI enabled IoT platforms and interworking), any correspondence related to this paper should be addressed to Dohyeun Kim.

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

References

1. Panarello, A.; Tapas, N.; Merlino, G.; Longo, F.; Puliafito, A. Blockchain and IoT integration: A systematic survey. *Sensors* **2018**, *18*, 2575. [CrossRef]
2. Novo, O. Blockchain meets IoT: An architecture for scalable access management in IoT. *IEEE Internet Things J.* **2018**, *5*, 1184–1195. [CrossRef]
3. Botta, A.; De Donato, W.; Persico, V.; Pescapé, A. Integration of cloud computing and internet of things: A survey. *Future Gener. Comput. Syst.* **2016**, *56*, 684–700. [CrossRef]
4. Hang, L.; Kim, D.-H. Design and implementation of an integrated IoT blockchain platform for sensing data integrity. *Sensors* **2019**, *19*, 2228. [CrossRef]
5. Kubendiran, M.; Singh, S.; Sangaiah, A.K. enhanced security framework for E-health systems using blockchain. *J. Inf. Process. Syst.* **2019**, *15*, 239–250.
6. Blockchain to Secure IoT Data. Available online: <https://www.geeksforgeeks.org/blockchain-to-secure-iot-data/> (accessed on 13 August 2021).
7. Casino, F.; Dasaklis, T.K.; Patsakis, C. A systematic literature review of blockchain-based applications: Current status, classification and open issues. *Telemat. Inform.* **2019**, *36*, 55–81. [CrossRef]
8. Satyanarayanan, M. The emergence of edge computing. *Computer* **2017**, *50*, 30–39. [CrossRef]
9. Nyamtiga, B.W.; Sicato, J.C.S.; Rathore, S.; Sung, Y.; Park, J.H. Blockchain-based secure storage management with edge computing for IoT. *Electronics* **2019**, *8*, 828. [CrossRef]

10. Sengupta, J.; Ruj, S.; Bit, S.D. A comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT. *J. Netw. Comput. Appl.* **2020**, *149*, 102481. [\[CrossRef\]](#)
11. Chiang, M.; Zhang, T. Fog and IoT: An overview of research opportunities. *IEEE Internet Things J.* **2016**, *3*, 854–864. [\[CrossRef\]](#)
12. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge computing: Vision and challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [\[CrossRef\]](#)
13. Iqbal, N.; Jamil, F.; Ahmad, S.; Kim, D. A novel blockchain-based integrity and reliable veterinary clinic information management system using predictive analytics for provisioning of quality health services. *IEEE Access* **2021**, *9*, 8069–8098. [\[CrossRef\]](#)
14. Yang, R.; Yu, F.R.; Si, P.; Yang, Z.; Zhang, Y. Integrated blockchain and edge computing systems: A survey, some research issues and challenges. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1508–1532. [\[CrossRef\]](#)
15. Kim, H.-W.; Jeong, Y.-S. Secure authentication-management human-centric Scheme for trusting personal resource information on mobile cloud computing with blockchain. *Hum. Cent. Comput. Inf. Sci.* **2018**, *8*, 11. [\[CrossRef\]](#)
16. Vukolić, M. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *International Workshop on Open Problems in Network Security*; Springer: Cham, Switzerland, 2015.
17. Xiong, Z.; Zhang, Y.; Niyato, D.; Wang, P.; Han, Z. When mobile blockchain meets edge computing. *IEEE Commun. Mag.* **2018**, *56*, 33–39. [\[CrossRef\]](#)
18. Feng, Q.; He, D.; Zeadally, S.; Khan, M.K.; Kumar, N. A survey on privacy protection in blockchain system. *J. Netw. Comput. Appl.* **2019**, *126*, 45–58. [\[CrossRef\]](#)
19. Samaniego, M.; Deters, R. Hosting virtual iot resources on edge-hosts with blockchain. In Proceedings of the 2016 IEEE International Conference on Computer and Information Technology (CIT), Nadi, Fiji, 8–10 December 2016.
20. Samaniego, M.; Deters, R. Virtual resources & blockchain for configuration management in IoT. *J. Ubiquitous Syst. Pervasive Netw.* **2017**, *9*, 1–13.
21. Shafagh, H.; Burkhalter, L.; Hithnawi, A.; Duquennoy, S. Towards blockchain-based auditable storage and sharing of iot data. In Proceedings of the 2017 on Cloud Computing Security Workshop, New York, NY, USA, 3 November 2017.
22. Li, D.; Wong, W.E.; Guo, J. A survey on blockchain for enterprise using hyperledger fabric and composer. In Proceedings of the 2019 6th International Conference on Dependable Systems and Their Applications (DSA), Harbin, China, 3–6 January 2020.
23. Urquhart, A. The inefficiency of bitcoin. *Econ. Lett.* **2016**, *148*, 80–82. [\[CrossRef\]](#)
24. Dannen, C. *Introducing Ethereum and Solidity*; Apress: Berkeley, CA, USA, 2017; Volume 318.
25. Li, R.; Song, T.; Mei, B.; Li, H.; Cheng, X.; Sun, L. Blockchain for large-scale internet of things data storage and protection. *IEEE Trans. Serv. Comput.* **2018**, *12*, 762–771. [\[CrossRef\]](#)
26. Liu, B.; Yu, X.L.; Chen, S.; Xu, X.; Zhu, L. Blockchain based data integrity service framework for IoT data. In Proceedings of the 2017 IEEE International Conference on Web Services (ICWS), Honolulu, HI, USA, 25–30 June 2017.
27. Cho, H.-H.; Wu, H.-T.; Lai, C.-F.; Shih, T.K.; Tseng, F.-H. Intelligent charging path planning for IoT network over blockchain-based edge architecture. *IEEE Internet Things J.* **2020**, *8*, 2379–2394. [\[CrossRef\]](#)
28. Sajana, P.; Sindhu, M.; Sethumadhavan, M. On blockchain applications: Hyperledger fabric and ethereum. *Int. J. Pure Appl. Math.* **2018**, *118*, 2965–2970.
29. Du, Y.; Wang, Z.; Leung, V. Blockchain-enabled edge intelligence for IoT: Background, emerging trends and open issues. *Future Internet* **2021**, *13*, 48. [\[CrossRef\]](#)
30. Kang, J.; Yu, R.; Huang, X.; Wu, M.; Maharjan, S.; Xie, S.; Zhang, Y. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet Things J.* **2018**, *6*, 4660–4670. [\[CrossRef\]](#)
31. Singh, S.; Ra, I.H.; Meng, W.; Kaur, M.; Cho, G.H. SH-BlockCC: A secure and efficient internet of things smart home architecture based on cloud computing and blockchain technology. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719844159. [\[CrossRef\]](#)
32. Li, W.; Wang, Y.; Li, J.; Au, M.H. Towards blockchain challenge-based collaborative intrusion detection. In *International Conference on Applied Cryptography and Network Security*; Springer: Cham, Switzerland, 2019.
33. Li, W.; Wang, Y.; Li, J.; Au, M.H. Toward a blockchain-based framework for challenge-based collaborative intrusion detection. *Int. J. Inf. Secur.* **2020**, *20*, 127–139. [\[CrossRef\]](#)
34. Li, W.; Tug, S.; Meng, W.; Wang, Y. Designing collaborative blockchain signature-based intrusion detection in IoT environments. *Futur. Gener. Comput. Syst.* **2019**, *96*, 481–489. [\[CrossRef\]](#)
35. Stanciu, A. Blockchain based distributed control system for edge computing. In Proceedings of the 2017 21st International Conference on Control Systems and Computer Science (CSCS), Bucharest, Romania, 29–31 May 2017.