A Blockchain Based New Secure Multi-Layer Network Model for Internet of Things

Cheng Li, Liang-Jie Zhang
National Engineering Research Center for Supporting Software for Enterprise Internet Services
Kingdee International Software Group Co., Ltd.
Shenzhen, China
zhanglj@ieee.org

***Abstract*—This paper presents a multi-layer secure IoT network model based on blockchain technology. The model reduces the difficulty of actual deployment of the blockchain technology by dividing the Internet of Things into a multi-level de-centric network and adopting the technology of block chain technology at all levels of the network, with the high security and credibility assurance of the blockchain technology retaining. It provides a wide-area networking solution of Internet of Things.**

Keywords-Internet of things, Blockchain, Network model, Security

I. INTRODUCTION

At present, the widely used Internet of Things (IoT) network model are usually in two styles as Fig. 1 illustrated: one is to use cloud platforms as the data processing centers to connect all things, such as Azure[1], PREDIX[2], Watson[3], etc.; the other is to use blockchain technology to support point-to-point network, like ADEPT[4], AllJoyn[5], Filament[6] and so on. Both models have their strengths and weaknesses.

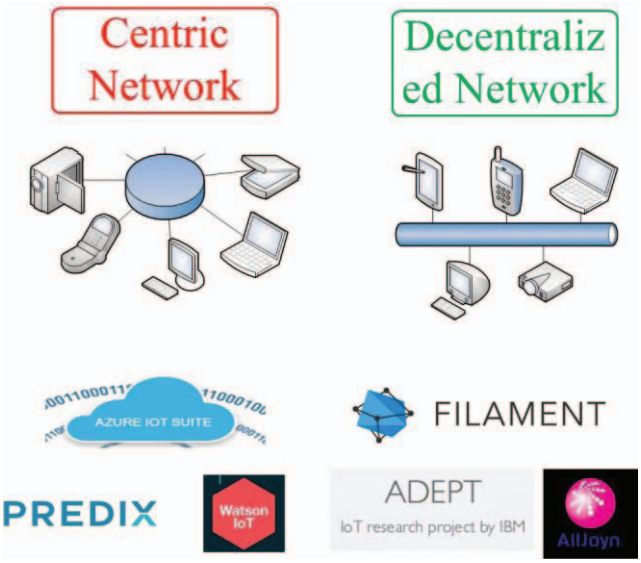

Figure 1. Centric & Decentralized Network.

The centric network model can support cross-regional networking of devices and utilize the services provided by the central server to greatly enhance and leverage the performance of the devices themselves. It can also obtain a large amount of data from a variety of heterogeneous devices to form the basis of Big-data services. Correspondingly, the central model has a high performance requirement for the cloud platform, a large demand for network bandwidth, and a potentially centralized risk. The decentralized network model has the flexibility to set up a network where equipment is transparent and resilient, and is capable to support real-time services. Based on blockchain techniques, it can also improve the degree of trust between devices. Its shortcomings lies in the equipment performance requirements are high, and is not conducive to large-scale network formation. In addition, the control ability of administrators and the overall performance of the network is weaker than those in the centric network model.

The new network model based on multi-layer distributed accounting can be regarded as an organic combination of the two methods, which not only effectively utilizes the performance and capabilities of the cloud server, but also significantly improves the overall security and reliability of the IoT with taking advantages of blockchain techniques. The most critical point is that the network model allows the existing center-based networks' iterative upgrade, which makes large-scale applications and deployments possible. This article mainly discuss the model from the security point of view and to elaborate the mechanism to achieve secure IoT.

In this paper, Sec. II will overview the model with discussion of its features. Sec. III describes some of the key definitions of the model for later description, while Sec. IV gives a concrete and practical reference network model. Discussion of security are in Sec. V. Sec. VI is the conclusion.

II. OVERVIEW

The network model divides the whole IoT into two parts: the edge layers and the high-level layers, as Fig. 2 shown.

A. Edge Layers

Edge layers provide entities of localized Internet of things and interfaces to high-level layers. In general, the edge layer is consistent with the current centralized network model: the cloud server manages device data and processes requests. In terms of security, the device is authenticated by the cloud server through the device's universal unique

978-1-5386-2011-3/17 $31.00 © 2017 IEEE
DOI 10.1109/IEEE.ICIOT.2017.34

identity (UUID) and the corresponding seamless hash algorithm. Powerful devices can choose to support non-symmetric encryption algorithm for data transmission. However, the edge layer is still different from the current common cloud service system of IoT:

1) The number of devices of a single edge layer is less thus its network efficiency is higher. According to the current application situation, one of the main factors limiting the ability of central networking is network bandwidth limitation. IoT equipment often needs heartbeat signals to keep contact with cloud. When many devices are networked, high concurrent access to the server often occurs. At this time, even if the cloud server's computing power is sufficient to handle high concurrent requests, the Internet's bandwidth limits and network instability and even service providers' constraints tend to cause delays, congestion and even loss of response and other issues in IoT practical applications. As a comparison, according to the bandwidth, an edge layer can limit the number of devices to a certain extent to avoid this application problem. Correspondingly, since the flexibility of cloud services, cloud servers' performance in the edge layer can be appropriately reduced by redeployment, thus avoiding waste of resources.

2) The devices in edge layers tend to be geographically or regionally concentrated. In theory, the edge layer is only a logical concept, and should not limit the region of devices. In fact, since the edge layer is similar to a local area network, the regionalization of the devices can be more conducive to the actual management and optimal allocation of cloud resources. All in all, the edge layer is a small centric IoT managed by a designated cloud platform or private data center with a certain type of security regulations.

B. *High-level Layers*

High-level layers are the part that connects the edge layer and implements the wide area networking capabilities of the IoT. In fact, the edge layer can be defined internally by the central node. From the view of whole network structure, it only needs to provide a data access interface to the high-level layer to indicate its identity. In other words, the edge layer is equivalent to a node that belongs to the higher layer which it connects to. The logic also applies to the connections between high-level layers. Thus, the complete network model will have the following structure: treating the edge layer as Layer 0, the superior high-level layer as Layer 1, then Layer 2, Layer 3, and so on. Unlike the edge layer, in the high-level layer, the network is decentralized. All nodes in the same layer run in a distributed way. For example, based on the Byzantine Fault Tolerance (BFT)[7] algorithm to maintain a block chain of distributed records, all the nodes belongs to one certain layer are easy to achieve self-management and a certain degree of fault tolerance. The advantage of the multi-layer network model is listed as below:

1) To enhance the application efficiency of the blockchain technology, and reduce the difficulty of its deployment. At present, the main problem of blockchain technology is that the recording block has high requirements for node capability, and the response speed of the distributed consensus algorithm is low when the number of nodes is large. When deployed in a multi-layer network model, each node in the high-level layers is physically at least one data center and certainly with enough computing ability. What's more, we can also limit the number of nodes per layer to 23-26 order to reduce the time and space costs required for blockchain records.

2) The entire IoT has a strong ability to self-adjust and resist risks. can be adjusted and anti-risk ability. At this time the block chain is deployed in each high-level layer, which means that even if single high-level layer is destroyed, other layers can also provide blockchain records.

The specific concepts in this model will be described below.

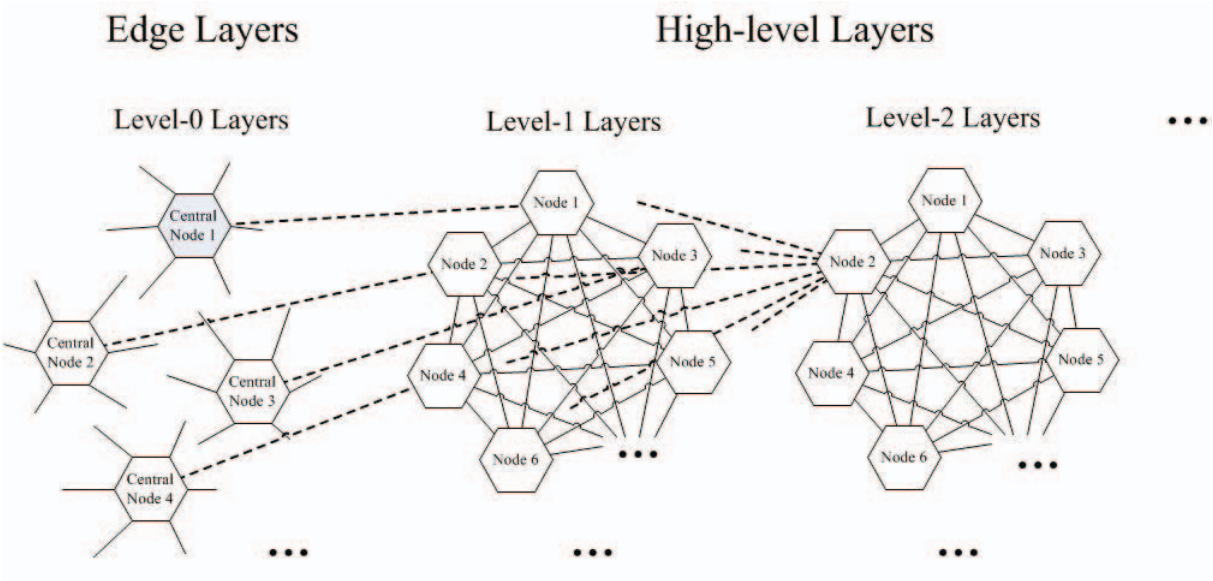

Figure 2. Overview of the proposed network model

III. DEFINITIONS

This section will give some basic definitions that conform to the network model. These definitions present some principles of the model and also help to understand the reference model in Sec. IV.

A. Entities

1) Objects

Objects are generalization concepts of devices in the model. Besides including the traditional concepts of objects like equipment, facilities and other materials, the concept of Objects also includes users, services, APIs, resources, and so on abstract things - as long as the things have the ability of network communications. Its definition is, a network entity which connects to a central node of edge layer and can be communicated with via the network. In other words, the whole network actually do not need to care about the specific forms of one object more than a point which can upload and download data. It is worth emphasizing that in many IoT models, users are often differentiated from the devices. However, this paper argues that there is no essential difference between the users and the devices at the level of network. The user requests can be handled in the same way as the device requests.

2) Nodes

A node is a network entity with capabilities as data transferring, data parsing, data storage, and so on. According to this definition, the ability of the nodes in the network is mainly as follows:

a) Data transmission. It includes data transmission between central node and objects in an edge layer, between high-level layers' nodes, and from subordinate layer to the superior layer.

b) Data analysis. Data analysis of the network model is actually two aspects: data analysis and screening based on a security algorithm and addressing process based on a certain rule.

c) Data storage. For high-level layers, nodes store blockchain data as distributed ledgers; for edge layers, the central node records information of its objects.

d) Other abilities. Because of the diversity of the edge layer, the central node may have to undertake a variety of other tasks, such as data encryption and decryption, automatic management of objects, response to the administrator's request and so on. These functions usually do not affect the overall model of network, but they may be some of the elements in a certain model.

3) Edge Layers

An edge layer is a network entity consisting of a certain number of objects and a central node who manages them. What's more, it can be regarded as a node of its superior layer. The edge layer can be of free architecture, but must meet the following principles:

a) Interfaces to superior layer. The central node must provide a standard data interface for addressing, allowing bidirectional data transfer and participating high-level layer activities as a node.

b) Self-independence. Any object of a specified edge layer can not communicate directly with other edge layers or higher layers (the actual application does not necessarily need to be so strict if the communication does not affect the overall functionality and security of IoT). All communication must be done from interfaces described above.

4) High-level Layers

A high-level layer is a network entity consisting of a number of nodes based on a blockchain protocol. What's more, it can be regarded as a node of its superior layer (if exists). The high-level layer must meet the following principles:

a) Interfaces to superior or subordinate layer. The high-level layer specifies trusted nodes as an upwardly or downwardly addressable and bidirectional data transfer interface by some algorithm. On the one hand, the interface is used to receive the superior layer data and then resolve requests. On the other hand, the interface is used to send trusted data that has been recorded in the blockchain to the superior layer. In practice, this interface can be one node or multiple nodes or even all nodes, corresponding to unicast, multicast or broadcast process of network communication. The specific form is determined by the selected algorithm.

b) Distributed consensus. According to the superior layer data or the subordinate layer data obtained by nodes, using some algorithm, like PoW[8], PBFT[9] or Paxos[10], to implement distributed and create and maintain a data summary record as a blockchain on all nodes.

c) Self-independence. Any node in a high-level layer can not send messages to nodes in other layers without distributed consensus.

d) Freedom. All nodes in the high level have permission to connect to and leave the layer at any time if authorized.

B. Processes

1) Addressing Between Layers

All nodes in a high-level layer maintain local inter-layer addressing methods. That is, any node is aware of addressing methods of nodes in this layer, in the superior layer and the subordinate layer it connects. (Precisely, the node knows the addressing method of all subordinate layers of this layer, but the subordinate layer, which is not subordinate to this node, is usually not addressed by this node unless the corresponding node in this layer is faulty.) Generally, these information will be recorded in the blockchain after distributed consensus. The so-called addressing methods include two aspects:

a) The mapping between logical layer and practical address. The name or ID of a given node or layer can be mapped to an address of a specific bearer network, such as an IP address of a TCP / IP network.

b) The authentication mechanism provided by each addressing object. Each addressed object needs to provide a mechanism for verifying its identity, for example a password after asymmetric encryption. Because the addressing method save a mapping method related to real information, it is necessary to prevent the leakage. A random encryption and a lifetime are typically set for this method. Since there is a

validation mechanism also for nodes in the same layer, if a node is broken, other nodes will find the matter and apply a distributed consensus for the replacement of the addressing method to it.

2) Cross-layer Addressing

As mentioned above, all nodes in a high-level layer have addressing functions for the superior and subordinate layers. Therefore, the overall network transmission process is: in high-level layers, when data is migrated from one layer to another, the layer information before migration is added to the the source address, and the information of migrated layer will be removed from the destination address; in an edge layer, the central node completes the source address of the object according to local addressing rules and fill the destination address with source address from received messages or other ways when sending data, while it passes the data directly to the specified object after data acquired since the destination address is agreed with its local addressing rules.

3) Block Establishment & Update

In the high level, a new block will be established or updated when a new distributed consensus process occurs. Each node will record consistent messages locally, including the record sequence, the content summary, the time of occurrence, and so on. While a new block is added to the blockchain system, a globally unique ID will be generated. Besides, the block can set its lifetime depending on the network characteristics and capabilities. All record in the block is hashed and encrypted by asymmetric encryption.

IV. Reference Model

This section provides a reference implementation of the network model in conjunction with specific techniques and specifications. The implementation can be applied to non-high real-time requirements, cross-regional cooperation networking scenarios. This model will focus on discussing the implementation of the network layer and omit the details of other aspects. The following will be discussed in terms of the edge layer and the high-level layer.

A. Edge Layers

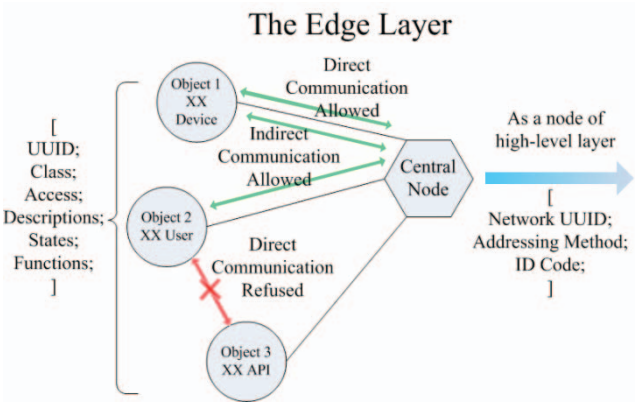

Figure 3. The Edge Layer.

The structure of the edge layer is shown in Fig. 3. The central node of the edge layer will maintain a NoSQL database to store objects' information in the edge layer. The main body of the information includes the UUID, the identity information, states data collections, function collections and format documents and so on.

1) Universally Unique Identifier (UUID)

The UUID is automatically added by the center node of edge layer once a new object is added to this layer. It consists of the edge layer URL determined by the cross-layer addressing process (i.e., all the superior layers' ID with itself) with the local unique identifier.

2) Identity Information

An object reports the identity information to the central node when the UUID is obtained, including its class, permissions, descriptions, and so on. Its class indicates the category to which it belongs, for example, a smartphone whose category may be Device - Mechanical Device - Smartphone - XX brand. And for an IoT user, its class may be User - Individual Users - Name. Permissions record the object's authority information. From the perspective of the network layer, it mainly refers to access to other objects or layers. In practice, other type of appropriate permissions may be necessary. Descriptions are documents and manuals for introductions and directions of the object.

3) States Data Collections

The state properties of the object will be created after the new identity is established. State properties include static and dynamic properties. Static properties include the initialized property of the object when it is generated in the network – i.e. an IoT address. The dynamic properties contains names and corresponding values of all the data which the object can uploaded to the central node.

4) Functions Collections

The collections contain the actual functions of objects and the self-management functions of the edge layer. The former is determined by the specific object, while the later is determined by specific management methods. From the network perspective, the latter is relatively more important since it contains parts as data analysis, network requests processing, addressing and other modules.

5) Format Documents

Format documents are used to record the local data analysis format, and the superior layer's standard data format.

6) Communication Rules

The actual communication process in the edge layer is divided into the following categories by source and target:

a) From one object to another. This process is not supported. Given that the edge layer is centered in this reference model, it is not supported directly. All messages between two objects must be passed by the central node.

b) From one object to the center node. Data from the object are sent to the central node with its UUID and digital signature. If the object is of weak computing ability, the signature can be an unencrypted hash code or even nothing based on needs of IoT security. Since only the central node performs as a foreign data entry and exit, the network is actually equivalent to a private local network in Internet.

Therefore, existing Internet security rules are available. After receiving the data, the center will verify the identity by UUID, and then call the data analysis functions to parse it according to the object's class. After confirming the request, the central node will execute or refuse the request in accord with the permission messages of the object.

c) From the center node to an object. Data from the central node is reorganized by data analysis module and sent to an accessible address decoded by the method of local addressing module. Those modules are restored in database of the central node and can be find by UUID of objects.

B. High-level Layers

High-level layers' structure is shown in Fig. 4. There is no central node in the high level, so each node is data independent. The data management within each node is done by itself. But the data exchange between nodes is recorded by the common books maintained by all nodes - the blockchain. There are a variety of options to use specific blockchain coherence algorithms, such as the Proof of X[8,11] for a public-chain, the BFT algorithms (PBFT[9],Quorum[12]) or the non-BFT (Paxos[10], Raft[13]) algorithms for a private-chain,. The security guarantees of different types of algorithms are different.

In this model, the purpose of the high-level layer is not to solve daily communications of edge layers, but the proof of existence of contact between objects. That is to say, only the contracts, which defines rules of objects' interactions, like how to communicate with each other, how to allocate the benefits, and how to provide services to each other, will communicate through the high-level layer. Such a mechanism will ensure that the contracts will be recorded by at least one blockchain.

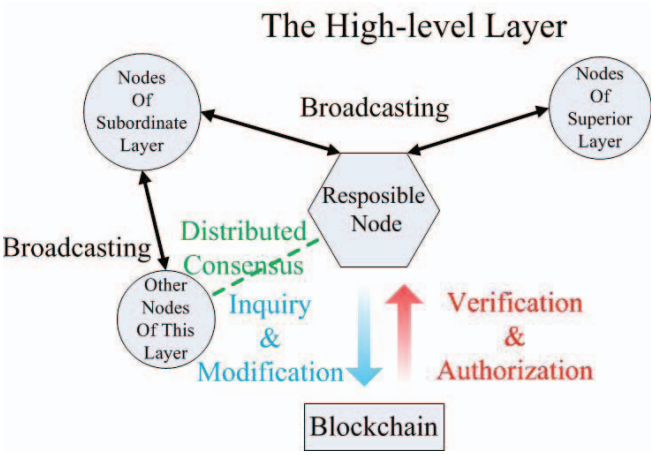

Figure 4. The High-level Layer.

1) Establishment of High-level Layers

a) External consultation. The blueprint of layer is negotiated through other channels instead of IoT network, such as real communication, Internet communications and so on. Rules and members are decided through other ways whose security are independent of IoT network.

b) Application from one node. The node issues an application to another existing high-level nodes by cross-layer addressing. The other nodes agreed with the application will leave its own layer and join the newly created high-level layer. Then the first node will send an application to another high-level layer and request to be its subordinate node. If success, the whole new layer will become a node of that layer and the establishment application is over.

Whatever means of establishment are used, nodes of a new layer should create a beginning block as start of blockchain, with record of important messages, like the establishment time, the public key of each node, the selected protocols of distributed consensus, information of existing nodes, the subordinate layer's authentication method corresponding to each node, the superior layer's addressing method and corresponding authentication methods and so on.

2) Connections of Nodes

When a new node applies to become a member of a high-level layer, it needs to send the application information to all nodes of that layer. If the application is approved with distributed consensus, the node will receive the consent message of other nodes with the valid blockchain information of the layer. After that, all nodes (include the new one) will automatically renew the first block of their blockchain by adding messages of this new node.

For the layer who loses a node, it will also change its first block by adding an invalid tag into the messages of the missing node.

3) Communications

Communications are defined as two type. One type is the *working communication*. Object A and B are connected by their central nodes of edge layers, and the two edge layers are actually working in a peer-to-peer mode. The communications rules, such as addressing, encryption, rights and interests, valid period, etc., are set by a contract between the two edge layers previously. If the contract is safe and correct enough, working in this mode ensure efficient and reliable cooperation of objects. Therefore, the other type of communication is the *contract communication*. It will help two objects or two edge layer establish a secure, effective and widely admitted contract. The process is described as below.

Assume that object A have made a request to create a contract with object B. This request will experience a high-level layer communication process and realize a blockchain record.

Now A submits a communication request P to B through its own edge layer. P consists of ID information I, communicating information C and main context M.

$$P = \{I, C, M\}$$

For *the first communication* between A and B, the ID information contains only the current content summary $\mathrm{Sum(M)}$.

$$I = \{Sum(M)\}$$

The content summary $\mathrm{Sum(M)}$ is actually an overall hash value for the main context M and the current time. The communication information C includes the source address A_s and the destination address A_d, which are actually combined sequences of layer addresses.

$$C = \{A_d, A_s\}$$
$$A_d = \{A_{L_1}, A_{L_2}, \dots A_{L_N}\}$$
$$A_s = \{A_{L_0}\}$$

P will first be broadcasted to all nodes of L_1, the higher layer of A's edge layer. Assume one node N_1 of L_1 receives P. N_1 will ask other nodes of L_1 for distributed consensus to confirm such statements as below.

Statement 1: *P really comes from L_0 (i.e. P does come from the layer that it claims to point to in the source address A_s).*

Statement 2: *No block ID information exists in I;*

Statement 3: *There is no block B_p corresponding to P;*

If Statement 1 does not hold, then P is an invalid contract request and L1 will ignore it. The network can recognize the authenticity of Statement 1 because of the following reason: if each layer can confirm the correctness of the nearby layer's source address by checking its blockchain, then the overall correctness of address will be guaranteed. In other words, if a fake contract occurs, the fake contract must be recognizable unless an attacker breaks the consensus mechanism of a whole high-level layer. The specific situation will be discussed in Sec. V. If Statement 1 is true, check Statement 2.

If Statement 2 is not stated, it means that this request is not *the first communication* between A and B. We will discuss the situation later. If Statement 2 is true, check Statement 3.

If Statement 3 is not true, then N_1 is not the first node to receive the request P (some node has already create the block), so there is no need to apply to create a block (just wait for others' applications). There are many methods to check Statement 3, e.g. compare $\mathrm{Sum(M)}$ and recent element of K in B_p.

If all the above statements are true, a new block B_p will be established (if B_p is already existed but expired, then extend its valid period). B_p will acquire a corresponding UUID (address of N_1 plus local sequence ID) I_{BPL1}, and record the following information:

I_{BPL1};

Contract Summary Array K (initially only the first term, the Sum (M) and current time T_c);

Responsibility Node N_r of this layer (initially N_1);

Contract Accomplishment Flag F (initially false);

Block Failure Time T (aimed to save storages, user defined);

The N_1 node then adds I_{BPL1} to I, move A_{L_1} from A_d to A_s and broadcasts P to L_2 layer.

After P is broadcasted in L_2 layer, the structure of P changes to P' as below.

$$P' = \{I', C', M\}$$
$$I' = \{Sum(M), I_{BPL1}\}$$
$$C' = \{A_d', A_s'\}$$
$$A_d' = \{A_{L_2}, \dots A_{L_N}\}$$
$$A_s' = \{A_{L_0}, A_{L_1}\}$$

Similar things going on when P arrived at L_N layer.

$$P^N = \{I^N, C^N, M\}$$
$$I^N = \{Sum(M), I_{BPL1}, \dots I_{BPLN-1}\}$$
$$C^N = \{A_d{}^N, A_s{}^N\}$$
$$A_d{}^N = \{A_{L_N}\}$$
$$A_s{}^N = \{A_{L_0}, A_{L_1}, \dots A_{L_{N-1}}\}$$

Notice that all the block ID has been recorded in P^N. That is to say, B and A is able to use the same blocks to finish the accomplishment of contract if more *following communication* is needed.

Now consider the *following communication*. If B replies A by sending request P_2. Now the structure of P_2 becomes as below.

$$P_2 = \{I_2, C_2, M_2\}$$
$$I_2 = \{Sum(M_2), I_{BPLN-1}, \dots I_{BPL1}\}$$
$$C_2 = \{A_{d2}, A_{s2}\}$$
$$A_{d2} = \{A_{L_{N-1}}, \dots A_{L_0}\}$$
$$A_{s2} = \{A_{L_N}\}$$

Assume one node N_2 of L_{N-1} receives P_2. N_2 first check Statement 1, 2, 3 and do things like N_1, unless it find Statement 1 is true but Statement 2 is false. If happened, N_2 will check another two statement as below.

Statement 4: *N_r in $I_2(1)$ is equal to N_2;*

Statement 5: *Sum(M_2) does not exist in summary array K of $I_2(1)$;*

If Statement 4 does not meet, it means N_2 is not the responsible node who are pointed to deal the request. Thus the request can be ignored. But we can set more rules to design a robust system here. For instance, if Statement 4 is not true , N_2 will first try to question N_r if P_2 has been dealt with. If N_r failed to response or some other situations happen , N_2 can apply to be responsible node N_r for P_2 in this layer. If Statement 4 is true, check Statement 5.

If Statement 5 is not satisfied, at least one message with same $\mathrm{Sum(M)}$ of P_2 must has been sent. Since we require a timestamp in M_2 which leads to absolutely different $\mathrm{Sum(M_2)}$ of different P, P_2 must has been sent. That is to say, there is something wrong in the network- for example, a DDoS attack. Therefore, N_2 will ignore P_2 and inform network administrators.

Both Statement 4 and Statement 5 are satisfied indicate that P_2 should be handled by N_3 now. N_3 then applies to modify the block $I_2(1)$ as below.

I_{BPLN-1}; (No change)

Contract Summary Array K = { { Sum (M), T_c } , { Sum (M_2), T_{c2} }}; (add new summary and new time)

Responsibility Node Nr of this layer; (No change)

Contract Accomplishment Flag F; (No change)

Block Failure Time T; (No change or enlarge)

The N_2 node then moves the sequence of I_2, move $A_{L_{N-1}}$ from A_{d2} to A_{s2} and broadcasts P_2' to L_{N-2} layer. Now P_2' looks as below.

$$P_2' = \{I_2', C_2', M_2\}$$
$$I_2' = \{Sum(M_2), I_{BPLN-2}, \dots I_{BPL1}, I_{BPLN-1}\}$$
$$C_2' = \{A_{d2}', A_{s2}'\}$$
$$A_{d2}' = \{A_{L_{N-2}}, \dots A_{L_0}\}$$
$$A_{s2}' = \{A_{L_N}, A_{L_{N-1}}\}$$

With similar process, the *following communication* of A and B can be easily realized.

In this case, the contract between A and B will be gradually established and the communication digest can be recorded for at least one high-level blockchain. You can also introduce a termination Statement to terminate the contract establishment process.

Statement 6: *Request P includes a tag of contract ending;*

If Statement 6 is established, the responsible node N_r will be responsible for setting the flag F to true. Correspondingly, F will be a test condition when checking the existing of a certain block.

4) Management of Blockchain

High-level layer also provide contract proof services. That is, all blocks are allowed to be queried according to the ID number, as long as the inquirer has the appropriate authority. It indicates that query interfaces for the blockchain is needed. Besides, nodes is able to initiate a block cleanup request, which requests all other nodes in the layer to delete blocks out of the expiration date, to reduce storage requirements.

5) Self-examinations

The high level node initiates a review event at random intervals. Two categories of the event are described as below.

The first category is a review of nodes in the same layer. A node sends a request to test the legitimacy of all other nodes. The request requires that the first block will set invalid tag of a node if it is considered failure unanimously, that is, it loses contact or can not pass the validation mechanism recorded in the first block.

The second category is a review of superior layer. A node sends a request to check the connectivity of all nodes in its superior layer. If all nodes of the superior layer lose contact or can not be approved by reserved validation methods, the alarm mechanism of nodes in this layer will be triggered. And this layer will try to find a new superior layer and all running communications through the old superior layer will be frozen. Finally, the first block updates messages of new superior layer.

V. SECRUITY ANALYSIS

This section discusses the security of this IoT model. It is noted that the nature of the edge layer is equivalent to that of a centric network, and its security is mainly ensured by the center node of the edge layer. Therefore, we will mainly discuss the impact of high-level layers.

A. Universal Identification

The realization of Universal Identification becomes the foundation of global trust. The identity of any object in its local edge layer is implemented by a local policy. For other layers, the authentication is actually done by cross-layer addressing. Recognition of this object from other nodes or other objects is derived from the source address information carried in the communication request sent by it. According to the cross-layer addressing process, the address information is marked by all high-level layers in routes after distributed consensus. Therefore, the identity camouflage will be meaningless.

On one hand, if an attacker disguises as an object by giving a same address, the response messages will be routed to the real address through the high-level layer, unless the camouflage breaks all the high-level layers on the path. As mentioned earlier, the BFT distributed consensus algorithm can guarantee functions of a network with nearly 33% failed nodes, while the percentage of PoX algorithm may be even higher. It greatly enhances the credibility of identity authentication. On the other hand, the regular self-examinations in high-level layer also reduce the likelihood of long-term presence of malicious nodes.

B. Authentication

The process of granting rights to objects is realized through their contracts. Objects from different edge layers co-work in accordance with their respective interfaces and the corresponding way to achieve cooperation after reach agreement by signing contracts in IoT network. As such, their respective rights and requirements are recorded by the blockchains of participated high-level layer during the contract establishment process. According to the returning block ID, the summary of the contract can be queried directly at any time. Because of the non-repudiation of this summary, the interests of each objects can be guaranteed, unless someone breaks down all the high-level layers that record the summary.

C. Privacy

Since the blockchain records contracts between objects, it is necessary to assess the possibility of privacy disclosure.

One part of privacy is the real-world identity of objects. Real-world ID of an object is actually encrypted when passing through the center node of its edge layer, what is recorded in blockchains is actually a pseudonym of the object and an IoT address of the edge layer. What's more, even how the IoT address related to real IP is encrypted in blockchains. Thus someone cannot recognize an object's real ID by only spying on blockchains.

Another privacy is about context of contracts. Actually, only a summary, maybe a hash code of real context, of each message is recorded in blockchains. The summary can be used to examine integrity and correctness of an existed contract but is hard to translate to complete context reversely.

These two points make sure that writing blockchains will not increase the risk of privacy leak.

D. Influences of Flexibility

An IoT may be required to have a certain degree of flexibility, that is, allowing the object or node connect or leave the network freely. In this model, a local object's

connecting only affects the central node of an edge layer, which is supported by different local strategies to deal with such situations.

Thus, the focus is on how to deal with changes in network nodes of high-level layer. In general, high-level nodes is expected not be constantly changing to ensure network efficiency. But we still hope the change will not affect the functions of entire network. In this model, if any node connect or leave the network, it will be known and recorded by other nodes of the layer in process of distributed consensus. Therefore, as long as there is at least one node, a high-level layer will be able to run properly. Even if all nodes are failed, the worst result is only a network partition: the subordinate layers of this layer become temporarily independent network without any influences of their internal functions. The existence of the self-examinations will remind those independent networks try to re-connect to another higher layer. What's more, the addition and loss of nodes will not significantly affect the performance of IoT: any high-level layer does not achieve its function depending on a specific node, the increase in the number of nodes will only required more time-cost of distributed consensus algorithms. This increase in the cost of time is acceptable. After all, each node's entity is at least a data center or cloud service platform.

E. Super Attackers

Assuming that a very powerful attacker does break a high level layer. So he can completely manipulate the recording process of blockchain in this layer, and provide false layer addressing method for other layers.

Assuming all requests that span this layer will be intercepted and manipulated. However, due to the encryption of the contract, the attacker can only achieve the information forwarding and interception, and can not directly tamper with it. Forwarding IoT messages do no harm but increase network traffic. Recipients will treat them as garbage. Interception of messages will only lead to alert of reconnections, since a subordinate layer will find its superior layer losing contact after self-examination. That is, the attacker can only make a certain degree of interference with the communication of IoT.

If the attacker does not interfere with the communication, but wantonly tamper with the blockchain. As a result, the contract that was originally recorded may fail or can not be found at this layer. But unless the contract is established only through one layer, other layers can still provide proof of the contract. And, he can not directly affect the object itself by spying on the contract (the contract content can not be easily spied on because of the summary and encryption). Only when the attacker is strong enough to crack the asymmetric encryption, or private key leaked, the objects can be influenced.

VI. CONCLUSIONS

The multi-layer network model based on block-chain technology provides a feasible solution for the establishment of wide-area secure IoT network. It at least provide four advantages.

1) The IoT equipment is coordinated locally by a centralized instrument, with enhanced ability and safety.

2) The presence of multiple centers reduces the computational load and network load of IoT, as well as reduction of concentrated risk.

3) Compared to the traditional centric network, it reduce little communication efficiency because most of the daily communications are peer-to-peer between centers.

4) Contracts record in multiple blockchains ensure secure and reliable IoT network.

Potential defects may be related to the following.

1) Enhanced cost of total network.

2) Taking much time and resources to maintain contracts.

In the future, we will try to build practical applications for this model, aiming analysis of those uncertain problems and improvement of the network model.

ACKNOWLEDGMENT

Thanks for directions from Prof. Chunxiao Xing from Tsinghua University. And thanks for discussions and opinions from Dr. Huan Chen and IoT working group in Kingdee Research.

This work is supported by the central grant funded Cloud Computing demonstration project of China, R\&D and industrialization of the SME management cloud (Project No. [2011]2448), hosted by Kingdee Software (China) Company Ltd., under the direction of National Development and Reform Committee of China; the construction fund of National Engineering Research Center for Supporting Software of Enterprise Internet Services (Project No. 2012FU125Q09); the Guangdong province project, under grant No. 2015B010131008; the fund for leading talents of Guangdong Province [2012]342; the Shenzhen high-tech project, under grant No.JSGG20160331101809920.

REFERENCES

[1] Familiar, Bob. "IoT and Microservices." Microservices, IoT, and Azure. Apress, 2015. 133-163.

[2] Morris, Henry D., et al. "A Software Platform for Operational Technology Innovation." International Data Corporation (2014): 1-17.

[3] High, Rob. "The era of cognitive systems: An inside look at ibm watson and how it works." IBM Corporation, Redbooks (2012).

[4] Panikkar, B. S., et al. "ADEPT: An IoT Practitioner Perspective." (2014).

[5] Villari, Massimo, et al. "Alljoyn lambda: An architecture for the management of smart environments in iot." Smart Computing Workshops (SMARTCOMP Workshops), 2014 International Conference on. IEEE, 2014.

[6] Malviya, Hitesh. "How Blockchain Will Defend IOT." (2016).

[7] Castro, Miguel, and Barbara Liskov. "Byzantine fault tolerance." U.S. Patent No. 6,671,821. 30 Dec. 2003.

[8] Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." (2008): 28.

[9] Castro, Miguel, and Barbara Liskov. "Practical Byzantine fault tolerance." OSDI. Vol. 99. 1999.

[10] Lamport, Leslie. "Paxos made simple." ACM Sigact News 32.4 (2001): 18-25.

[11] King, Sunny, and Scott Nadal. "Ppcoin: Peer-to-peer crypto-currency with proof-of-stake." self-published paper, August 19 (2012).

[12] Cowling, James, et al. "HQ replication: A hybrid quorum protocol for Byzantine fault tolerance." Proceedings of the 7th symposium on Operating systems design and implementation. USENIX Association, 2006.

[13] Agrawal, Prathima. "Fault tolerance in multiprocessor systems without dedicated redundancy." IEEE transactions on computers 37.3 (1988): 358-362.