International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

# Blockchain Solution for Network Discovery and Maintenance

***Mr. Harshal Haridas[1], Prof. Pujashree Vidap[2], Aovi Anjankar[3], Vaishnavi Bhujbal[4], Aaditi Nirfarake[5], Sanyam Chhoriya [6]***

*[1,2,3,4,5,6]Pune Institute of Computer Technology*

**A B S T R A C T**

Network discovery is important because every single IT team needs to have network visibility to fulfill its duties. Without a thorough knowledge of what devices are connected to the network, their relationships, and how they communicate with each other, IT teams are unable to understand the ins and outs of the network. As a result, if a network experiences downtime due to any attacks on central or online repositories it can be nearly impossible to figure out the point of failure. Hence network discovery and monitoring software is required. Log data is crucial to detect mischievous activities conducted in a network. Traditional log management systems that depend on cloud and centralized storage servers are vulnerable to a single point of failure and lack transparency and trust since the adversaries tamper log records. Blockchain is decentralized, distributed and highly immutable making it tamper-proof and almost impossible to hack. Such technology is essential when it comes to attacks on sensitive data.

**Keywords:** Blockchain, Network Management, Hyperledger Fabric

## Introduction

### *1.1 Network Management*

Network discovery and management has become an essential part of computer science not only to keep a track of all the nodes entering and leaving a network but also to detect a single point of failure and thereby protect our network from attacks which may tamper with the highly sensitive data that nodes contain. Network management is a challenging task due to various arrangements i.e topologies nodes form together and a thorough knowledge is required about how devices are connected and communicate with each other. Manual network management of large scale networks is unfeasible due to the need for engineers specialized in different aspects and types of network devices and their management, limited time, need to define a strategy for configuration management, and the effort to track the configuration state of a large number of different devices. These factors obviously increase the costs and effort required for network management. Traditionally network management is done so through a network administrator. The network itself is monitored using network management frameworks. The network administrator would be tasked with finding any point of failures if they were to arise. This could be troublesome if multiple issues were to arise within the network at once.

### *1.2 What is Blockchain?*

Blockchain has gained enormous attention both as a security and optimization solution and has revolutionized our conventional ways of approaching problems in the technical field. It became popular in 2008 when Satoshi Nakamoto was working on the technology for it through Bitcoin. Blockchain is a technology for storing an immutable history of transactions in a decentralized platform by using cryptographic principles. Many industries have become interested in adopting blockchain within their IT systems for better security. In blockchain, the blocks are connected to each other via links which are established by the hash of each block. Each block stores the hash of the previous block and shares a distributed ledger. When any new transaction is done a new block is added to the blockchain network and is updated in the ledger. This unique feature of blockchain makes it highly immutable and almost impossible to hack as anyone who wants to hack the blockchain ledger would have to alter the data on over half of the devices that are connected to the network. They would also have to account for the fact that each transaction block is a hash of its previous block. Therefore, such a task is much harder to accomplish than it sounds. Blockchain is a decentralized network hence the technology does not depend on any single node for its efficient functioning and also increases flexibility and scalability. Even though blockchain was first invented for building a public and open trustless network without any

[1] Harshal Haridas - Chief Cybersecurity Architect, Honeywell
[2] Pujashree Vidap - Assistant Professor, Pune Institute Of Computer Technology
[3] Aovi Anjankar - Computer Engineering Student , Pune Institute Of Computer Technology
[4] Vaishnavi Bhujbal - Computer Engineering Student , Pune Institute Of Computer Technology
[5] Aaditi Nirfarake -Computer Engineering Student , Pune Institute Of Computer Technology
[6]Sanyam Chhoriya -Computer Engineering Student , Pune Institute Of Computer Technology

central authority, it is evolving towards permissioned and private platforms for enterprises. Private blockchains are similar to public blockchains, they also are immutable, nodes share the same ledger, but the access to the network is permissioned. This means that permission and role for each node have to be granted.

### *1.1. What are Hyperledger Fabric and smart contracts?*

Hyperledger Fabric is one of the most popular permissioned blockchain frameworks and was developed by the Linux foundation. It can be used to develop smart contracts (called as chaincode in Hyperledger Fabric community) using general-purpose programming languages, e.g. Go, Node.js, and Java. Smart contracts are executable codes that run on top of the blockchain to facilitate, execute, and enforce an agreement between untrustworthy parties without the involvement of a trusted third-party. Typically, smart contracts are developed using domain-specific languages (DSLs), which have specific restrictions and features for blockchain, but Hyperledger Fabric applies general-purpose programming languages to reduce learning costs for developers. Therefore, because of the absence of specific restrictions and features, the types of risks associated with Hyperledger Fabric may be different from these associated with the smart contracts written by DSLs. Hyperledger Fabric is a platform for software solutions that use a distributed general ledger (database). It is a private, permissive blockchain infrastructure that provides a modular architecture with delineating roles between nodes within the Blockchain system, executing smart contracts, and configurable consensus and membership services. Its advantages are: identity management, channel privacy and confidentiality, efficient data transmission processing, chaincode functionality, and modular design.

This paper focuses on studying and implementing the use of blockchain technology in managing networks for better security and flexibility. For this, we use hyperledger fabric as a framework to develop and deploy smart contracts for the network. Blockchain being decentralized, such a system does not depend on any central authority and thus there is no single point of failure. It is highly immutable and shares a distributed ledger thus making it highly transparent and secured. Also, many points of failures could be contained by just making proper restrictions and rules for the network on the smart contract.

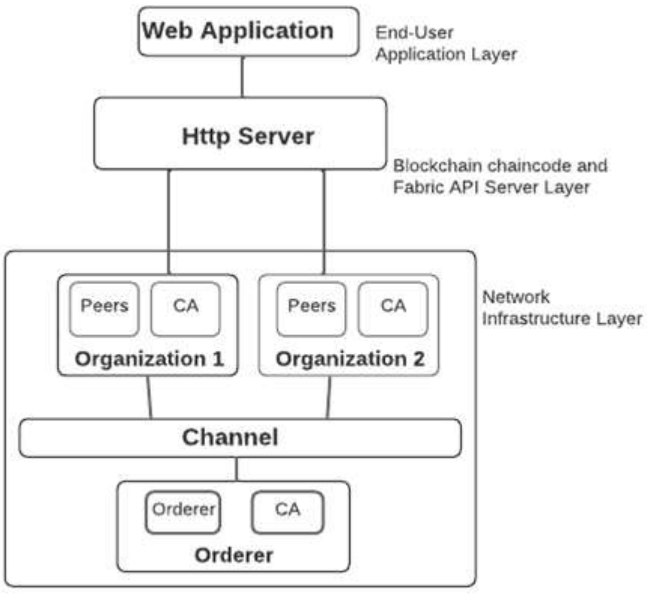
## 2. Literature Reviewed

Various studies have been done in reference to the use of blockchain in network management and various other related topics. In [1], a blockchain based network management system is proposed using ethereum as the framework and using smart contracts for writing different conditions and behavior. [2] focuses on a blockchain-based system that is suggested for securely discovering and storing networks. Experiments were performed using Mininet, Cisco Packet Tracer, and Ethereum blockchain with the network inference algorithm. [3] aims at presenting the Hyperledger frameworks and highlight all the details of Hyperledger Fabric in order to determine whether its application in practice is justified and concluding that Hyperledger Fabric is the most comprehensive and the most flexible Hyperledger framework with the largest number of use cases. [4] presents a performance and scalability analysis of permissioned blockchain platforms, including PoA based Ethereum, Quorum, Corda, and Hyperledger. [5] provides a comprehensive and comparative study of the 5 major frameworks (Fabric, Ethereum, Quorum, MultiChain and R3 Corda) with regard to the community activities, performance, scalability, privacy and adoption criteria. [6] presents a comprehensive survey of blockchain-enabled smart contracts from both technical and usage points of view .A taxonomy of existing blockchain-enabled smart contract solutions and the existing smart contract-based studies were discussed. In [7], the focus is on possible risks associated with Hyperledger Fabric smart contracts. [8] advocates the advantage by combining blockchain, SDN and CIDS, and then design a general framework of blockchain-based collaborative intrusion detection in SDN, shortly BlockCSDN. [9] examined blockchain technology as a state-of-the-art technology and provided a systematic review of blockchain regarding decentralized authentication and also provides useful and classified information which can enhance the understanding of how various authentication systems can be combined with blockchain technology. [10] proposes a log record scheme combining blockchain, which ensures the consistency of log records through the blockchain consistency protocol and also introduces the architecture and implementation of the log system and verifies the feasibility of the system through experiments. In [11], a blockchain-based network log data storage, query, and audit system was proposed and they implemented and deployed the scheme in a physical networking environment to collect log records.

[12] mainly analyzes the automatic discovery mechanism of nodes based on the Kademlia algorithm, including the principle of the protocol, the process of communication handshake and the specific algorithm mechanism.

## 3. System Design

### *3.1. Architecture Design*


Web Application
End-User Application Layer
Http Server
Blockchain chaincode and Fabric API Server Layer
Peers
CA
Organization 1
Peers
CA
Organization 2
Network Infrastructure Layer
Channel
Orderer
CA
Orderer

Architechture Diagram

#### *3.1.1 .Application:*

The system application has 2 modules which are User and Admin. Different modules have different functions, and they all connect to the HTTP server through predefined APIs. Any Node to join the organization network will be displayed on Admin portal and admin will acknowledge the Node by calling the predefined APIs. ReactJS framework is utilized to create the user interface to enable a user-friendly experience. REST API calls are used for frontend to backend communication, and JSON web tokens are used for authentication. The backend code and smart contract are written in JavaScript, and ExpressJS serves as the REST API server.

#### *3.1.2. HTTP server:*

The HTTP server is a middleware. HTTP server must interact directly with Hyperledger fabric network to complete specific business logic in addition to receiving and processing requests from applications.

#### *3.1.3.Organisation:*

In this system, the Organisations are the different departments of the enterprise. Each organization has a local fabric CA and a set of peers. The local fabric CA manages the certificates of all peers in the organization. Only the peers that pass the verification can obtain the certificate ID and join the blockchain. There are different types of nodes in an organization, e.g., committing peer, endorsing peer, ledger peer, and anchor peer. A peer can install multiple SCs, and any operation results by SCs execution will be inevitably stored in the blockchain ledger.

#### *3.1.4.Orderer:*

This component is responsible to receive all the transaction requests on the specified channel, and sort these requests by the consensus protocol, then package the verified transactions into blocks and finally broadcast them to the fabric network through communication protocols. Different ordering service implementations which can be used in Fabric implementation are Raft, Kafka, Solo.

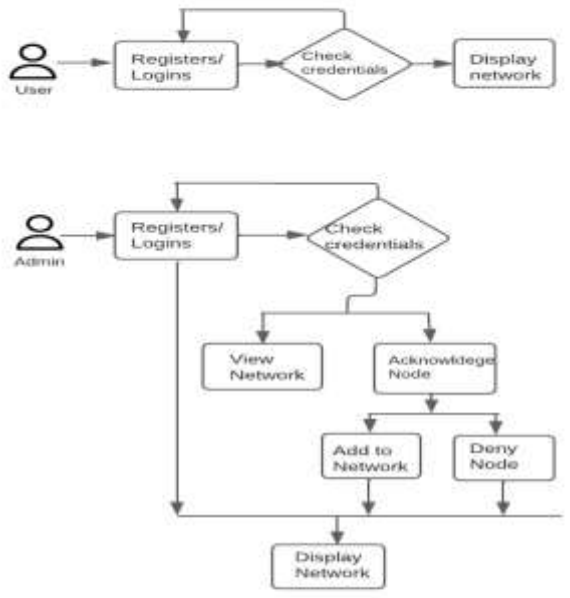
#### *3.1.5.Channel:*

A channel provides a private bridge for a set of peers to communicate with each other. In this system, the channel maintains the communication between organizations.

### *3.1.6.SC:*

A SC is a program that implements a specified interface. The states of the blockchain ledger can be inevitably updated by executing a SC. In this paper, we propose a set of SCs to realize specific business logic. After a SC is instantiated in the channel, all peers that installed the SC can interact with blockchain ledger through the blockchain network by invoking the interface defined in the SC.

### *3.2. System Flow Diagram*



## 4. Design and Implementation

We design and implement a Network Management system with Hyperledger Fabric as the underlying blockchain architecture to acquire high throughput and scalability. All node information in the system is stored in the key-value database (e.g., LevelDB, CouchDB) in the form of key-value pairs. We design various data structures to store the node features and keep them traceable by analyzing transactions on the chain. We also design a variety of SCs to support transactions in the enterprise network.

### *4.1 Technology stack*

To implement an end-to-end blockchain system using HLF, various components are involved in the workflow. The following components are used in the development of the framework:

- Hyperledger Fabric: Hyperledger Fabric platform is an open source blockchain framework.
- Docker Compose: Used to deliver software packages called containers (docker-compose-ca.yaml, docker-compose-couch.yaml, docker-compose-net.yaml).
- Couch DB: Open-source database, which allows the storage of data in JSON format and is used as an external state of a database for Hyperledger fabric .
- Node JS: This is an open-source cross-platform backend where the script runs in the terminal and executes JavaScript code outside of the browser. In this project, Node JS is used to provide an API to react with Hyperledger Fabric blockchain (which performs the first level of user authentication and acts as the gateway to the Fabric smart contracts) .
- React JS: This framework has been used to build the client application web interface.

## *4.2. Enterprise Network*

The HLF network, which consists of departments of the enterprise organization interacting in the distributed ledger channel with their peers, serves as the framework for the system. According to Figure 1, every department of the enterprise in the network is linked to a single channel. Using the test network offered by HLF, our network maintenance system is configured. Moreover, the organizations using the current network are changed to represent the departments of the enterprise by altering the docker files, configuration files, and related certificates. The main changes needed to add the use case organizations are to the configtx.yaml file's organization names and the corresponding CAs. As shown in Figure 1, two departments of an enterprise organization and a channel are added to the improved network, and all necessary credentials for the company and participating peers are generated.

## *4.3. Databases*

The two peer databases supported by HLF are LevelDB and CouchDB.. LevelDB is the by default database of HLF and saves data as key-value pairs whereas, in CouchDB data is stored as JSON documents. Rich queries over JSON documents are supported by CouchDB in distinction to LevelDB, which only offers composite key queries. Because of this, CouchDB is used as a peer database to model each patient's patient health record as a JSON datatype.No indexing or design document (DDOC) features of CouchDB have been used yet, although they can be considered in the future. The number of CouchDB Docker images depends on the number of peers and runs on the same server as the peer. Since each peer has its ledger, an HLF network requires a single CouchDB image for each peer.

Each node in the network which represents a device is identified using its IP address. The Node structure has the fields like the owner of the device, the basic information of the node(eg. IP address, MAC address) , the description of the Node (e.g. timestamp of joining, bandwidth used ) and so on.The data structure for recording nodes is shown in Table 1, which contains the basic information of the Node.

**Table 1 - The data structure of a node**

| Prefix | Node |
|---|---|
| Fields | IP Address |
| | MAC Address |
| | Switch Port |
| | Bandwidth |
| | Additional Node Information |

## *4.4 Chaincode Design and Deployment*

Contracts are used to implement all the executable business logic of the application; therefore, smart contracts are used to execute assets such as create, read, update, or delete activities on the distributed ledger. Depending on the architecture and programming language used, chain codes can be various functions or even different files (or classes). In this case, smart contracts have been implemented using JavaScript. One function has been written for each capability of the proposed system to match the interface with the HLF network to maintain the simplicity and modularity of the architecture.

The entity on which network transactions are expected to take place is often the focus of smart contract development. When an admin admits a new node into the network , the contract AddNode() assists in creating a new transaction in the distributed ledger. The contracts UpdateNode() used to update the configuration of the node. If the node shows some mischievous behavior then the admin has rights to remove that node from the network,the contract RemoveNode() assists in creating a new transaction in the distributed ledger. The HLF blockchain network has a function called GetHistory, which allows users to retrieve the history of transactions that have taken place on a certain entity, and is advantageous since the global state simply keeps track of the most recent or updated state of a record.

There are four processes involved in deploying chaincode to the HLF network: packaging the chaincode; peer installation of chaincode; approving a chaincode definition for a channel; and committing a chaincode definition. Once the HLF network is operational, all four processes may be carried out simultaneously by using the deployCC command. The route of our chaincode and the language in which it is written must both be passed with the appropriate flags.

**Table 2 - The data structure of transaction**

| Prefix | Transaction |
|---|---|
| Fields | ID |
| | Admin |

| |
|---|
| Node |
| Bandwidth |
| Timestamp |

1) Smart Contract Design: The process of the adding node is shown in Algorithm 1. First, it will check if the same node exists in the network if it is already present meaning the coming node might be malicious so that it will not be added in the network and the transaction status is updated to rejected and it is recorded on blockchain ledger. And if the node is not already present in the network then the admin will check the required information and decide whether to add the node in network or not and update the transaction status accordingly and the transaction is recorded on ledger.

**Algorithm 1** Admit the Node to network

**Input:** VerifiedNodes: list of nodes in the network, totalNodes: total number of nodes in the network, node: node who wants to join the network, admin: admin of system

```
1: TxID = generateID()
2:  nt = new Transaction
3:  nt.ID = TxID
4:  nt.node = node
5: nt.admin = admin
6: nt.state = "initiated"
7: if node is in verifiedNodes then
8:          nt.state = "rejected"
9:          PutState(nt , TxID)
10: else
11:         if node is valid then
12:             nt.state = "admitted"
13:          verifiedNodes.append(node)
14:             totalNodes = totalNodes+1
15:             PutState(nt , TxID)
16:         else
17:             nt.state = "rejected"
18:             PutState(nt , TxID)
19:         end if
20: end if
```

The process of removing the node from the existing network is shown in Algorithm 2. If any unusual activity is found in the node's behavior then admin has rights to remove the node from the network.

**Algorithm 2:** Remove Node from network

**Input:** VerifiedNodes: list of nodes in the network, totalNodes: total number of nodes in the network, node: node to be removed, admin: admin of system

**Output:**

```
1: TxID = generateID()
2:  nt = new Transaction
3:  nt.ID = TxID
4:  nt.node = node
5: nt.admin = admin
```

```
6: nt.state = “initiated”
7: **if** node is in verifiedNodes **then**
8:         nt.state = “removed”
9:     verifiedNodes.remove(node)
10:        totalNodes = totalNodes-1
11:        PutState(nt , TxID)
12: **else**
13:        nt.state = “rejected”
14:        PutState(nt , TxID)
15: **end if**
```

### *4.5. SDK's (Software Development Kit)*

Hyperledger Fabric Client SDK offers APIs (application programming interfaces) for interacting with smart contracts, adding transactions to a ledger, and

- Fabric-Ca-Client: The fabric-ca offers APIs for participants (i.e., admin, patient, and doctor) to sign up and enroll to create trustworthy identities on the blockchain network. The package develops a new CA client that can communicate with the hospital’s CA server to enroll and register participants.
- Fabric-Network: The APIs needed to connect to the Fabric network, submit transactions for queries, or alter the ledger are included in this package. Gives access to APIs that may be used to manage the wallet that is used to maintain identities and to build a connection profile using the connection profile JSON that is produced when a CA is formed.
- Wallet: The wallet serves as an identity and is a crucial component of the Hyperledger Fabric SDK, as was already explained. It is used to save the Fabric metadata, the private key in an identity file that has been authorized by a certificate authority, and the public key. Wallets come in a variety of forms, including file, in-memory, and database wallets. The file type is used in the creation of the patient data management system. The Gateway class utilizes the mspID and the type stored in the user’s wallet during connection formation to connect to the network and check the user’s access privileges to the channel in question.
- Use of the API: The Gateway class, included in the fabric network package, is the primary class that enables the communication between the Fabric SDK and the network. When an assert has been performed, a gateway or link is established to a peer or user inside the blockchain network, allowing access to the chaincode and channels. The registerUser() function in the SDK is called when the Register user/admin action is executed, and it then collects the user’s information, including their ID. A wallet is then generated and an identity file for that user is added to it after obtaining the network parameters. This acts as the user’s identification while attempting to enter the Fabric network. The enterprise organizations departments administrative staff must complete this procedure, and they must also register as administrators in the same way users must first register. When a node gets added to the network, the smart contract’s AddNode() method is called, which creates a new record for that Node with the necessary data. To prevent each user from manipulating the data, some APIs are only available to specific users. Updating patient health records, on the other hand, is only available to admin because it updates the Nodes configuration.

### *4.6. Client/Front-End Development*

- Admin Login: Each department of enterprise organization that joins the network receives an admin. When adding the department to the network, administrative information must be present. The appropriate department CA configuration’s fabric-ca-server-config.yaml file contains the admin information (username and password).
- User Login: The user node in the network can login into the system to view the network. The user has to provide the valid credentials in order to successfully login to the system.
- Display network (Admin): Once the admin signs into the system, a dashboard is shown, which includes a list of every node waiting to join in the network with their features. The admin can admit or deny the Node based on its attributes.
- Display network (User): Once the user has successfully logged into the system, it will be able to view the different nodes present in the network along with their respective node description.
- Admit Node: The admin can acknowledge the node by admitting it into the network.
- Deny Node: The admin can deny the entry of node in the network based on certain attributes.

- View History: The admin can see all the transaction history on the ledger.

## Conclusion

The paper thus proposes a blockchain-based network management system which is capable of managing and monitoring blockchain based networks using hyperledger fabric as a framework on the basis of permissions to view or connect nodes to the network. The use of Blockchain which is decentralized, distributed and highly immutable makes the system tamper-proof and almost impossible to hack. Also it is desirable when it comes to such technology as it is essential when it comes to single point of failure and also attacks on sensitive data.

### *Acknowledgements*

We would like to express our special thanks to the sponsor of our project Mr. Harshal Haridas, Chief CyberSecurity Architect, Honeywell and our project guide Prof. Pujashree Vidap, Pune Institute of Computer Technology for their continuous support and guidance during the project work. We thank them for this golden opportunity to do this wonderful project from the first time we encountered this subject to the final revision of this thesis. Their excitement, knowledge, and meticulous attention to detail served as an inspiration and kept us on track.

### References

Sarwar, Sadiq. "Network Management Using an Ethereum Blockchain." (2020).

Prashar, Deepak, et al. "Blockchain-based automated system for identification and storage of networks." Security and Communication Networks 2021 (2021): 1-7.

Krstić, Marija, and Lazar Krstić. "Hyperledger frameworks with a special focus on Hyperledger Fabric." Vojnotehnički glasnik/Military Technical Courier 68.3 (2020): 639-663.

Monrat, Ahmed Afif, Olov Schelén, and Karl Andersson. "Performance evaluation of permissioned blockchain platforms." 2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE). IEEE, 2020.

Polge, Julien, Jérémy Robert, and Yves Le Traon. "Permissioned blockchain frameworks in the industry: A comparison." Ict Express 7.2 (2021): 229-233.

Khan, Shafaq Naheed, et al. "Blockchain smart contracts: Applications, challenges, and future trends." Peer-to-peer Networking and Applications 14 (2021): 2901-2925.

Yamashita, Kazuhiro, et al. "Potential risks of hyperledger fabric smart contracts." 2019 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE). IEEE, 2019.

Li, Wenjuan, et al. "BlockCSDN: towards blockchain-based collaborative intrusion detection in software defined networking." IEICE TRANSACTIONS on Information and Systems 105.2 (2022): 272-279.

Mohsin, Ali H., et al. "Blockchain authentication of network applications: Taxonomy, classification, capabilities, open challenges, motivations, recommendations and future directions." Computer Standards & Interfaces 64 (2019): 41-60.

Huang, Jiansen, Hui Li, and Jiyang Zhang. "Blockchain based log system." 2018 IEEE International Conference on Big Data (Big Data). IEEE, 2018.

Rakib, Mohammad Habibullah, et al. "Towards Blockchain-Driven Network Log Management System." 2020 IEEE 8th International Conference on Smart City and Informatization (iSCI). IEEE, 2020.

Zheng, Liwen, et al. "Automatic discovery mechanism of blockchain nodes based on the Kademlia algorithm." Artificial Intelligence and Security: 5th International Conference, ICAIS 2019, New York, NY, USA, July 26-28, 2019, Proceedings, Part I 5. Springer International Publishing, 2019.

Ndzimakhwe, M.; Telukdarie, A.; Munien, I.; Vermeulen, A.; Chude-Okonkwo, U.K.; Philbin, S.P. A Framework for User-Focused Electronic Health Record System Leveraging Hyperledger Fabric. *Information* **2023**, *14*, 51. https://doi.org/10.3390/info14010051