Healthcare Ethereum Blockchains

Manju Mohan, Vivek Mudgal ,Shashank Tomer , Pranshu Ranakoti, Benjamin John Oommen, Aditya Sahu
Technergize – Cyber Security
CSIS - 2018

Abstract

Blockchain for health care has the potential to standardize secure data exchange in a less burdensome way. The main aim of the project is to attain higher levels of medical records security using Ethereum Blockchain methods. Medical information and casualties of people in a region can be monitored every week. It also helps to know actual rate of spread of diseases, especially in backward areas. To spot blood donors at the right time and required blood group. Finally, it also helps organ donation, to incentivize medical stakeholders (researchers, public health authorities, etc.) to participate in the network as blockchain miners. This provides them with access to aggregate, anonymous data as mining rewards, in return for sustaining and securing the network via Proof of Work.

Keywords **– Blockchain; Ethereum; medical records; stakeholders; miner; Proof of Work**

I. INTRODUCTION

Today humans manually attempt to reconcile medical data among clinics, hospitals, labs, pharmacies, and insurance companies. It does not work well because there is no single list of all the places data can be found or the order in which it was entered. We may know every medication ever prescribed, but it can be unclear which medications the patient is actually taking now. Further, although data standards are better than ever, each health records stores data using different workflows, so it is not obvious who recorded what, and when.

Through the HIPAA Privacy Rule, providers can take up to 60 days to respond (not necessarily to comply) to a request for updating or removing a record that was erroneously added. Beyond the time delay, record maintenance can prove quite challenging to initiate as patients are rarely encouraged and seldom enabled to review their full record. Patients thus interact with records in a broken manner that reflects the nature of how these records are managed. Interoperability challenges between different provider and hospital systems pose additional barriers to effective data sharing. This lack of coordinated data management and exchange means health records are fragmented, rather than cohesive. Patients and providers may face significant hurdles in initiating data retrieval and sharing due to economic incentives that encourage “health information blocking”.

Imagine that every health records sent updates about medications, problems, and allergy lists to an open-source, community-wide trusted ledger, so additions and subtractions to the medical record were well understood and auditable across organizations. Instead of just displaying data from a single database, the health records could display data from every database referenced in the ledger. The end result would be perfectly reconciled community-wide information, with guaranteed integrity from the point of data generation to the point of use, without manual human intervention.

In this work, an ethereum blockchain structure is applied to Electronic Medical Records. The blockchain uses public key cryptography to create an append-only, immutable, time stamped chain of content. Copies of the blockchain are distributed on each participating “node” in the network. The Proof of Work algorithm used to secure the content from tampering depends on a “trustless” model, where individual nodes must compete to solve computationally intensive puzzles (hashing exercises) before the next block of content can be appended to the chain. These worker nodes are known as “miners”, and the work required of miners to append blocks ensures that it is difficult to rewrite history
on the blockchain.

The mining proof-ofwork (PoW) exists as a cryptographically secure nonce that proves beyond reasonable doubt that a particular amount of computation has been expended in the determination of some token value n. It is utilised to enforce the blockchain security by giving meaning and credence to the notion of difficulty. However, since mining new blocks comes with an attached reward, the proof-of-work not only functions as a method of securing confidence that the blockchain will remain canonical into the future, but also as a wealth distribution mechanism.

For both reasons, there are two important goals of the proof-of-work function; Firstly, it should be as accessible as possible to as many people as possible. The requirement of, or reward from, specialised and uncommon hardware should be minimised. This makes the distribution model as open as possible, and, ideally, makes the act of mining a simple swap from electricity to Ether at roughly the same rate for anyone around the world. Secondly, it should not be possible to make super-linear profits, and especially not so with a

high initial barrier. Such a mechanism allows a well-funded adversary to gain a troublesome amount of the network's total mining power and as such gives them a super-linear reward as well as reducing the network security.

II. RELATED WORK

Recent work by MIT Labs, MedRec prototype enables patients with one-stop-shop access to their medical history across multiple providers: smart contracts on an Ethereum blockchain. These contract data structures are stored on the blockchain and associate references to disparate medical data with ownership and viewership permissions and record retrieval location. This provides an immutable data-lifecycle log, enabling later auditing. This include a cryptographic hash of the record in the smart contract to establish a baseline of the original content and thus provide a check against content tampering.

MedRec facilitates reviewing, sharing and posting of new records via a flexible user interface, designed to reflect best-practices from the Blue Button health record competition.

Interoperability challenges between different provider systems pose significant barriers to effective data sharing. Patients face hurdles in authorizing data exchange (with other consulting physicians or even family members) due to the lack of a common interface or standard system that orchestrates record access across databases. Healthcare apps can serve thousands or millions of participants, which may incur enormous overhead when large volumes of data are stored in a blockchain–particularly if data normalization and denormalization techniques are not carefully considered. Not only is it costly to store these data, but data access operations may also fail if/when the cost exceeds the Ethereum gas limit.

The MedRec system uses database "Gatekeepers" for accessing a node's local database governed by permissions stored on the MedRec blockchain. Peterson et al. [Peterson et al. 2016] presented a healthcare blockchain with a single centralized source of trust for sharing patient data, introducing "Proof of Interoperability" based on conformance to the FHIR protocol as a means to ensure network consensus.

III. SYSTEM IMPLEMENTATION

A. Overview

Our system is an extended version of MedRec. Blockchain technology supports the use of smart contracts, which allow us to automate and track certain state transitions (such as a change in viewership rights, or the birth of a new record in the system). Via smart contracts on an Ethereum blockchain, we log patient-provider relationships that associate a medical record with viewing permissions and data retrieval instructions (essentially data pointers) for execution on external databases. We include on the blockchain a cryptographic hash of the record to ensure against tampering, thus guaranteeing data integrity. Providers can add a new record associated with a particular patient, and patients can authorize sharing of records between providers. Acquires medical information and causalities of people in a region and it can be monitored every week. Helps to know about the actual rate of spread of diseases, especially in backward areas. Add on the feature that helps organ donation. To spot the blood donors at the right time and required blood group. It also helps to share the data with insurance company and organisations like World Health Organisation (WHO). For large storage of data IPFS (InterPlanetary File System) is used. We incentivize medical stakeholders (researchers, public health authorities, etc.) to participate in the network as block chain "miners". This provides them with access to aggregate, anonymous data as mining rewards, in return for sustaining and securing the network via Proof of Work.

In the following sections, the design principles of our system and its implementation are presented.

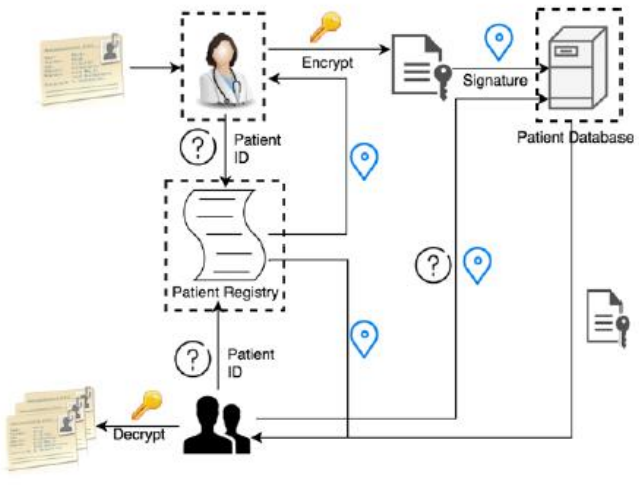

Figure 1: Structure and Workflow of Patient-Doctor system

B. Blockchain and Ethereum

A blockchain is a decentralized computing architecture that maintains a growing list of ordered transactions grouped into blocks that are continually reconciled to keep information up-to-date, as shown in Figure 1.

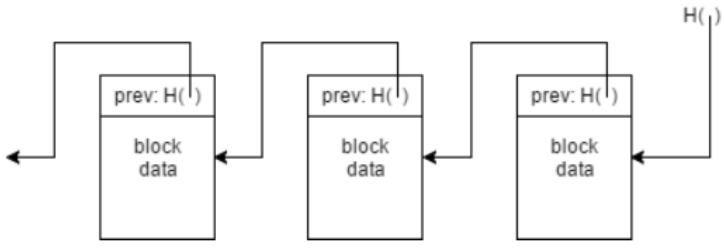

Fig. 2. Blockchain Structure: a Continuously Growing List of Ordered and Validated Transactions

Only one block can be added to the blockchain at a time. Each block is mathematically verified (using cryptography) to ensure that it follows in sequence from the previous block. The verification process is called "mining" or Proof of Work [Nakamoto 2012], which allows network nodes (also called "miners") to compete to have their block be the next one added to the blockchain by solving a computationally expensive puzzle. After the nodes compute the challenge (Merkle tree + hash of the previous block) they need to and a proof (string) which when concatenated with a challenge and hashed using SHA-256 gives an output which has specific amount of leading 0s. E.g when we combine challenge and proof and hash it, the output should have for example 40 leading 0s. This is very challenging problem because SHA-256, like other hash functions, has the property that it is impossible to compute the input for a given output, and also similar input strings have completely different hashes.

For example:

SHA-256 of cat = 77af778b51abd4a3c51c5ddd97204a9c3ae614ebccb 75a606c3b6865aed6744e

SHA-256 of Cat = 48735c4fae42d1501164976afec76730b9e5fe467f68 -

0bdd8da_4bb77674045

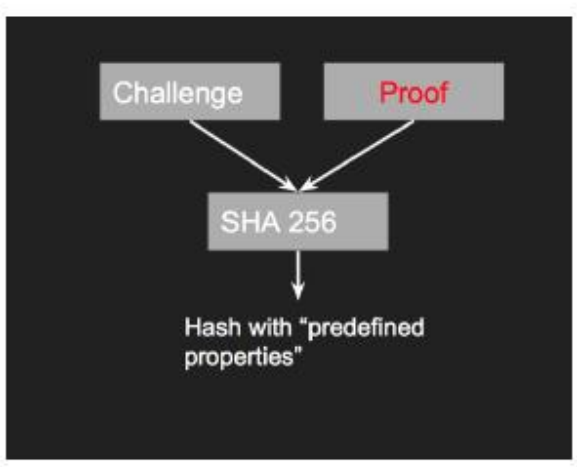

Figure 3. Bitcoin Proof of Work Concept

The winner then announces the solution to the entire network to gain a mining reward paid via cryptocurrency. The mining process combines cryptography, game theory, and incentive engineering to ensure that the network reaches consensus regarding each block in the blockchain and that no tampering occurs in the transaction history. All transaction records arekept in the blockchain and are shared with all network nodes. This decentralized process ensures properties of transparency, incorruptibility, and robustness (since there is no single point of failure).

The Ethereum blockchain is a distributed state transition system, where state consists of accounts and state transitions are direct transfers of value and information between accounts. Two types of accounts exist in Ethereum:

(1) externally owned accounts (EOAs), which are controlled via private keys and only store Ethereum’s native value-token "ether" and (2) smart contract accounts (SCAs) that are associated with contract code and can be triggered by transactions or function calls from other contracts.To protect the blockchain from malicious attacks and abuse (such as distributed denial of service attacks in thenetwork or hostile infinite loops in smart contract code), Ethereum also enforces a payment protocol, whereby afee (in terms of "gas" in the Ethereum lexicon) is charged for memory storage and each computation executed in acontract or transaction. These fees are collected by miners who verify, execute, propagate transactions, and then group them into blocks.

Testnets simulate the Ethereum network and EVM. They allow developers to upload and interact with smart contracts without paying the cost of gas. Smart contracts must pay gas for their computations on the Ethereum network. However, testnets provide free

or unlimited gas. That allows developers to test contracts without having to pay real money for their execution. Testnet nodes come in two main flavors:

1. *lightweight* Ethereum nodes used for small scale local testnets. Ex: ethereumjs-testrpc - Useful for early stage contract development. This is what you will be using most of the time. Calls to lightweight testnet nodes complete very quickly and provide good error messages.

2. *heavyweight* Ethereum nodes used for large scale networked testnets. Ex: Geth - Useful for connecting to public networked testnets. The most popular public testnet is called Ropsten which is useful during later stage contract development. Connecting to Ropsten through Geth simulates the real Ethereum network. That makes it appealing for mature contracts that you want to battle test.

C. *Smart Contracts*

Ethereum smart contracts can be built in a Turing complete programming language, called Solidity. This contract language is compiled by the Ethereum Virtual Machine (EVM), which enables the Ethereum blockchain to become a platform for creating decentralized apps (DApps) that provide promising solutions to healthcare interoperability challenges. Solidity has an object-oriented flavor and is intended primarily for writing contracts in Ethereum. A class in Solidity is realized through a "contract," which is a prototype of an object that lives on the blockchain. called. Contracts may contain persistent state variables that can be used as data storage and functions that interact with the states. Several smart contracts are made for the different functionalities such as for blood donation with features to search the donar by location, name, blood group,age and request for blood. And, similar features to organ donation, searching by name, organ, age, location(PHC). Patient – Doctor record manager includes patient details, doctor details, medical prescription and other reports. It also includes Fitbit health tracker, tracking allergies, medication procedure. Also biometrics is used for user authentication.

D. Truffle

Truffle is a development environment, testing framework and asset pipeline for Ethereum, aiming to make life as an Ethereum developer easier. It is one of the most widely used IDEs in the Ethereum community. Developers can use it to build and deploy DApps for testing purposes with many features that make it more attractive to users with a Web 3.0 dev background. Automated contract testing with Mocha and Chai. A configurable build pipeline that supports both web apps and console apps. Generators for creating new contracts and tests (like rails generate) Instant rebuilding of assets during development (truffle watch) Console to easily work with your compiled contracts (truffle console)
Script runner that lets you run JS/Coffee files with your contracts included (truffle exec) Contract compilation and deployment using the RPC client of your choice.Support for JavaScript, CoffeeScript, SASS, ES6 and JSX built-in.

All the above mentioned smart contracts are compiled using truffle using windows powershell.

E. IPFS or InterPlanetary File System

IPFS or InterPlanetary File System is a peer-to-peer distributed system that connects all networks using the same system of files. Specifically, IPFS is a content-addressable, P2P hypermedia distribution protocol. IPFS is the protocol to upgrade the web. There is no trust between nodes, and there is no single point of failure. IPFS is a distributed version-controlled filesystem of hashes. The fundamental principle of IPFS is that all data is part of the same Merkle DAG, a content-addressed block storage model with content-addressed hyperlinks. It doesn't matter where your information is located. As long as the address structure is standardized, this information could be accessible from any platform, database or system. Unlike traditional networked provider-to-provider systems, with IPFS there are no privileged nodes. IPFS is the result of a mashing of distributed hash tables (DHT), BitTorrent, Git and Self-Certified Filesystems. The IPFS protocolalso contains a set of seven subprotocols or principles that synthesize prior peer-to-peer concepts that assemble to form the backbone of IPFS.

1. Identities: peer node identification.
2. Network: govern the connections to other peers.
3. Routing: information relevant to locate peers and stored objects.
4. Exchange: protocol managing how blocks are distributed.
5. Objects: Merkle-DAG, content-addressable immutable objects and links.
6. Files: a versioned controlled file system.
7.Naming: mutable naming (permanent objects) with content-addressed DAG objects.

8. Applications: any application running over IPFS to leverage the new connected web.

The IPFS stack is a combination of eight elements: identity (of each node) + network + routing (distributed hash tables) + exchange (BitTorrent) + merkledag (git) + naming (Self-Certified Filesystems) + applications (web). Together these elements form the IPFS stack, a stack that will be used to standardize the accessibility of medical records. IPFS is the most impactful data structure.

F. Connecting to Ethereum Client

Ethereum clients expose a number of methods over JSON-RPC for interacting with them from within an application. However, interacting directly over JSON-RPC passes on a number of burdens to the application developers,
such as:

• JSON-RPC protocol implementation
• Binary format encoding/decoding for creating and interacting with smart contracts
• 256 bit numeric types
• Admin command support - e.g. create/manage addresses, sign transactions

web3.js

This is the Ethereum compatible JavaScript API which implements the Generic JSON RPC spec. It's available on npm as a node module, for bower and component as an embeddable js and as a meteor.js package. web3j is a lightweight Java library for integrating with clients (nodes) on the Ethereum network.Web3js is automatically injected with the help of Mestamask or Mist.
Core features:

• Interaction with Ethereum clients over JSON-RPC via Java types
• Supports all JSON-RPC method types
• Supports all Geth and Parity methods for managing accounts and signing transactions
• Sending of client requests both asynchronously and synchronously
• Auto-generation of Java smart contract function wrappers from Solidity ABI files

Currently, the go-ethereum and Parity clients are supported.

IV. FUTURE WORK AND CONCLUSION

As we look to take our system from a prototype to a meaningful tool for enterprise, government and patient use, we have identified several thrusts of future work. First, we continue our process of actively engaging with healthcare stakeholders across the industry, from hospitals and provider offices, to pharmaceutical companies, to insurance companies, to healthcare startups, U.S. Government institutions and World Health Organisation. In future, the system will be able to share the information with insurance company. Also to provide a secure way for sponsors to donate money for people in need without the intervention of others (more trust in charity).

The system provides a proof-of-concept system, demonstrating how principles of decentralization and blockchain architectures could contribute to secure, interoperable EHR systems. Using Ethereum smart contracts to orchestrate a content-access system across separate storage and provider sites, authenticates medical record access while providing patients with comprehensive record review, care auditability and data sharing.

V. ACKNOWLEDGMENT

The authors would like to thank IEEE SAC, Kolkata Section for their support and collaborating on the project.

VI. REFERENCES

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

[2] Morgen Peck - IEEE Future Directions Blockchain Initiative white paper blockchainincubator.ieee.org

[3] DR. Gavin Wood - Ethereum: A secure decentralised generalised transaction ledger

[4] Kish, Leonard J., and Eric J. Topol. "Unpatients [mdash] why patients should own their medical data." *Nature biotechnology* 33, no. 9 (2015): 921-924.

[5] Mandl, Kenneth D., David Markwell, Rhona MacDonald, Peter Szolovits, and Isaac S. Kohane. "Public Standards and Patients' Control: how to keep electronic medical records accessible but private." *Bmj* 322, no. 7281 (2001): 283-287.

[6] Ariel Ekblaw*, Asaph Azaria*, John D. Halamka, MD†, Andrew Lippman* - A Case Study for Blockchain in Healthcare: “MedRec” prototype for electronic health records and medical research data

[7] Ethereum community- Ethereum Homestead Documentation. Release 0.1

[8] Christian Muller and Dalmir Hasic, Department of Computer Sciences University of Salzburg- Blockchain: Technology and Applications

[9] Peng Zhang, Vanderbilt University, Nashville, TN Jules White, Vanderbilt University, Nashville, TN Douglas C. Schmidt, Vanderbilt University, Nashville, TN Gunther Lenz, Varian Medical Systems, Palo Alto, CA- Design of Blockchain-Based Apps Using Familiar Software Patterns to Address Interoperability Challenges in Healthcare

[10] https://medium.com/@PeterBNichol/an-ipfs-addressable-storage-model-for-healthcare-with-blockchain-740fabc1a062

[11] https://solidity.readthedocs.io/en/develop/index.html