Healthcare-Team-5 (HealthChain)

Health-Chain

Goal: Provide better Health care guidance to patients using Blockchain Technology (Considered 3 related Scenarios)

Provided by: Shekufeh Shafeie(Pro Member)

Other Pro- Co-worker-Team Members Name:

-Roop Patel

-Dat Le

July 2021

Scenarios of using blockchain in different healthcare settings:

We think about all different but related scenarios as follow

Scenario 1: Primary patient care;

Scenario 2: Data aggregation for the research purposes and incentive the patient for sharing the data;

Scenario 3: Connecting different healthcare players for better patient care;

Better means: By employing blockchain technology, our solution allows to facilitate the consent management and speed up data transfer. We developed a chaincode that allows a patient to easily impose fine-grained access control policy for his data and enables efficient data sharing among clinicians/ Health care providers.

Senario4: Remote Health System for Patient Monitoring.

Senario5: Digital Passport as response to Covid-19

***Name of Product: Provide better Health care guidance to patients**

-Product Goal:

Provide better Health care guidance to patients- including Scenario 1: Primary patient care

What Better means in our Product/ Application are as follows:

-providing accountability of access, maintaining the complete and updated information with a verifiable record of the provenance, including all accesses/sharing/usages of the data.

-It supports privacy-preserving data sharing and management

-Motivate users/patients to be part of the research by building up incentives for users to share their profile data, in terms of rewards (micro-payments or credits). In this way, users become owners of their data and can decide how their data is collected and used, as well as shared.

-Product Specification:

- A blockchain-based records system would **empower patients to retain full control over** their data, choosing when, and to whom, to grant permission to access their information.

- Capable of storing and managing patient information in a verifiable manner, publicly accessible in real-time by anyone in the healthcare service provider chain (if authorized by the patient)
-Provide an up-to-date, comprehensive picture of patient health, enabling physicians, pharmacies and other providers to provide better health care guidance to patients.

- Prevent the double-spending of digital currency could be used **to prevent double-filling of pharmaceutical prescriptions**, while **blockchain private keys** could be implemented as a way to authenticate identity and validate insurance coverage.
-patient data can be stored directly on the blockchain (with privacy protections) or off-chain (and managed through on-chain access rights). We prefered off-chain. It's advantages are as follows: blockchain-based system of electronic health records would not only enable healthcare providers to take a holistic view of patient information and produce better health outcomes; it would rebalance the information paradigm and put the 'personal' back in 'personal health'.
-That could be giving permission to a healthcare provider in a **physician-patient setting** or could extend to **commercial uses** – authorizing access to your data by a pharmaceutical manufacturer conducting clinical research in exchange for you receiving micropayments of digital currency. For example, using the Bitcoin SV blockchain, EHR Data is developing the world's first global electronic health record, which will enable individuals to securely own, control and earn money from their personal medical information, while also furnishing health care providers and researchers with better real-time access to data.
-The platform can support a variety of use cases to produce better health outcomes, ranging from opioid drug to Covid-19 tracking, all while ushering in a new era of personal power for patients.
- Governments and healthcare professionals can be involved on administering the program. A universal electronic health record would ensure that regardless of where each Covid-19 vaccine course was administered, that the correct type and dosage was given at the right intervals. This individual, immutable record of vaccination could subsequently be used for digital 'passports' that verify admissibility to **public places or transport as third party**.

All the Entities and Parties that are involved in the Health care high level can be considered according to the below figure.

Functions and grants: store, exchange, and view information

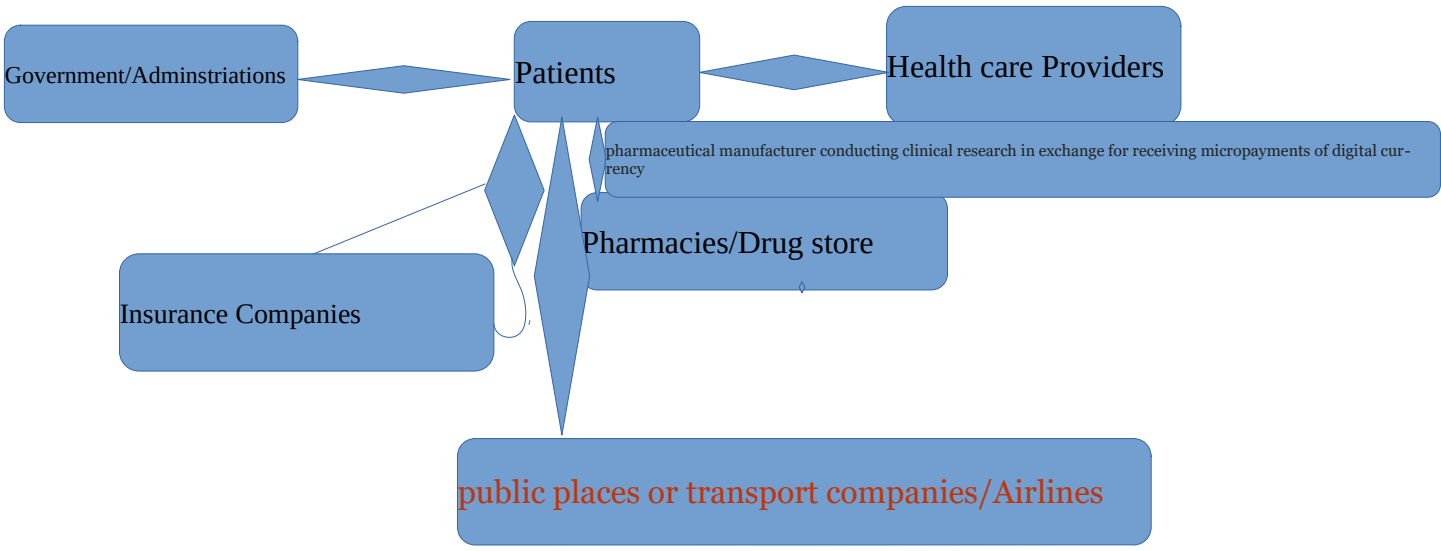


Figure0: All the Entities and Parties that are involved in the Health care high level

Some of the Primary Healthcare Providers can be supposed as follows:
Primary Care Physician, Nurse Practitioners, Family Practice Doctors, Internal Medicine Doctors, Gynecologist or Obstetrician, Geriatricians and Pediatricians. However, we just simply considered Doctors as of now.

We propose a platform for user modeling with blockchains that allows users to share data without losing control and ownership of it and applied it to the domain of required Healthcare services. Our new platform provides solution to three important problems: ensuring privacy and user control, and incentives for sharing.

It tracks who shared what, with whom, when, by what means and for what purposes in a verifiable fashion. We presents a case study of applying the framework for a healthcare provider like a hospital system and staff (Doctor) as one of the enterprise nodes of Multichain which collects users' profile data and allows users to receive rewards while sharing their data with other third parties and Healthcare service providers according to their privacy preferences expressed in smart contracts.

The user data from the repository is converted into an open data format and shared via stream in the blockchain so that other nodes can efficiently process and use the data. The smart contract verifies and executes the agreed terms of use of the data and transfers digital tokens as a reward to the user. The smart contract imposes double deposit collateral to ensure that all participants act honestly.

Required Background

This section presents an overview of blockchain and smart contracts.

-Blockchain

Blockchain is a data structure used to create a public or private distributed digital transaction ledger which, instead of resting with a single provider, is shared among a distributed network of computers.

-MultiChain

MultiChain is a private permissioned blockchain that provides the privacy and control required in an easy to configure and deploy package (Greenspan, 2013).

-Ethereum

Ethereum is an open-source, public permissionless blockchain to create decentralized applications (dapps) where users interact with the online services in a distributed peer-to-peer manner that takes place on a censorship-proof foundation. Developers can design interfaces and business logic with any of the known programming languages and tools.

-Smart Contracts

The smart contracts are instances of contracts deployed on the Ethereum blockchain (Buterin, 2015) although the term was originally coined earlier (Szabo, 1997) in the context of electronic commerce protocols between strangers on the Internet. A smart contract stores the rules which:

1. Negotiate the terms of the contract,

2. Automatically verify the contract, and

3. Execute the agreed terms.

Data Authorship as an Incentive to Data Sharing

Proposed Platform

The privacy-related legislation, General Data Protection Regulation (GDPR) as of May 25, 2018, regulates the processing of the personal data of the individuals and demands personal data erasure. However, data on the blockchain are always immutable. This could be a challenge while adopting blockchain as part of the solution. Therefore, we have a general architecture of the user data sharing system based on blockchain and off-chain data storage as shown in Figure 01. The general architecture ensures that the actual user data is never exposed to the blockchain. The user data is first hashed and encrypted before uploading it into the off-chain storage. The data owners from their client applications can directly store them on the off-chain storage. The terms and conditions regarding the access to user data are encoded in smart contracts along with the metadata and hash of the data and published on a blockchain platform (Ethereum). The hashes of the data on the blockchain prevent the middleware from tampering the data. The content-based addressing makes hashes of data serve as their identifier for retrieval. When the data consumer invokes the smart contracts for accessing the user data, only the successful invocation of the contracts results in the release of the key for decrypting the user data. Then the trusted program (Ning et al., 2018) extracts the hash from the blockchain, uses this hash to retrieve the data from the off-chain storage, decrypts it and releases the data to the data consumer while settling the incentives for the data owner. Blockchain and smart contracts support users by giving the users full transparency over who accesses their data, when and for what purpose, allowing the users to specify a range of purposes of data sharing, kinds of data that can be shared, and classes of applications/companies that can access the data, and providing an incentive to the users for sharing their data (in terms of payment for the use of the data by applications, as specified by the contracts).

The general architecture presents the underlying off-chain user data storage mechanism which could be a centralized data store hosted by a trusted party. When trust resides within a centralized service provider for all the storage and management of data, it is hard to mitigate the various risks, for example, of data being misused, hacked or sold to any other bodies without user consent and even destroyed when the company defaults. Therefore, we present a new platform with separate private permissioned blockchain called MultiChain as a solution to both on-chain and off-chain data storage, encryption, hashing and tracking of data, together with Ethereum (for access control). Off-chain blockchain implementation with user data storage can be successfully achieved with the limited number of peers in the MultiChain (Greenspan, 2013). Users can optionally store any published data in off-chain that saves storage place and bandwidth.

The similar idea of storing off-chain data and accessing them via MultiChain has also been proposed in our previous research works (Shrestha et al., 2017; Shrestha and Vassileva, 2018) and others' works such as Yang et al. (2019) and Ferrer-Sapena and Sánchez-Pérez (2019). MultiChain nodes handle key operations such as hashing and encrypting the user data, storing the encrypted file locally (outside of blockchain), committing the hash of the file on the blockchain, searching the required data, verifying the data and delivering the data.

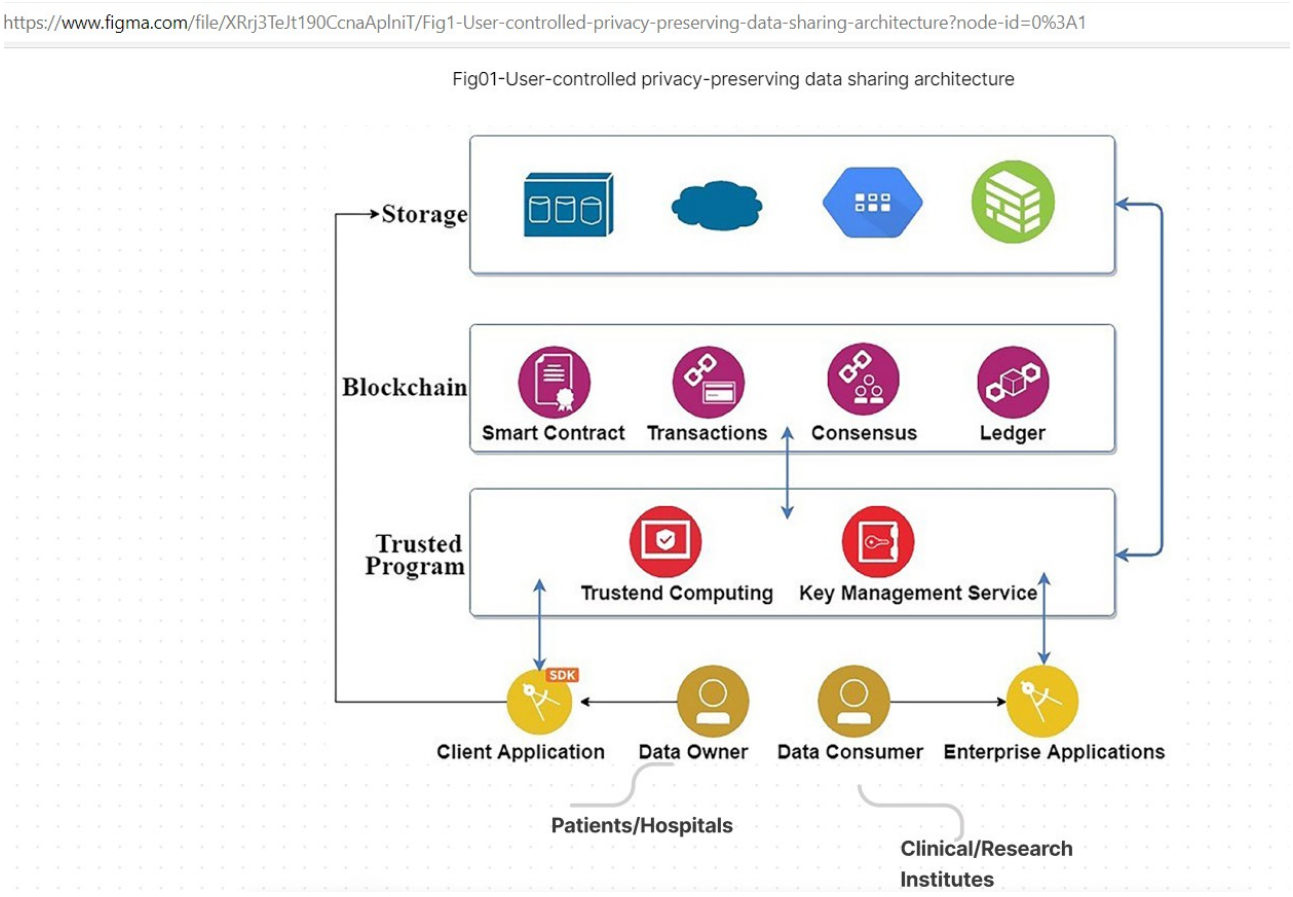


Figure 01- User-controlled privacy-preserving data sharing architecture.

https://www.figma.com/file/XRrj3TeJt190CcnaAplniT/Fig01-User-controlled-privacy-preserving-data-sharing-architecture?node-id=0%3A1

We, therefore, introduce two blockchains: MultiChain to share user profile information among enterprises in their private network, and Ethereum to store securely the access-control policies as smart contracts. **A user can register into the system by providing her basic profile information and activating the smart contracts on an Ethereum network node.** The **Ethereum node** automates the functionality to support the user-controlled privacy regarding (a) with whom the users' data will be shared, and (b) how the user will be incentivized once her data are being shared with other third parties. The communication with the Ethereum network node is made directly with our own hosted enterprise nodes. Since the smart contracts are stored on Ethereum, the users can have their own ether addresses which they use to safely store ether (ETH) in their own cold wallet (offline wallet) as well. **Once the users' data are being used by any other participating organization, the corresponding users (data provider) will be incentivized with ether.** We explain the workflow in section User Incentives for Sharing.

For sharing the user data among enterprises, MultiChain blockchain is installed on each participating enterprise end, which can publish the user data as items into the stream and share them in the network according to the smart contracts set by the users.

Data Sharing Solution as part of Scenario1:

Figure 02 below presents a functional model of the new platform. It will be explained below using as an example Healthcare data sharing system in the context of the online Health care system industry. The parties involved in our health care data sharing scenario are: imaginary Hospital(s), Drug stores and pharmacies, and Research Institutes(Pharmacy manufactures).

With valuable data created for patients at every stage of their journey, healthcare industries can leverage this opportunity to collect and share these data and offer better-personalized services to them as well. We are specifically looking into the blockchain and smart contract technologies to offer an effective incentives solution to the customers/patients for sharing their data, and track who shared what, with whom, when, by what means and for what purposes in a verifiable fashion in this project. We take user/patient data as the standard data required by hospital systems that include Patient-ID, Patient name, Patient Date of Birth, Age, Race, Gender, Blood Type, Existing Medical Condition(History of Disease), Nationality, Home address, contact email address, the purpose of visit the Hospital/Health Care Provider/ Doctor and the duration of stay in the hospital (check-in and check-out dates). The Hospital Admission/Service System (Imaginary Hospital) has the first node in the network that is responsible for the collection of the user/patient data. By default, this node acts as an admin who can grant other nodes admin privileges too. In our case, the permissions for other nodes:

“connect,” “send,” “receive,” “issue,” “create,” “mine,” “admin,” and “activate” are set by the first admin node. These permissions allow the nodes to perform certain operations. With the *connect* permission, the node can see the blockchain content. The *send* and the *receive* permissions allow a

node to send and receive funds, respectively. The *create* permission allows the node to create a stream for sharing data and this is one of the key features of the proposed data sharing solution. (The streams' creation and items publication are explained in more detail below.) The *issue* permission allows a node to issue assets whereas *mine* permission allows them to take part in providing solutions to adding the validated blocks into the chain.

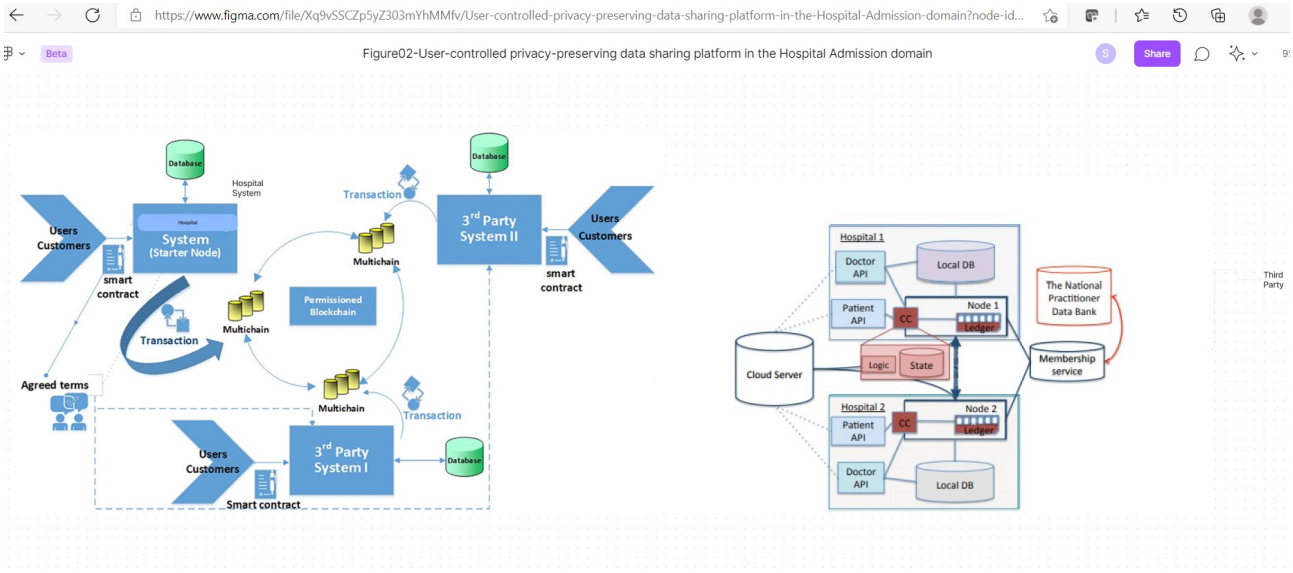


Figure02- User-controlled privacy-preserving data sharing platform in the Hospital Admission domain

https://www.figma.com/file/Xq9vSSCZp5yZ303mYhMMfv/Figure02-User-controlled-privacy-preserving-data-sharing-platform-in-the-Hospital-Admission-domain?node-id=0%3A1

With ***activate* permission**, the node can grant, or revoke *connect, send*, and *receive* permissions to other nodes that wish to join the consortium blockchain network. Finally, with ***admin* permission**, the node can change all the permissions for other nodes in the blockchain network. Besides, through the smart contract, only the selected eligible nodes can access or consume the customer data by subscribing to the corresponding published streams, which we explain in the next section. The data provider which includes the customer and the node offering user data are incentivized as per the negotiation made on the **options between the two parties**. We confined the functions (methods) into our smart contract as per the roles of the participating entities to successively execute different tasks between the data provider and the consumer.

A Big Picture of All mentioned Scenarios in one:

Figure 1, below shows a graphical representation of the combination of the aforementioned scenarios.

Scenario 1: Primary Patient Care. Using blockchain technology for primary patient care can help to address the following problems of the current healthcare systems:

A patient often visits multiple disconnected hospitals. He has to keep the history of all his data and maintain the updates. This leads to the situation when required information may not be available.
Due to the unavailability of the data, patient may have to repeat some tests for laboratory results. This is common when the results are stored in another hospital and can not be immediately accessed.

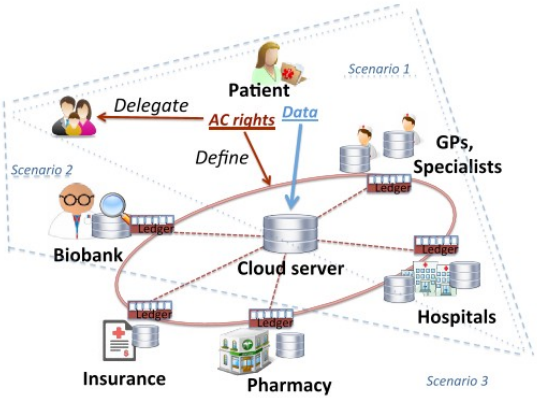


Figure 1: Scenarios of using blockchain in different healthcare settings: Scenario 1: Primary patient care; Scenario 2: Data aggregation for the research purposes; Scenario 3: Connecting different healthcare players for better patient care.

The healthcare data are sensitive and their management is cumbersome. Yet, there is no privacy-preserving system in clinical practice that allows patients to maintain access control policy in an efficient manner.
Sharing data between different healthcare providers may require major effort and could be time consuming. Next, we propose two approaches that can be implemented separately or combined to improve patient care.
Institution-based: The network would be formed by the trusted peers: healthcare institutions or general practitioners (caregivers). The peers will run consensus protocol and maintain a distributed ledger. The patient (or his relatives) will be able to access and manage his data through an application at any node where his information is stored. If a peer is off-line, a patient could access the data through any other online node. The key management process and the access control policy will be encoded in a chaincode, thus, ensuring data security and patient's privacy.
Case specific (serious medical conditions, examination, elderly care): During a patient's stay in a hospital for treatment, rehabilitation, examination, or surgery, a case-specific ledger could be created. The network would connect doctors, nurses, and family to achieve efficiency and transparency of the treatment. This will help to eliminate human-made mistakes, to ensure consensus in case of a debate about certain stage of the treatment.

Scenario 2: Data Aggregation for Research Purposes. It is highly important to ensure that the sources of the data are trusted medical institutions and, therefore, the data are authentic. Using shared distributed ledger will provide traceability and will guarantee patients' privacy as well as the transparency of the data aggregation process. Due to the current lack of appropriate mechanisms, patients are often unwilling to participate in data sharing. Using blockchain technology within a network of researchers, biobanks, and healthcare institutions will facilitate the process of collecting patients' data for research purposes.

Scenario 3: Connecting Different Healthcare Players for Better Patient Care. Connected health is a model for healthcare delivery that aims to maximize healthcare resources and provide opportunities for consumers to engage with caregiver and improve self-management of a health condition23. Sharing the ledger (using the permission-based approach) among entities (such as insurance companies and pharmacies) will facilitate medication and cost management for a patient, especially in case of chronic disease management. Providing pharmacies with accurately updated data about prescriptions will improve the logistics. Access to a common ledger would allow the transparency in the whole process of the treatment, from monitoring if a patient follows correctly the prescribed treatment, to facilitating communication with an insurance company regarding the costs of the treatment and medications.

Implementing the Scenarios. In order to implement the three healthcare scenarios presented above, we must choose between a permissionless and a permissioned blockchain implementations. Below we present the facts that favor a permissioned system implementation.

The anonymity of users and impossibility to verify the identity of account holders (as in case of permissionless blockchain) could cause impersonalization and data misuse.

Patients' healthcare data are of high sensitive nature. Even monitoring communication between a patient and a specific clinician may reveal some sensitive data about the patient, therefore violating the privacy.

Fast response of a system is required as any update of the information about a patient's treatment could be crucial for the patient.

The need to pay for transaction execution, for example, updating permissions for a medical doctor to access a piece of healthcare information or sharing some data for research could limit the usability of the system.

Application in Radiation Oncology:

Sharing Clinical Data(Medical Records) between Healthcare Providers

In this section, we present a prototype design and implementation of a system to support electronic medical record sharing for primary patient care (Scenario 1). More precisely, we focus on patients that are receiving a cancer treatment via ionizing radiation, which is usually performed in the Department of Radiation Oncology of a hospital. First, we describe **a specific use-case scenario and the benefits of the system. Second**, we present the **architecture of the system** and describe the data structure and functionality of the system. Finally, we discuss how privacy, security, and scalability are ensured within the proposed framework.

Use Case Scenario

Cancer is a serious medical condition that may require a long-lasting treatment and a life-time monitoring of a patient. Therefore, it is crucial for the patient to maintain his medical history and to be able to access or share his medical data during the treatment and post-treatment monitoring. Due to the mobility of a patient, the management of the data generated during every patient's visit can be cumbersome especially given the sensitive nature of healthcare data. How to guarantee that the patient's data are complete, stored securely, and can be accessed only according to the patient consent in a fast and convenient manner?

We tackle this problem by applying the blockchain technology to create a prototype of an oncology-specific clinical data sharing system. To present our solution, we can take as an example an oncology information system, {*ARIA25, which is widely used to facilitate oncology-specific comprehensive information and images management. ARIA combines radiation, medical and surgical oncology information and can assist clinicians to manage different kinds of medical data, develop oncology-specific care plans, and monitor radiation dose of patients.*}
Different types of data stored in this system can be structured depending on the clinician's request and exported in PDF format. The documents that contain the data such as history and physical exams, laboratory results, and delivered radiation doses are of the high importance for the clinicians and are most commonly used during the treatment.
Currently, if any of these data have to be transferred from Hospital 1 to Hospital 2, the following procedure takes place. First, the patient (or his official representative) has to sign a consent – a document that specifies the data to be transferred and contains the information about the recipient of the data (Hospital 2). Then, the information has to be printed and mailed to the recipient. Consent management and data transfer in this case can become complicated and inconvenient: the patient may need to contact the caregiver and sign a consent in the hospital from which he is not receiving care anymore. Data transfer can take time, and on receiving the hard copy of the patient data, a clinician will have to introduce them into the system again. Moreover, with this approach, it is very difficult for the patient to maintain any access control of his data and to have a complete view of the data.
By employing blockchain technology, our solution allows to facilitate the consent management and speed up data transfer. We developed a chaincode that allows a patient to easily impose fine-grained access control policy for his data and enables efficient data sharing among clinicians.

System Architecture

Figure 2 presents the architecture of our framework for the oncology-specific data management. The framework consists of the Membership service, Databases for storing healthcare data off-chain, Nodes managing consensus process and APIs for different user's roles.
Currently, we focus on Doctor and Patient, but the roles and their functionality could be extended depending on the scenario.
The main functionality of the **Membership service** is to register users with different roles(currently Doctor and Patient). **The roles define the functionality of the chaincode that is available to the user**. During the registration of a user as a Doctor, it is important to ensure that it is not a potential malicious user, but a qualified medical doctor. To verify this, the National Practitioner Data Bank could be consulted by the membership service. The **membership service is also hosting a certification authority involved in the generation of a key pair for signing** (SKS U ; PKS U) and encryption key pair (SKE U ; PKE U) for every user (U).
Patient (P) also generates a symmetric encryption key (SKAES P) that will be used to encrypt/decrypt the data corresponding to the patient, P . This key will also be used to generate pseudonyms so that only authorized users could verify whether the ledger stores any information about the patient. When a patient (P) needs to share his data with a doctor (D), the patient could share this key, SKAES P , using the encryption public key of the corresponding clinician (PKE D). If the symmetric key SKAES P is compromised, the patient could generate a new one, run a proxy re-encryption algorithm26 on the data stored in the cloud and then share a new key with the clinicians according to the desired access control policy.

The patient's data are stored off-chain in the following Databases. First, a local database management system in the hospital that stores the oncology-related data (for example, ARIA in our use case). Second, a cloud based platform (Varian Cloud) that stores patient's data organized based on the data category (for instance, according to the sensitivity level of the data, or their semantics), and encrypted with corresponding patient key, SKAES P . A registered clinician could assess or upload the data in the cloud repository based on the access control policy defined by the patient and implemented in the chaincode Logic.

A **custom chaincode is deployed on every Node** that acts as a Hyperledger validating peer. Nodes receive all transactions submitted by the users through a role-based APIs. The Node, selected as a leader, organizes transactions in a block and initiates the PBFT consensus protocol. Transactions are executed by all nodes according to the implemented chaincode Logic. **The State stores the information about patients in a key-value pair format. A Key – a Patient Id in the system –** is a pseudonym of the patient that is generated as a hash of the concatenation of the symmetric key SKAES P and a Uniquely Identifiable Information of the patient (UIIP): H SKAES P k UIIP . Combination of SSN (if applicable), date of birth, given names, and a ZIP code of the patient could be used as UIIP . A Value is a patient record stored as a byte array. Next we describe the data structure in detail.

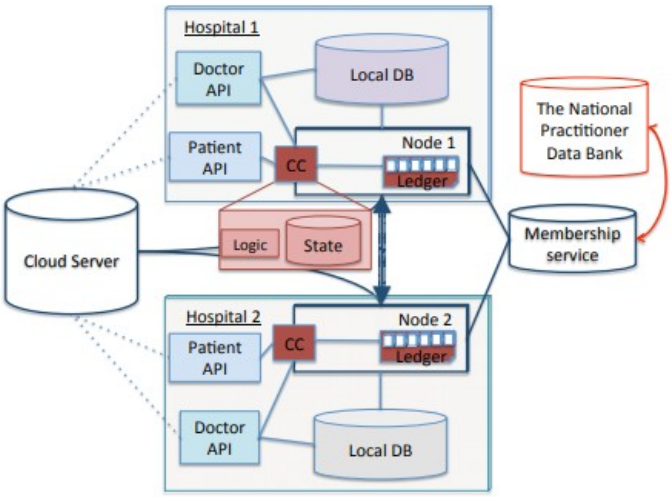


Figure 2: The System Architecture of Blockchain Based Data Management and Sharing for Radiation Oncology

Proposed Patient-Centered Blockchain Model Application as Website Interface with explaining the back end implementation idea:

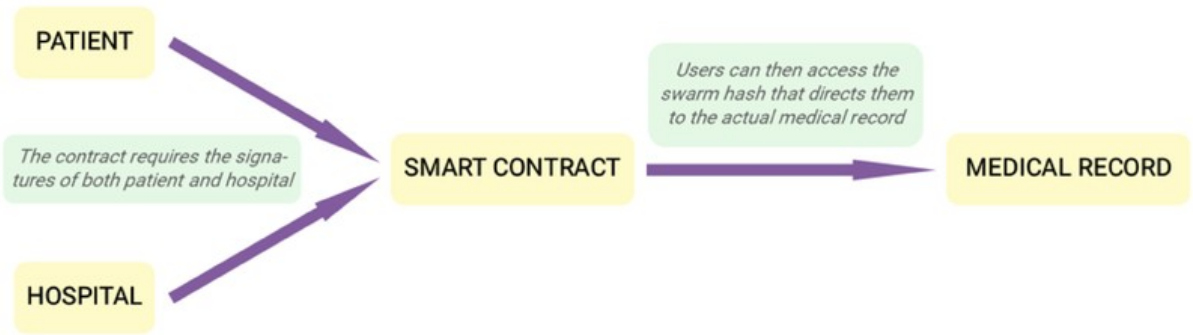


Figure 03

We want to propose a medical data management system using blockchain technology that is secure and allows patients to retain ownership over their own records while allowing hospitals to have easy access (see the above Figure 03). We base our system on the **Ethereum service**, a decentralized platform that allows developers to run applications on a custom-built blockchain (Ethereum Foundation, 2018). Since many times, blockchains do not innately offer sufficient storage, we store the actual medical records on a decentralized cloud storage such as **Ethereum Swarm**, a native base layer service of the Ethereum web3 stack that functions as a distributed storage platform (Swarm, 2018). Each medical record then has a unique swarm hash, which combined with the decryption key, form the root chunk. Only those that know the reference to the root chunk can access the content. Thus, the root chunks are securely stored in smart contracts through the blockchain and are released only under specific conditions.

To solve the problem of data ownership and control, we can employ **multi-signature (multisig) contracts**. Multisig requires multiple users, in this case the **patient and the hospital, to both use their private keys to sign a transaction for authentication.** This way, the patient cannot alter the record without permission of the hospital, but he still has control over who can access his record. A new swarm hash must be generated each time after the data has been accessed (since the old swarm hash is now known), so we add a "last accessed" timestamp. The change in the data will automatically change the **swarm hash**, which can then be secured once again until it receives the required permissions for access. This model not only provides the security and immutability provided by blockchain, but also offers a multisig solution to data ownership and accessibility.

Also, as we were searching we found www.kaleido.io, which is a platform for Document Exchange (kaleido.io). We can leverage Kaleido Document Exchange Blockchain service to store and share data with other users of the system too. We can use Kaleido Document Exchange API to connect our website with its service. Users will be able to easily upload, view and share their medical records with other users (healthcare workers) by clicking a couple of buttons without even having any knowledge of or experience using blockchain technology. The Kaleido document Exchange also provides us with all the event logs on per user basis which can be helpful to track any new upload, deletion, or changes to medical records.

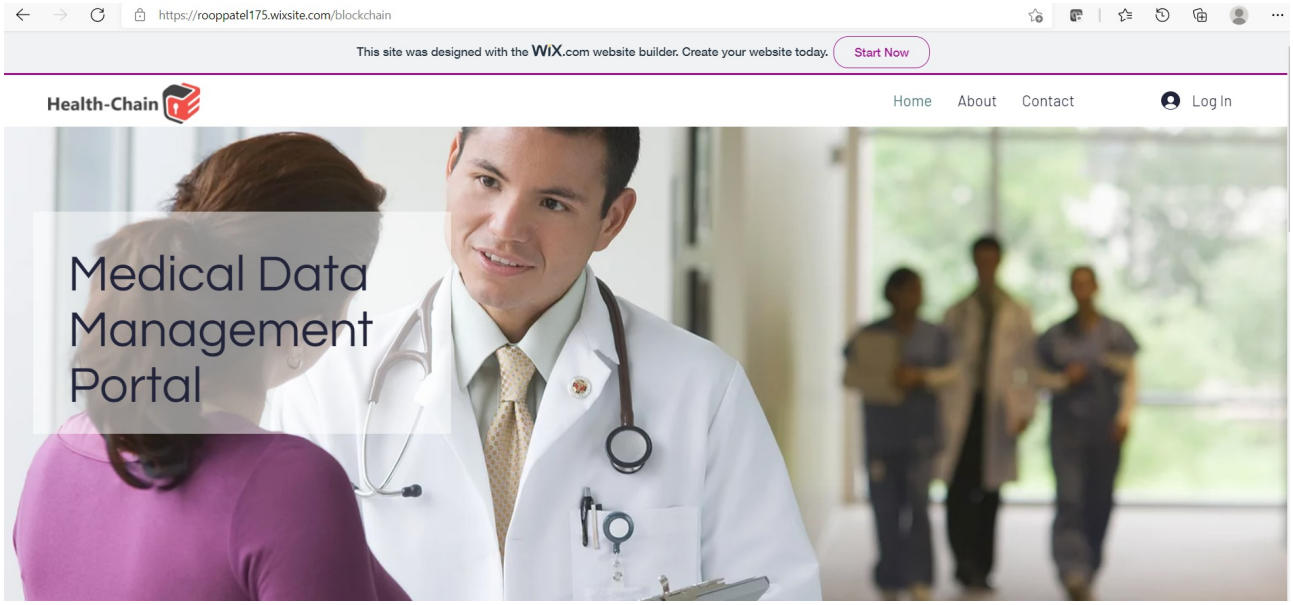
Our Methodolgy:

combining blockchains and off-blockchain repository to create a data sharing and management model focused on security and privacy

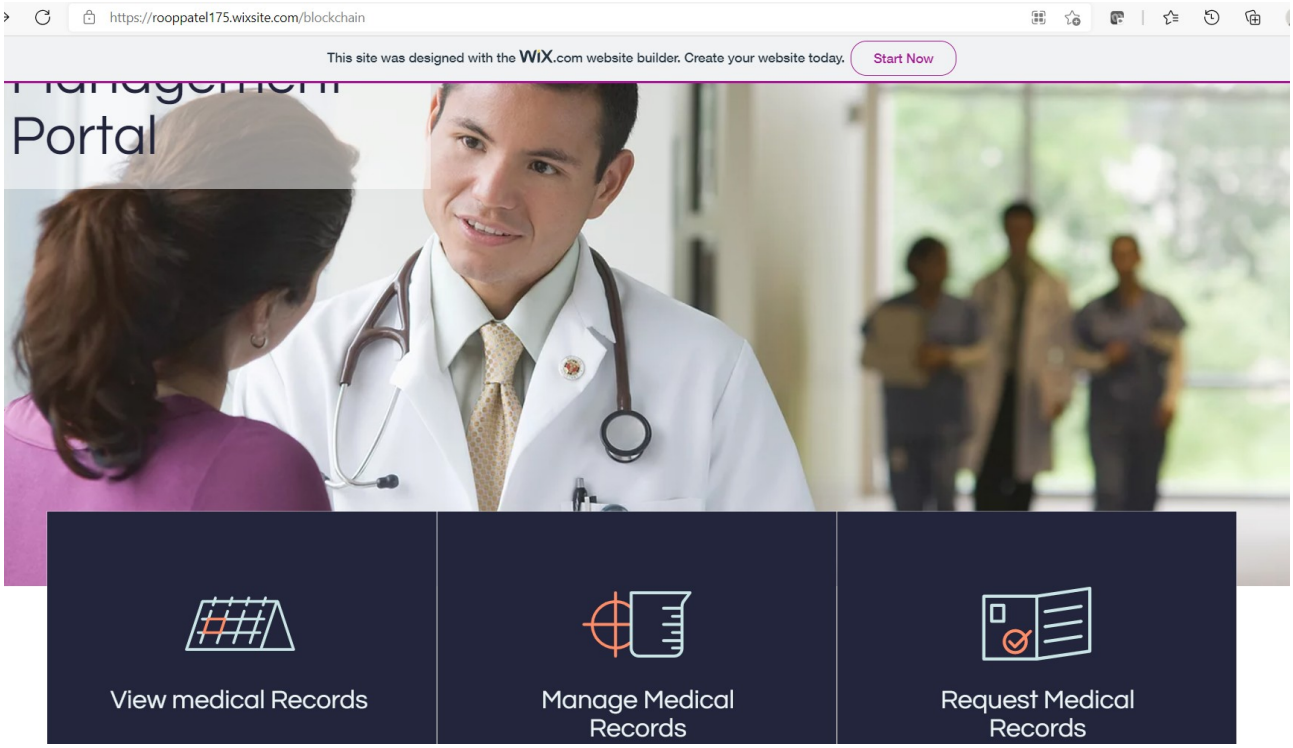
Our Preliminary Interface, Management Portal:

Here is our Membership service (User Interface) for Health care system that aims to Provide better Health care guidance to patients and would help to register users with different roles (currently for us some Hospital related users such as Doctor and Patient):

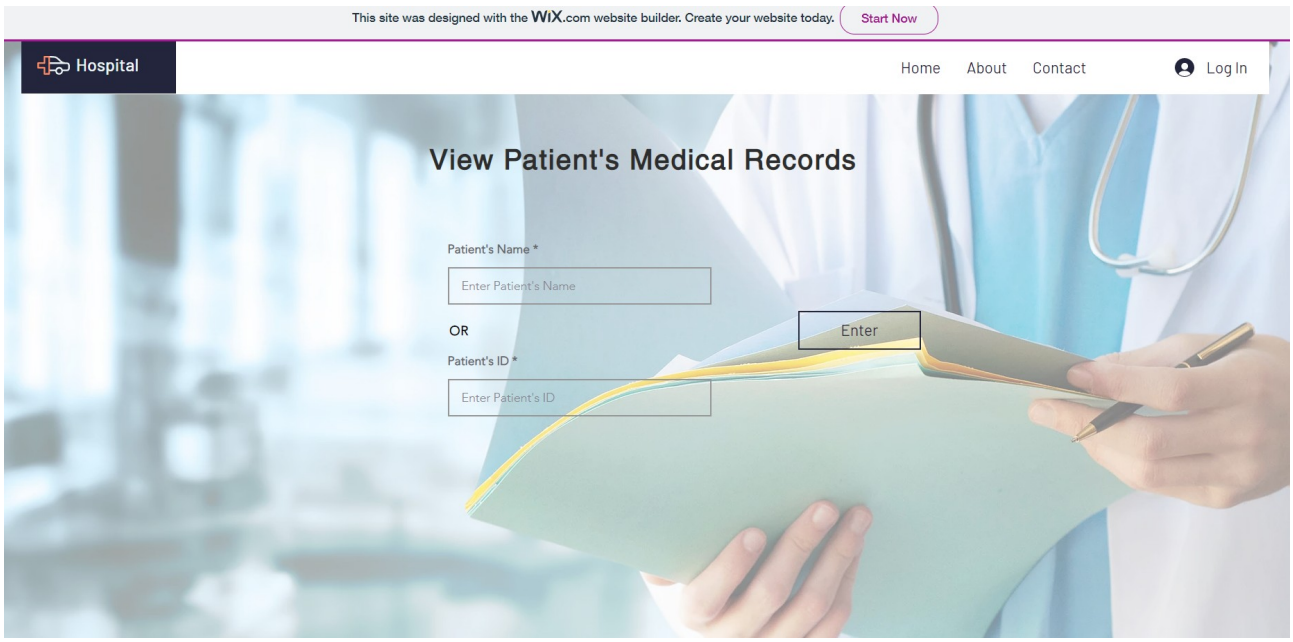
[Home | Blockchain (wixsite.com)](https://rooppatel175.wixsite.com/blockchain)
https://rooppatel175.wixsite.com/blockchain

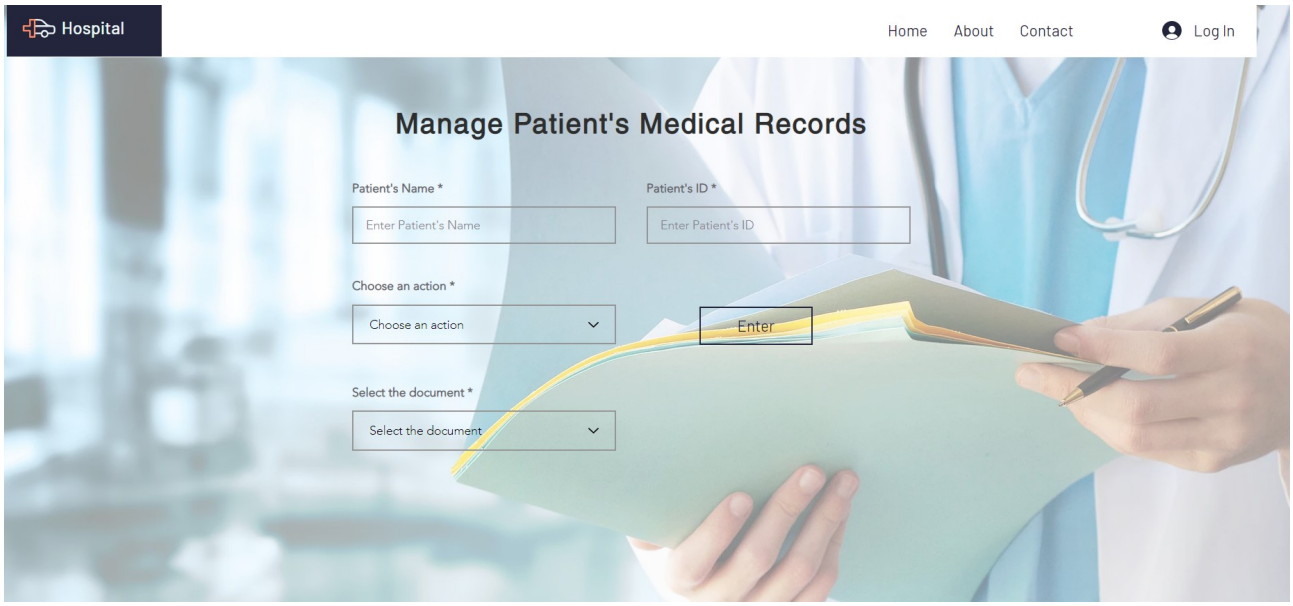


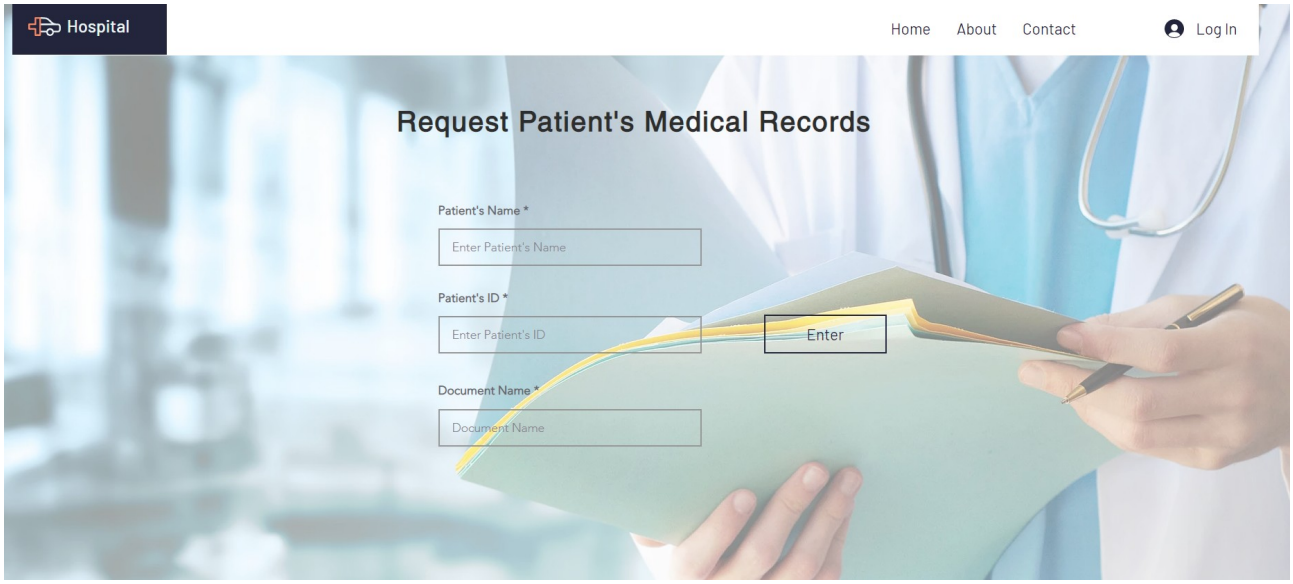
*****



This site was designed with the WIX.com website builder. Create your website today. Start Now







One simple Smart Contract terms and conditions for our proposed designed portal can be defined as follows:

Terms and Conditions between Hospitals and patients to store, share, access and modify patient data.

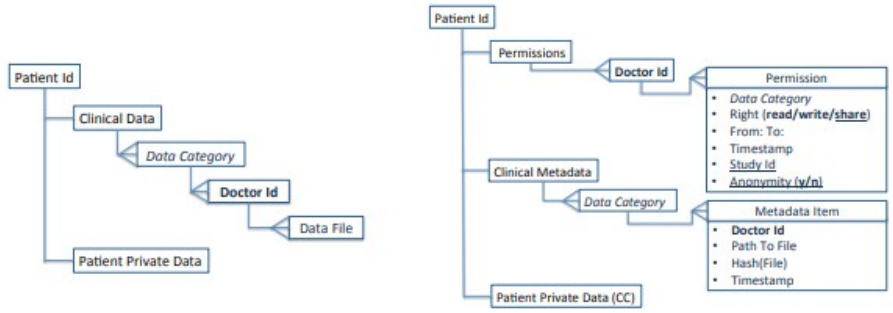
1. Healthcare workers require patients' permission to store, share and access patients' medical data.
2. Healthcare workers will be granted permissions based on their job title.
3. Healthcare workers can access and modify patients' medical data only if they have appropriate permissions.
4. Patients should be notified every time some new data is stored/added to their profile.
5. Patients should be notified every time their medical data is modified or shared (from one healthcare worker to another).
6. If a patient wants to add new data to their profile, they will require endorsement from a healthcare worker with appropriate permissions.
7. Patients does not require any permission to access their medical data.
8. Patients does not have any permission to modify their medical data.

Data Structure and System Functionality

Figure3 below shows how the **patient's data and metadata** are organized:
the patients' data are stored off-chain: locally (in the clinician database) and in the cloud as presented in **Figure 3 (a)** based on their categories. Currently we use three categories in our prototype: History and physical exams, Laboratory results, and Delivered radiation doses.
In the future, we plan to define categories based on both the semantics and the sensitivity level of the data. Data files related to the patient and uploaded by different clinicians are stored within the corresponding category. A patient could optionally store some private data or notes encrypted with the patient public key, PKS P.
Figure 3 (b) presents the structure of the patient's metadata that consists of the following blocks: **Permissions, Clinical Metadata, and Patient Private Data** (optional). **Permissions block** is organized as follows: every Permission corresponds to a Doctor Id, with which a clinician is registered in the system. Every permission specifies the timeframe (From: To:) during which a clinician has a Right to read the patient's data that fall into a specific Data Category, upload them to the cloud repository (write), or share the patient's data within a framework of a specific research study, Study Id.
For the latter the patient could also use Anonymity tag to specify if the data must be anonymized before sharing or could be shared as they are. **Timestamp** makes every permission unique and allows a patient to **update and track access control changes corresponding to the same Doctor Id.**

Clinical Metadata are a block that contains information about all the data files uploaded to the cloud by different clinicians. The Metadata Items are categorized based on the semantics of the corresponding data files. Every Item contains an Id of the clinician that uploaded the data (Doctor Id), a pointer to the file that is stored in the cloud,
Path to File, the Hash of the Data File, Hash(File), to ensure unforgeability of the data stored in the cloud, and the Timestamp of the event when the Data File was uploaded. Similarly to the Patient Private Data stored on the cloud, some private data could be added by a patient to be stored in the State associated with the chaincode (CC). The metadata are stored as a "value" part of a CC State. It can be accessed and modified using the functions that can be invoked on a CC.
To ensure a correct functioning of the developed chaincode we built a network that consists of a Membership service and four Nodes capable of running PBFT consensus protocol. Four is the minimum number of nodes needed to run the **PBFT consensus protocol.** We deployed CC on every node and issued a set of the "invoke" transactions (that trigger creation of a new patient metadata record, adding a permission, and uploading the metadata item), and "query" transactions to access the information from the State.
Currently, a patient is able to create a metadata record on the chaincode, add permissions, and retrieve his up-to-date metadata record, and, thus, his data that are stored on the cloud.
A user registered with a Doctor role is able to upload, access and share for research purposes the data in the cloud based on the permissions specified by the patient.
Verification of the access control rights (currently read, write or share) is done **via Logic of a chaincode** can be written in Go programming language22. For instance, every time a clinician would try to add new data on the cloud repository, a permission corresponding to this clinician has to be retrieved from the patient's metadata record. Then, the validity of the permission with respect to the data category and time frame is controlled.

Similarly, sharing patient data for the research purposes can not be performed by clinician without patient's agreement. This is guaranteed by the chaincode implementation.



(a) The structure of the patient's data stored in the cloud. (b) The structure of the patient's metadata stored on the chaincode.

Figure3: The structure of the patient's data stored in and the structure of the patient's metadata stored on the chaincode

We can develop and deploy a general hospital system admission service and admission web application on one UNIX machine. The web application can be developed in PHP with apache server and MySQL as backend. The machine running MultiChain serves as one of the nodes, that collects

patient's/customers' data with proper validation. The public Ethereum blockchain stores and executes smart contracts. All the registered customers/patients and data consumers have Ethereum addresses, which are used to transfer the ether while sharing the data as per the smart contracts. Users/patients or responsible staff create their profile in the hospital reservation system as the first node (Node 1) in the blockchain by registering their information and simultaneously choosing which of their data can be shared with third parties. The data stored in the repository is converted into an open data JSON format, which is published via stream in the MultiChain. The stream in MultiChain is used for general data storage and retrieval. Third parties, for instance, Research Institutes, and drug stores have different nodes that also contain Multichain in their system. They get the addresses and are given various *permissions* to be in the closed network of the blockchain. One node can have multiple addresses to conduct multiple transactions randomly between them, making the process more anonymous. Public key encryption is an underlying technology of MultiChain, so all the connected nodes generate their own pairs of public addresses and private keys.

MultiChain limits the blockchain access to a group of permitted users by expanding the "handshaking" process when two blockchain nodes connect governed by the following Algorithm A.

1. A node is identified as a public address.
2. A node verifies that the other's address is on its own version of the permitted list.
3. A node sends a challenge message to the other party.
4. A node sends back a signature of the challenge message to prove their ownership of the private key linked to the public address they presented. If either node is not satisfied with the results, it aborts the P2P connection.

Algorithm A. Peer-to-peer (P2P) connection (Greenspan, 2013)

As shown in Figure 003, we first need to initiate a Multichain in the Health care provider system-Hospital Admission system for the first node Imaginary Hospital (Node 1) in the blockchain. This node has an administrator role to grant other nodes associated access privileges. In our case, the permissions for other nodes are set by the first admin node, and they can be made true for all the nodes while setting chain parameters. **The basic chain parameters set in our Multichain are as below:**

#Basic chain parameters

1. chain-protocol = multichain # Chain protocol

2. chain-description = MultiChain model # Chain Description

3. root-stream-name = root # Root stream name

4. root-stream-open = true # Allow anyone to publish in root stream

5. chain-is-testnet = false # Content of the "testnet" field of API responses, for compatibility.

6. target-block-time = 15 # Target time between blocks (transaction confirmation delay), seconds (5–86,400)

7. maximum-block-size = 83,88,608 # Maximum block size in bytes.

The basic chain parameters have been set to limit permissions to any newly added nodes. Similarly, the global permissions in the Multichain are as shown below:

#Global permissions

1. anyone-can-connect = false # private blockchain.

2. anyone-can-send = false # transaction signing is not restricted by address.

3. anyone-can-receive = false # transaction outputs are restricted by address.

4. anyone-can-receive-empty = true #without permission grants, asset transfers and zero na$

5. anyone-can-create = false # selected can create new streams.

6. anyone-can-issue = false # selected can issue new native assets.

7. anyone-can-mine = false # selected can mine blocks (confirm transactions).

8. anyone-can-activate = false # selected can grant or revoke connect, send and receive permissions.

9. anyone-can-admin = false # selected can grant or revoke all permissions.

10. support-miner-precheck = true # Require special metadata output with cached scriptPubKey for input, to support advanced mine$

The multichain daemon is created using the following command with the chain name model:

multichain-util create model

multichaind model–daemon

This creates the MultiChain Core Daemon such that other nodes can connect to this node using the command:

multichaind model@[ip]:[port] (e.g., multichaind model@192.168.204.132:8353)

Here is the link to our figma prototyping for our proposed solution:

https://www.figma.com/file/CcBKCBw9U7Pf6jggA1SmML/Untitled?node-id=0%3A1

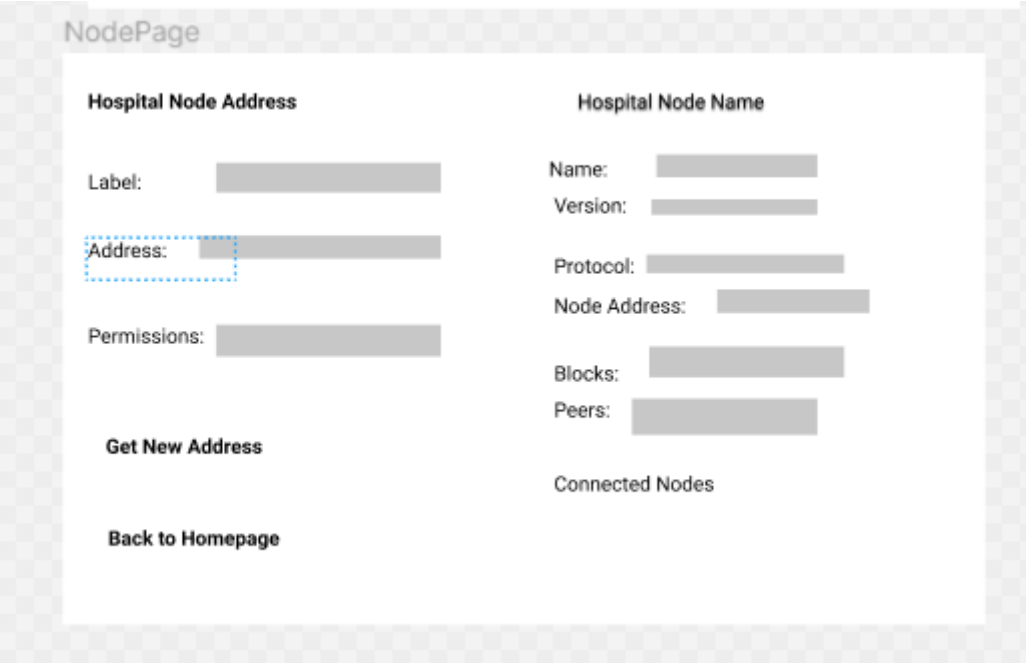

NodePage
Hospital Node Address
Label:
Address:
Permissions:
Get New Address
Back to Homepage
Hospital Node Name
Name:
Version:
Protocol:
Node Address:
Blocks:
Peers:
Connected Nodes

Figure 003-

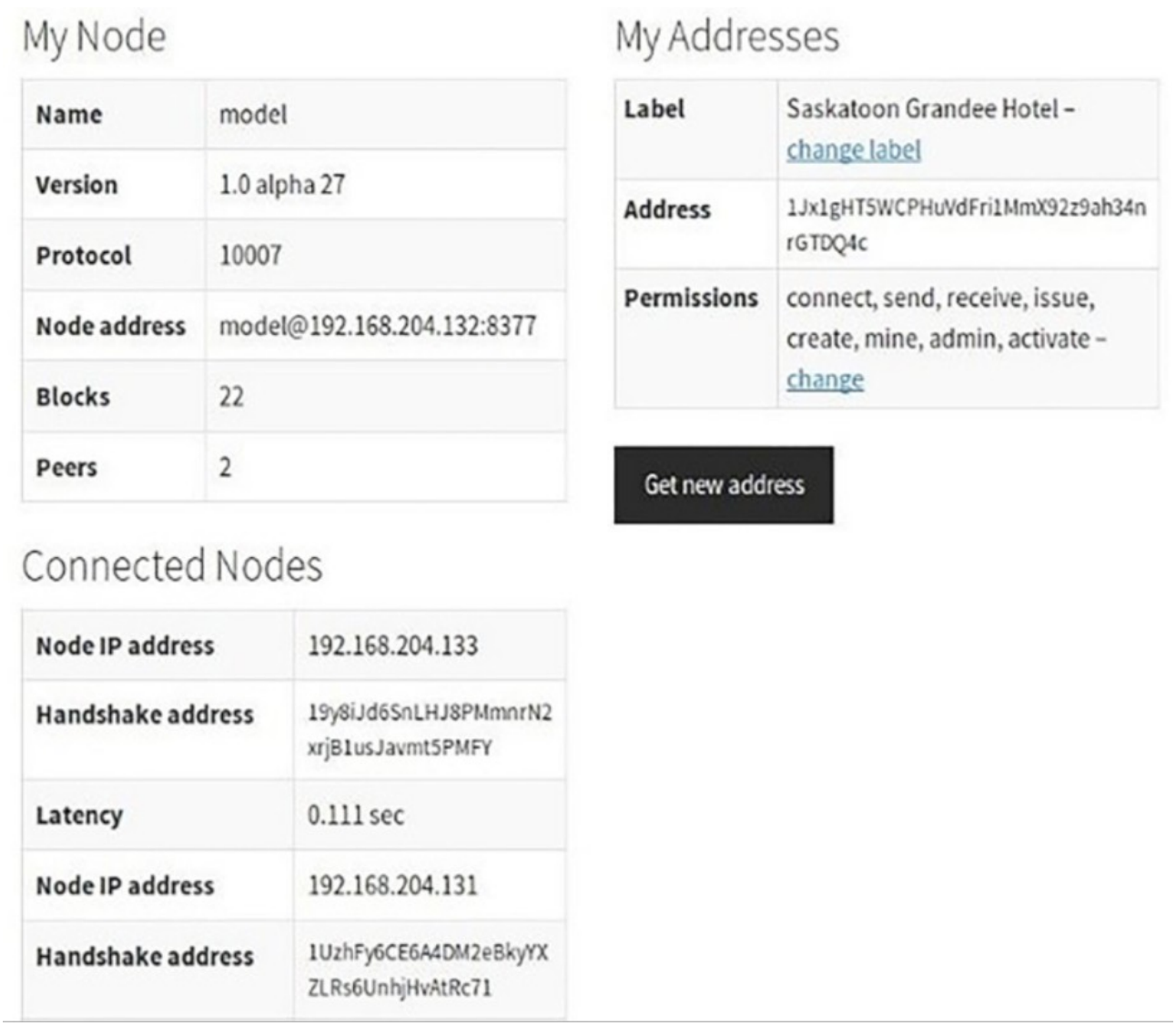


Figure 0003-

We then create two other nodes, Node 2 and Node 3 representing Research Institute, and Drugstore, respectively, as independent firms in the Healthcare domain. The creation of the nodes offered the individual addresses for those new nodes which were acknowledged by the first node to grant a "connection" permission to them into the MultiChain since it is the private blockchain.

Back on the first server, we add connection permissions for another node addresses as:

multichain-cli model grant [address] connect, send, ...

This is the first step in creating the blockchain. While granting the **connection permission**, further other permissions could also be set for other nodes.

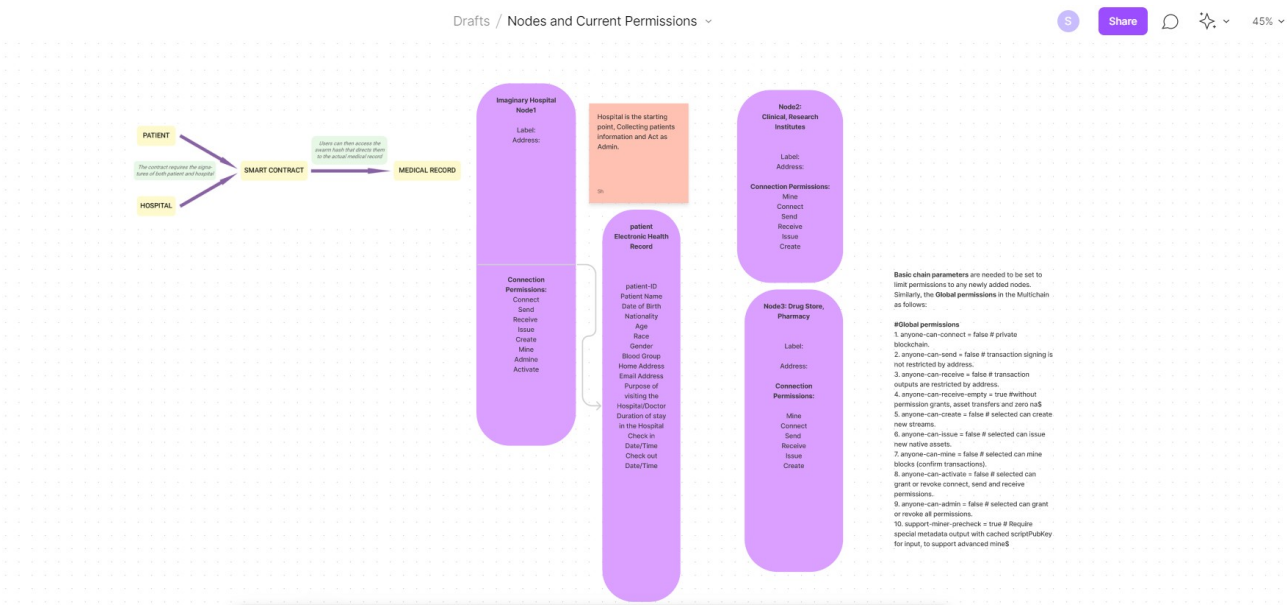


Figure4- Nodes and Current Permissions

https://www.figma.com/file/7Zn2f4jY6U4R4hYOMH63Kf/Nodes-and-Current-Permissions?node-id=0%3A1

Figure 5 shows how to publish the stream by uploading the items containing user profile data into the stream and **share them with other consortium enterprise nodes.**

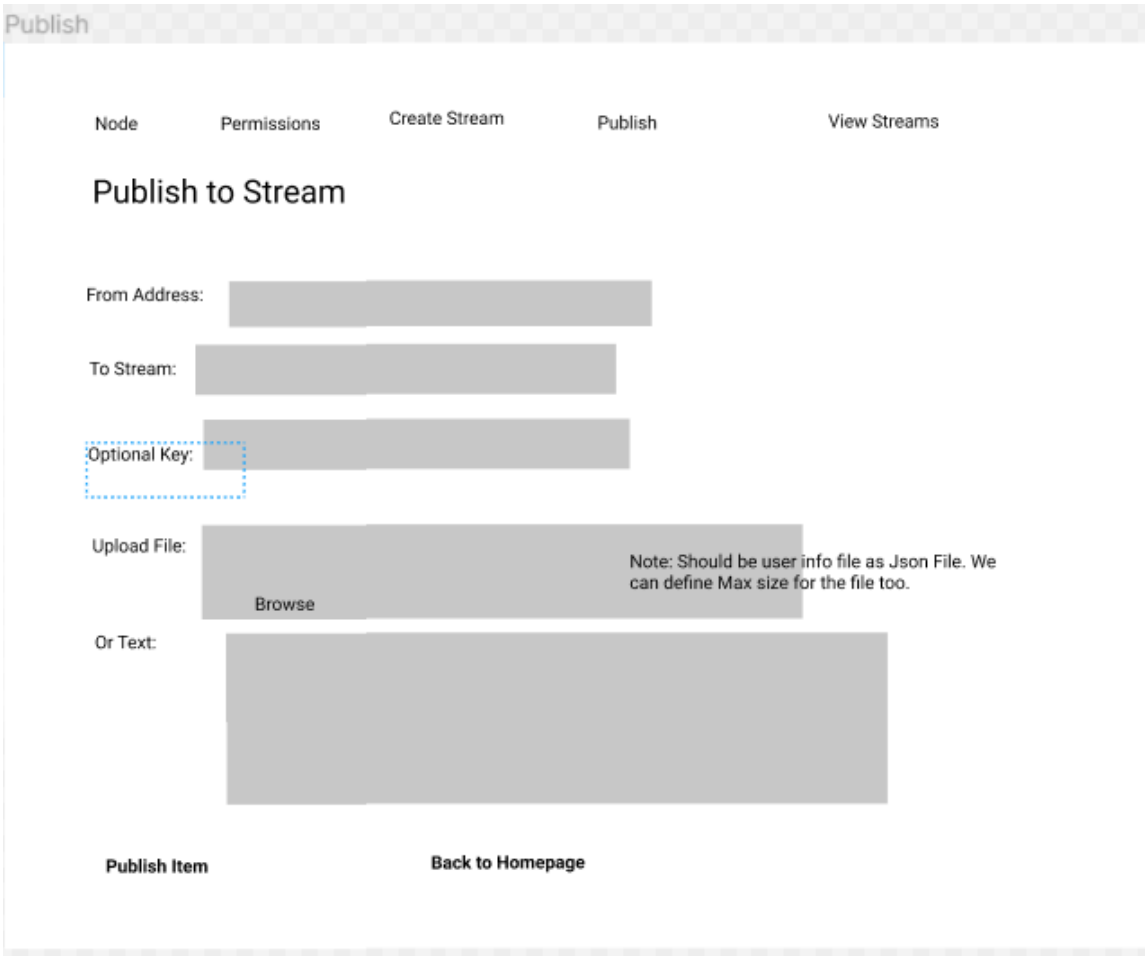


Figure 5-

https://www.figma.com/file/CcBKCBw9U7Pf6jggA1SmML/Untitled?node-id=0%3A1

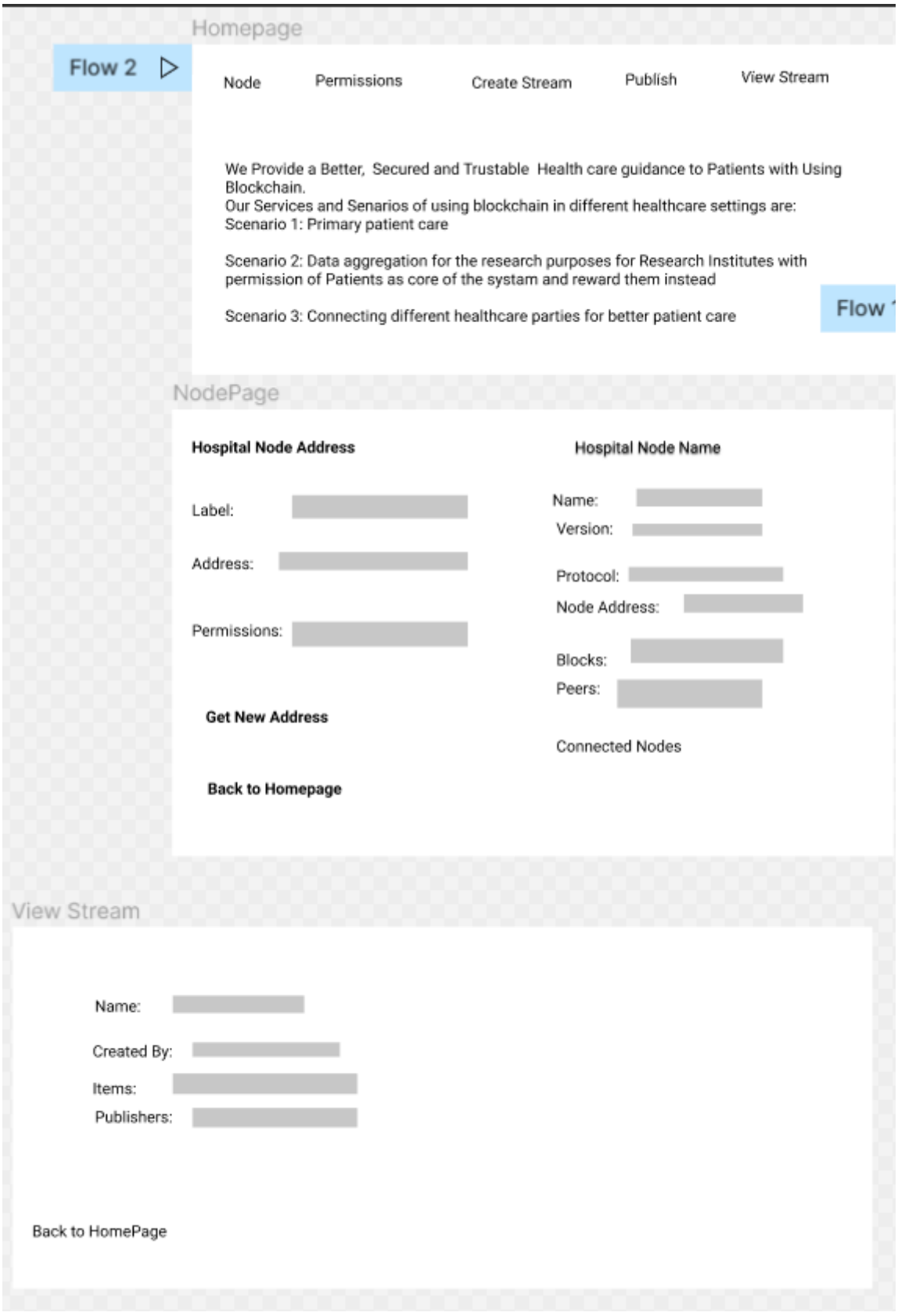


Figure6-

Figure 6- View Stream shows the list of the streams created by the Node 1-Imaginary Hospital. The first node was the Hospital Admission System, which collects the patients as users' data while being accepted in the hospital. The system collects basic user/patient information such as name, the pur-

pose of visit, nationality, duration of stay etc as we mentioned above. This information is particularly useful to other parties/institutes, the drugstore, and clinical institutes because they can provide other services that they might need during or after their visit and stay. The collected useful data from the user profile can be shared among the enterprise consortium as per the predefined agreed terms in the smart contract.

The collected data at Node 1 from the off-chain database is converted into JSON format before being published as items into the stream. There can be any number of streams and the data published in every stream is stored in full or referenced by a hash inside the transactions. Only the nodes with permission can view the contents of the streams.

All other nodes in the network easily can convert and store the received items into their own repositories. Every node in the MultiChain can have access to any stored raw data. To ensure data confidentiality, streams from the Multichain are used with a combination of symmetric and asymmetric cryptography for encrypting the data before being put into the chain. This method utilizes three blockchain streams (more details about this can be found in Greenspan, 2013):

1. Pubkeys stream: To distribute the participants' public keys under the RSA public-key cryptography.

2. Items stream: To publish the large pieces of data, each of which is encrypted using a symmetric AES cryptography scheme.

3. Access stream: To create stream-entry containing a secret password for data, encrypted with the participant's public key to provide data access. Therefore, only a subset of blockchain participants with the password can read the encrypted data.

All the participants in the system with their Ethereum account addresses are in the MultiChain network. The data provider node puts customer data in the local storage. The MultiChain enables to encrypt it using the data consumer node's public key and slice the large dataset into smaller segments. The encryption process starts with the generation of the RSA private-public key pair on the node (server with running bash shell and xxd installed on) that wants to access the customer data, using the openssl and then publishes the public key into the *pubkeys* stream for other data provider nodes to read as:

$ openssl genpkey -algorithm RSA -out [address].pem

$ openssl rsa -pubout -in [address].pem | xxd -p -c 9999

The data provider node uses the openssl with AES cryptography scheme to encrypt the data and produces the corresponding hexadecimal file as:

$ openssl enc -aes-256-cbc -in [datafile] pass:$(openssl rand -base64 48) | xxd -p -c 9999

It then publishes the encrypted file into the *items* stream by creating a transaction with their hashes and commits it into the blockchain. In addition to that, it also retrieves the public key of the data consumer node from the *pubkeys* stream to encrypt the secret password for data as:

$ [passwordshellvariable] | openssl rsautl -encrypt -inkey [publickeyshellvariable].pem -pubin | xxd -p -c 9999

It then publishes the secret password for data into the *access* stream by creating a transaction with their hashes using the data consumer node's address, item's label and commits it into the blockchain. The Institiute data consumer node subscribes to the stream searching for the data and finds the off-chain item with the help of their metadata and hashes. It then places the hash portion in its retrieval queue that queries the data in the P2P network. The node which possesses the data signature responds to the query. At this point, the smart contracts get triggered and with their successful execution, the tokens are transferred from the data consumer's Ethereum address to the customer's/User's account while delivering the requested data (with verified hashes) to the local storage of the consumer node using the same path. The eligible data consumer node uses its private key to retrieve the secret password for the data which was encrypted using its public key with the help of the same openssl and xxd. The application accesses all the MultiChain Community commands using the JSON-RPC API as they are available under open source licenses[11].The smart filters such as stream filters offer callbacks which are also shared with the JSON-RPC API on nodes to examine the validity of the data.

User Incentives for Sharing

We deploy smart contracts on Ethereum that guarantee the user receives the incentive when the user/patient data is consumed by the other parties and institute nodes. All the users/patients (data providers), participating institute nodes (data consumers) have Ethereum addresses which interact with the smart contracts. The users/patients can decide which consumers (applications or drug factories) can access their data. Here, the application sets the terms and conditions that the users/patients agree to allow the enterprise a license to collect and use the contents before using their services and specifies that the user who shared their data retain ultimate ownership to their content, but that institute node also receives the limited perpetual license (and right to sub license) to distribute such contents (for associated smart contracts). The smart contracts give the users/patients full transparency over who accesses their data, when and for what purpose. So, in the beginning, only the selected institutes access the user/patient data by subscribing to the published streams in the MultiChain. The incentive is given in the form of a digital token by transferring ethers to users' ether addresses.

Input: A_{cu}, A_{co}, A_{cc}, deposit, dataPrice, contractState

1. A_{cu}, A_{co}, A_{cc} are the set of all ether addresses of customers, data consumers and contracts, respectively.
2. Grant access to only $a_{cu} \varepsilon A_{cu}$, $a_{co} \varepsilon A_{co}$ who got registered into the system.
3. Change the contract state to Created.
4. a_{cu} deposits e_d.
5. Set data price to ep such that $e_p = ½ e_d$.
6. Contract balance of a_{cc} is $r_b = e_d$, where $a_{cc} \varepsilon A_{cc}$.
7. Allow a_{co} to choose the customer data of its interest.

Algorithm B. Signed Terms And Conditions

Input: A_{cu}, A_{co}, A_{cc}, deposit, dataPrice, contractState

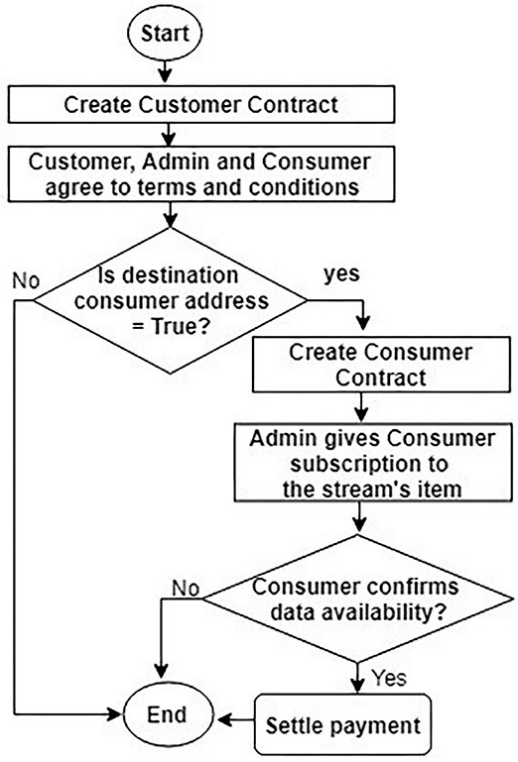
1. A_{cu}, A_{co}, A_{cc} are the set of all ETH addresses of customers, consumers and contracts, respectively.
2. a_{co} decides to consume the customer data acu, pays $2e_p$ such that consumer's deposit $= e_p$.
3. Contract balance of acc is $r_b = e_d + 2 * e_p$.
4. Change the contract state to Locked.
5. Grant a_{co} to access the customer data.
6. a_{co} confirms data availability
7. Change the contract state to Inactive.
8. Transfer deposit e_p from a_{cc} back to a_{co}.
9. Remaining Contract balance of a_{cc} becomes $r_b = e_d + e_p$.
10. Transfer r_b to a_{cu}.

Algorithm C. Confirmed Data Consume

Ropsten blockchain can be used to implement the contracts on an online Remix IDE[12].because the Ropsten is a test network with the same proof of work (PoW) consensus mechanism for the block's validation as in the mainnet of the Ethereum and the Remix is a free IDE to deploy any untrusted codes before going live. In addition to that, anyone can use Etherscan[13] to explore the Ropsten blockchain for searching for any transactions taking place on the blockchain. Also, we use meta-mask[14] to deploy the Injected web3 environment to connect the contracts with the ether account addresses.

Figure 7illustrates a flowchart for the workflow logic with the full process cycle, including the creation of contracts to handling the successful payment for data sharing.

Figure7- Flowchart for workflow logic of smart contracts.



We confine the functions (methods) into our smart contract as per the roles of the participating entities to successively execute different tasks between the data provider(patient and Hospital) and the consumers. The steps to generate the codes are described in *Algorithm B* and *Algorithm C*.

The variables hold the Ethereum addresses, data prices, and contract states. We create a setter function to make its parent or child change the state of the contract if needed, and the compiler automatically generated getter functions for all public variables. We added modifiers in the contracts to support inheritance by restricting the functions with some conditions and created events to keep arguments in the transaction logs that notify clients about what is happening with the contracts.

The contracts following the algorithms B and C can be deployed on the Remix. Thus, as seen from the algorithms, after accessing the user data, the corresponding user is incentivized by the data consumer. **Our approach delivers a usable blockchain-based model for a collection of user data, providing accountability of access, maintaining the complete and updated information with a verifiable record of the provenance, including all accesses/sharing/usages of the data.**

Conclusion and Summary

In summary, we did an experimental study of the new blockchain-based platform for sharing user profile data that allows users to retain control of the sharing and earn rewards. It is based on user-controlled privacy and data-sharing policies encoded in smart contracts. It also naturally supports building up incentives for users to share their profile data, in terms of rewards (micro-payments or credits). In this way, users become owners of their data and can decide how their data is collected and used, as well as shared.

To share user profile data in a distributed fashion, the concept of streams from the MultiChain has been successfully interpreted by taking the case of the Hospital Admission domain. We proposed a hospital admission system as one of the enterprise nodes of Multichain which collects users' profile data/patient's even from different health care providers to give better and easiest service to patients and allows users to receive rewards while sharing their profile data with other parties such as pharmacy factories and clinical and research institutes according to their privacy preferences expressed in smart contracts.

The user data from the repository is converted into an open data format and shared via stream in the blockchain so that other nodes can efficiently process and use the data. The smart contract verifies and executes the agreed terms of use of the data and transfers digital tokens as an incentive to the data owner. The smart contract imposes double deposit collateral to ensure that all participants act honestly.

We have provided a basic use of smart contracts on privacy-preserving data sharing and management. The proposed method have **combined blockchains and off-blockchain repository to create a data sharing and management model focused on security and privacy**.

The node can respond quickly in all our cases with a befitting transaction cost.

Future works:

-Implementing and coding the chain code with Ether Service

-Can be evaluating the usability and usefulness of the approach, and the trust users can have in the system.

-Improving the model by studying users' attitudes to data sharing and the incentives they would find attractive for sharing their data is recommended.