A Privacy-Preserving Framework Using Hyperledger Fabric for EHR Sharing Applications

Preliminary Communication

Vidhi Thakkar
FCAIT Department, GLS University, and Research Scholar, Indus University,
Ahmedabad, Gujarat, India
vidhi@glsica.org

Vrushank Shah
Electronics and Communication Department, Indus University,
Ahmedabad, Gujarat, India
ec.hod@indusuni.ac.in

Abstract – *Electronic Health Records, or EHRs, include private and sensitive information of a patient. The privacy of personal healthcare data can be protected through Hyperledger Fabric, a permissioned blockchain framework. A few Hyperledger Fabric-integrated EHR solutions have emerged in recent years. However, none of them implements the privacy-preserving techniques of Hyperledger Fabric to make transactions anonymous or preserve the transaction data privacy during the consensus. Our proposed architecture is built on Hyperledger Fabric and its privacy-preserving mechanisms, such as Identity Mixer, Private Data Collections, Channels and Transient Fields to securely store and transfer patient-sensitive data while providing anonymity and unlinkability of transactions.*

Keywords: *EHR, Permissioned blockchain, Hyperledger Fabric, Privacy-preserving Mechanisms, User anonymity*

1. INTRODUCTION

In recent years, we have witnessed a significant rise in the adoption of technology in several sectors, including healthcare. While technology has improved the productivity of healthcare organizations, technological applications such as Electronic Health Records have also raised significant privacy concerns [1, 2]. A distributed ledger technology called blockchain has emerged in recent years as a solution to address data privacy issues. Blockchain is an append-only immutable ledger made with blocks where each block points to the subsequent block using an address. If the data in a block is altered, the block's hash will change, rendering the entire blockchain invalid. Since 2016, Blockchain-based solutions have grown more popular and have found several applications across industries [3]. Integrating blockchain technology with EHR systems can overcome the challenges of centralization, data security and privacy, data integrity, interoperability, transparency, and accountability [4]. There are various permissionless and permissioned blockchain frameworks, such as Ethereum, Enterprise Ethereum, Ripple, Hyperledger Fabric, Hyperledger Iroha, Indy, Corda, Tendermint, chain, and Quorum [5] [6]. From these, Ethereum and Hyperledger Fabric blockchain frameworks are being used for healthcare organization use cases [1]. In terms of average transaction latency, throughput, scalability, and privacy, Hyperledger Fabric beats Ethereum. Hyperledger Fabric's modularity and channel design makes it more suited to businesses and use cases where trust, transparency and security are critical.

In the last few years, several Hyperledger Fabric-integrated EHR solutions have been published [7-13]. However, none of them preserves transaction privacy and user anonymity during transaction consensus. In EHR sharing systems, privacy is more crucial. User privacy and data privacy are the two notions of privacy with respect to blockchain integrated EHR solutions.

In the existing blockchain based EHR sharing use cases, data privacy concern is partially solved by using off-chain storages such as IPFS to store actual data and only metadata (hashed values and encrypted information) is stored on blockchain ledger. However, input parameters and state data of a transactions are still visible to all the network members. User privacy is concerned with user/transaction anonymity and unlinkability. Anonymity and unlinkability of the user are crucial for maintaining user privacy since they guarantee that no one can identify the user, the endorser, or the validator. In the existing Fabric-based EHR system, the user provides their identity and signature during the transaction endorsement so that the peers can later validate it. However, this disclosure makes EHR transactions linkable. All the input parameters are visible to all the peers of the channel though encrypted. Apart from that endorsement on transaction response can reveal the identity and the organization involved in the endorsement process, leading to vulnerabilities like DDoS attacks on the transactor nodes and endorsers. To overcome such challenges, this study proposes a permissioned EHR with Hyperledger Fabric's privacy preservation techniques and a few crucial developments required to protect the privacy of nodes on the channel.

The remainder of the paper is organized as follows: Section 2 discusses some significant contributions that have been made in this field. The related literature is discussed in Section 3. Section 4 provides preliminaries regarding the privacy-preserving features of Hyperledger Fabric. Section 5 presents Hyperledger Fabric Enabled Blockchain Architecture for EHR sharing systems. Section 6 provides information regarding the tools to implement the proposed blockchain-based framework. Section 7 shows a comparison of the proposed methods against other Fabric based EHR studies. Finally, Section 8 summarizes our conclusions.

2. CONTRIBUTIONS

Hyperledger Fabric is a permissioned blockchain with various privacy features for users. It differs from other permissioned frameworks because it restricts chain code execution to a group of peers, known as endorsing peers. This subset of peers on a channel executes chaincode according to the endorsement policy. Later, the validator examines the transaction during the validation phase to verify if the desired endorser signature matches the endorsement policy. EHR sharing use case requires confidentiality while executing transactions between a subset of organizations on the network. For instance, only a specific privileged subset of organizations should be aware of the transaction existence and be permitted access to its sensitive data assets. The disclosure of the endorser's signature and endorsement policy entails several security and privacy risks. To address this concern, this paper proposes privacy preserving permissioned EHR system to ensure transaction anonymity and unlinkability.

The key contributions are summarized as follows:

- The use of the Hyperledger Fabric to improve security, transparency, privacy, and interoperability.
- The use of Arweave peer-to-peer databases to save and retrieve data permanently.
- The use of Hyperledger Fabric's Java SDK and privacy-preserving features of Fabric such as Identity Mixer, private data collections, and transient field to achieve user unlinkability and transaction data privacy during consensus.

3. RELATED WORK

This section explores the latest research on Hyperledger Fabric integrated EHR sharing systems.

Huang et al. [7] suggested Medbloc, a secure EHR sharing platform built on Hyperledger Fabric. The system was created to aid the healthcare sector of New Zealand by using the Hyperledger Fabric version 1.1 and the PBFT (Practical Byzantine Fault Tolerant) consensus mechanism. Smart-contract-based access control systems have been implemented to store encrypted medical records on the blockchain.

Mahore et al. [8] suggested a permissioned blockchain paradigm to effectively maintain medical records while achieving privacy, security, scalability, and availability. By storing sensitive patient data off-chain and only retaining the hash of the data in a blockchain record, this approach solves the scalability problems that users often encounter while using the blockchain. Moreover, they employed a proxy re-encryption approach for transmitting encrypted data to the provider. They created their solution using the PBFT consensus algorithm, the solo ordering service, and Hyperledger Fabric version V1.4. They have also statistically examined their strategy using the Hyperledger Caliper to determine how effectively it works.

Later in 2020, more studies were published to create scalable and secure EHR-sharing systems using the Hyperledger Fabric. Dubovitskaya et al. [9] presented an ACTION-EHR for cancer patients. Researchers and healthcare professionals have exchanged EMR data using the prototype constructed through Hyperledger Fabric architecture. The Patient's medical information is encrypted with public-key cryptography and kept on servers off the blockchain. The system guarantees accurate access control and data confidentiality. The ACTION-EHR blockchain network is built using the Membership Service, orderer, cloud server, Hyperledger Fabric v1.4 SDK, and CouchDB for on-chain information management and permissions.

Tith et al. [10] used consortium blockchain and Hyperledger Fabric to create a distributed solution for merging existing EHRs. Encrypted data is transmitted from the patient to the physician using a centrally located server and a proxy re-encryption method. The eID has been salt hashed to prevent transactions from being tracked on the ledger. With Hyperledger Fabric and Hyperledger Composer, their approach has achieved scalability, transparency, traceability, availability, and dependability.

Chenthara et al. [11] published a framework for protecting electronic health records using permissioned blockchain technology. Only encrypted hashes of the records are kept on the blockchain and huge EHRs have been stored on distributed IPFS. Better assurances of data quality, scalability, privacy, and interoperability are achieved in their new paradigm. The components of Fabric's MSP, CA, PBFT consensus, CouchDB, and chaincode are used to create the system. Through this approach, they were successful in increasing scalability while assuring an elevated level of security and privacy.

Stamatellis et al. [12] developed the Hyperledger Fabric-based EHR management solution PREHEALTH to protect patient privacy. The solution discusses Identity Mixer to ensure transaction anonymity and unlikability. The approach is compliant with GDPR and emphasizes Hyperledger Fabric's privacy-preserving features. However, there has been no discussion regarding the implementation of Identity Mixer in the configuration data for Hyperledger Fabric.

Mani et al. [13] introduced PCHDM, a patient-centred healthcare data management solution. Their strategy uses off-chain IPFS to store heavy files and stores hashes of the encrypted healthcare data on-chain to ensure that the healthcare records are stored securely. Byzantine Fault Tolerance (BFT) consensus for the implementation of smart-contracts inside secure containers. Thus, prominent security, privacy and scalability are possible. Hyperledger Caliper benchmarks for transaction latency, throughput, and resource utilization are used to assess performance.

The above-mentioned Fabric-integrated EHR studies focus on providing a solution for patients to have complete control over their data. Their work utilizes Fabric components such as chain codes to define access policies, plug-and-consensus PBFT, MSP (Member Service Provider), CA (Certificate Authority), CouchDB database, and off-chain storage technologies to store patient's heavy data. Below in Table 1, we have analyzed their work using parameters such as Hyperledger Fabric version, Storage, Consensus Mechanism, Ordering Service, Immutability, Performance, Data Privacy during consensus, and User Privacy. These studies address interoperability, immutability, scalability, performance, and data protection issues of permissionless blockchain-based EHR systems. However, none of these studies has considered or implemented Hyperledger Fabric's security and privacy-preserving mechanisms to further protect the health sector's sensitive data inside the permissioned network. Only Stamatellis et al. [12] have proposed a secure EHR management solution with Hyperledger Fabric's privacy-preserving features, but they have also not demonstrated any practical implementations of their work.

Table 1. Existing Fabric integrated EHR sharing studies

Reference	Year	HF Version	Storage	Consensus/ Ordering Service	P1	P2	P3
[7]	2019	V1.1	onchain	PBFT	N	N	N
[8]	2019	V1.4	cloud	-	N	N	N
[9]	2020	V1.4	cloud	PBFT	N	N	N
[10]	2020	-	cloud	BFT	N	N	N
[11]	2020	V1.3	IPFS	PBFT	N	N	N
[12]	2020	-	-	-	N	N	N
[13]	2021	-	OrbitDB+IPFS	BFT	N	N	N

*P1= Data Privacy & GDPR, *P2= Transaction data Privacy during consensus, *P3= User privacy

4. PRELIMINARIES

4.1. HYPERLEDGER FABRIC

Hyperledger Fabric is an open-source enterprise blockchain platform developed by "The Hyperledger Foundation". Fabric is a permissioned network with a modular design, higher scalability, improved confidentiality and better performance. The Fabric enables the user to perform private transactions (private data collections and transient fields) and the development of parallel and independent lightweight channels to share data with only a portion of the nodes of the blockchain system. Fabric supports three CFT consensus algorithms: Raft, Kafka (deprecated), and Solo (deprecated). The Fabric components involve channels, peers, assets, identity, membership, transactions, chaincode, ledgers, and consensus procedures as part of the network. Participants sign up through a reputable Membership Service Provider (MSP) that keeps a list of permissioned identities made by Certificate Authorities (CA).

Hyperledger Fabric Components

MSP and CA: CA assigns certificates and works as public/private key issuers. MSP is responsible for issuing node credentials for authentication and authorization. All participating entities in Fabric must use digital certificates and cryptographic processes to identify and authenticate themselves through the MSP service.

Chaincode: Chaincode is a self-executing program. They are installed on the blockchain network and update the current state of the ledger when executed. In Fabric, the chaincode is installed on the endorser nodes only. Chaincode will be executed when certain conditions are met, and the outcomes of the transaction execution are sent to the blockchain network and subsequently added to all peers' ledgers. Fabric's chaincode logic can be written using Java, Golang, Node JS or Python.

Channels: Organizations with similar goals can be grouped into a single channel. Channel enables private and secure communication between various network parties.

Peers: The peer may be a Leader Peer (distributes block to other peers), an Anchor peer (other peers can discover and communicate with it), or an Endorser, or an Orderer, or a Validator peer.

Endorsement Policy: For a smart contract to be executed, Fabric has a configurable endorsement policy that determines the minimum number of peers that must endorse the proposal.

State Database: Endorsing Peers store output data related to endorsements in local state databases like CouchDB and LevelDB. LevelDB is a key-store database used as a state database and cannot handle complex queries. CouchDB is a document store database. Here the search is performed on keys and data values rather than only keys.

Ordering Service: The ordering service has numerous OSNs (Ordering Service Nodes). These nodes establish a global order of transactions and produce blocks for the broadcast to peers. Fabric supports Raft, Solo, and Kafka are the three ordering services. Amongst Solo and Kafka are deprecated in the latest version of Fabric, only Raft is working.

Privacy-preservation Mechanisms: Privacy protection features of Fabric include the following four aspects, namely the channels, the private data collections, the zero-knowledge proofs (ZKP) and the membership service provider. The channels keep transactions hidden from the larger network whereas PDC (Private Data Collection) helps maintain data privacy between specific groups of organizations on the channel. ZKP is for anonymous client authentication with Identity Mixer support.

4.2. CHANNEL

A Channel is a secret "subnet" which allows two or more network members to perform private and confidential transactions. Channels are used to hold various information such as transactions, chaincodes, policies, membership, and configuration. This information is utilized to maintain the network's security and privacy. Every channel has a separate distributed ledger to keep digital signatures, timestamps, i/o transaction data, and transaction IDs. But disclosure of channel data to all the network members can disclose sensitive information about a transaction. To overcome this, multiple channels can be created at once. However, the creation of a network with many channels brings down decentralization, limits interoperability, and raises network overheads.

4.3. PRIVATE DATA COLLECTIONS

Using Private Data Collection (PDC) feature, a subset of organizations over the same channel can transact with one another without worrying that another business will see their confidential information. In Fabric v2.0, an implicit collection for each organization is introduced; therefore, application chain code can be used even if the collection is not defined explicitly. Peers employ local state databases like CouchDB, MongoDB, and LevelDB to store data. Figure 1 depicts the two parts of a ledger as well as the location of the private data. The one is private data, which remains confidential and passed from peer to peer using the gossip protocol to other organizations according to the specified policy. The ordering service cannot see the private data. The other element of PDC is data hash. The Hash is endorsed, ordered, and written to each peer on the channel and serves as transaction proof and is auditable.

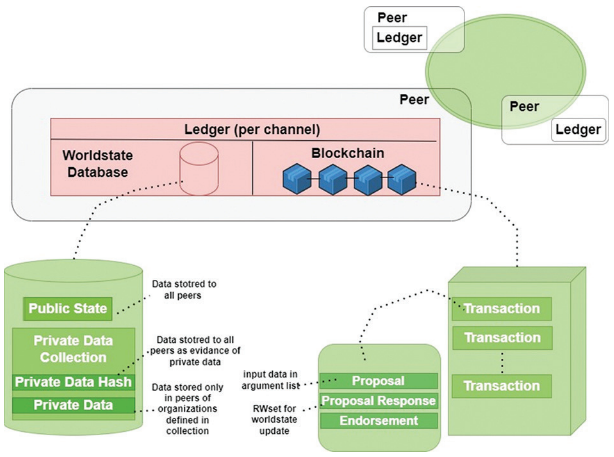

Fig 1. Ledger and location of private data

We have implemented the PDC feature for our permissioned EHR sharing application. Below is an example of creating the data definition in the chaincode. The medical data transfer sample divides the private data into three definitions and is defined in a JSON file called collection definition. The first collection is for both organizations, the second collection is only for Org1, and the third collection is only for Org2. We can check that the actual data is in the \$\$p database, whereas the hash of the actual data is in the \$\$h database.

```
Org1 and Org2 collection:
type PatientInfo struct {
PatinetID       string    `json:"patientID"`
Name string     `json:"patientname"`
Age     int      `json:"age"`
OrgName         string    `json:"Hostname"`
}
Org1 Private Details:
type Org1PrivateDetails struct {
Charges int `json:"fees"`\\fees of a doc
No_of_Patient_in_a_day int `json:"no_of_patient"`
}
Org2PrivateDetails
type Org2PrivateDetails struct {
Salary int `json:"salary"` // salary of doctor
AppraisedValue int `json:"appraisedValue"`
}
```

Discussion on PDC

DCs can be a useful way to store private and sensitive data for businesses, as the data stored on PDCs can be discarded. Unlike data on the blockchain, PDCs are not immutable, which makes them an interesting option in terms of GDPR compliance. To solve GDPR compliance with PDC blockToLive parameter can be set in config.json file.

4.4. TRANSIENT FIELD

Private Data is concerned with preserving the data privacy inside a defined subset of organizations, whereas Transient Field is an input technique for Private Data. Both are two distinct concepts from a technical standpoint, but they can be used together to achieve a certain level of security in Hyperledger Fabric Applications. The Transient Field [15] data keys will only be accessible by chain code installed on the docker container (Figure 2). The encryption/decryption key can be passed using a transient field bypassing the endorser. This key can be processed by chain code to encrypt/decrypt data and later saved to the PDC of the peer for better privacy. Transient data is passed as binary data, and base64 is encoded in the terminal. PDC and Transient Filed features can be used to achieve data confidentiality inside a Fabric network. However, using PDC for businesses has problems related to data privacy because all peers (inside and outside the subgroup) will preserve a record of private data hash as proof of data existence. There will not be any proof of transient information on the ledger.

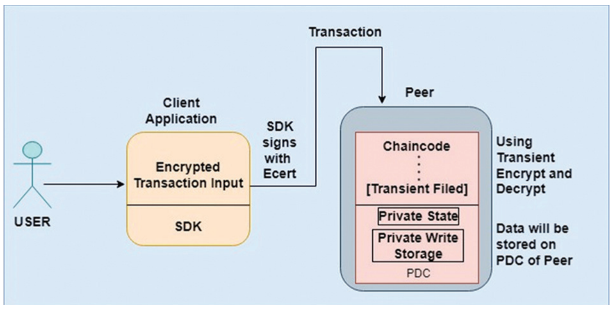

Fig 2. Example of Transient Field

4.5. IDEMIX

Idemix cryptographic protocol suite [16] uses ZKP to provide anonymity and unlinkability. It provides support for audibility and revocation. In Hyperledger Fabric, the client or user provides their identity and signature during transaction endorsement that is to be validated later by the peers. However, disclosing the identity of a user or a client to the endorsing peers for EHR transactions may compromise patient privacy and make transactions linkable. To deal with these concerns, Identity Mixer can be implemented on top of the Hyperledger Fabric. Using this Identity Mixer MSP, a user or the client can sign the transaction and remain anonymous, generating an unlinkable signature. We have implemented the Idemix concept in Hyperledger Fabric Java SDK since the support is only available for Fabric Gateway SDK for Java and Fabric Gateway Client API for Java.

Features:

Anonymity: The ability to transact without revealing the identity of the transactor.

Unlinkability: The ability of a single identity to send multiple transactions without revealing identity.

Actors:

Idemix flows involve the user, issuer and verifier as three different players. A set of user attributes are issued as a digital certificate by an issuer, also referred to as a "credential." A "zero-knowledge proof" of ownership of the credential is produced by the user, who also selectively discloses only the attributes they want to make public. With the help of the issuer's public key, the verifier can now verify the Zero Knowledge Proof provided by the user. Using the issuer's public key, the verifier verifies the proof provided by the user. The proof does not expose any further information to the verifier, issuer, or anybody else because it is zero knowledge.

Setting Up Idemix for Existing Fabric Network:

1. For Development Environments, the “idemixgen“ tool from Hyperledger Fabric Releases can be used to generate Issuer Public Key as well as Issuer Revocation Public Key.

   Steps:

   1) Generating CA Key Pair

   2) Generating Default Signer

2. For Production Environments, use “Fabric CA” as an Idemix Issuer.

   Steps:

   1) Generating CA Key Pair

   2) Generating Default Signer

Verifier:

Idemix MSP must be created using Issuers “IssuerPublicKey” & “IssuerRevocationPublicKey”. Later with the help of a “configtxgen“tool, a new Idemix Organization MSP definition in the form of JSON must be created by specifying the MSPID of the Idemix organization. Now, the Application channel configuration needs to be updated with the new Idemix organization definition. The above steps need to be done by any admin of the existing organizations in a channel. Also, a majority of the existing organizations in a channel have to approve adding a new MSP to the channel.

User:

Through Java SDK, Idemix credentials can be obtained for a user. By using this credential, users can sign and submit transactions.

Internal flow of generating Idemix credential for a user:

1. Get Certificate Authority Public Key and Revocation Authority Public Key. These are needed to request a new credential for a User. They can get from the "/cainfo"APIEndpoint of a CA.
2. This step is consisting of two steps:
   a) Registration was the same as the X509 registration.
   b) For Enrollment,
      - Get a nonce, which is required to construct a Credential Request.
      - Get an Idemix credential & CRI (Credential Revocation Information).

The process of submitting a transaction using IDEMIX

The process of submitting a transaction using Idemix Identity from CLI is shown below.

1. Generating an Idemix credential for the user

» FABRIC_PATH=~/Desktop/test

» cd $FABRIC_PATH/idemix-demo-fabric-samples/test-network/organizations/peerOrganizations/org1.example.com

» idemixgen signerconfig --ca-input=idemixmsp --admin --enrollmentId=appUser --org-unit=org1

2. Setting environmental variables for Idemix Identity

» export CORE_PEER_ADDRESS=localhost:7051

» exportCORE_PEER_TLS_ROOTCERT_FILE=$FABRIC_PATH/idemix-demo-fabric-samples/test-network/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt

» export CORE_PEER_TLS_ENABLED=true

» export CORE_PEER_LOCALMSPTYPE=idemix

» exportCORE_PEER_LOCALMSPID=Org1IdemixMSP

» export CORE_PEER_MSPCONFIGPATH=$FABRIC_PATH/idemix-demo-fabric-samples/test-network/organizations/peerOrganizations/org1.example.com/idemix-config

3. Invoking the Chaincode

» cd $FABRIC_PATH/idemix-demo-fabric-samples/test-network

» export FABRIC_CFG_PATH=$PWD/./config

» peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls –cafile

» ${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C mychannel -n basic --peerAddresses localhost:7051 –tlsRootCertFiles

» ${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses localhost:9051 –tlsRootCertFiles

» ${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"function":"TransferAsset","Args":["asset1","Vidhi"]}'

4. Query the Chaicode using Idemix Identity

» peer chaincode query -C mychannel -n basic -c '{"Args": ["GetAllAssets"]}' | jq.

Comparison with X.509

Compared to the conventional X.509, Identity Mixer signatures give advanced privacy protection. With this IDEMIX, the client can hide attributes and reveal only selected attributes. The original Enrolment id will be hidden from the orderer and other peers. We can see in Figure 3 there is an unlinkability of signatures generated as only selected attributes are disclosed.

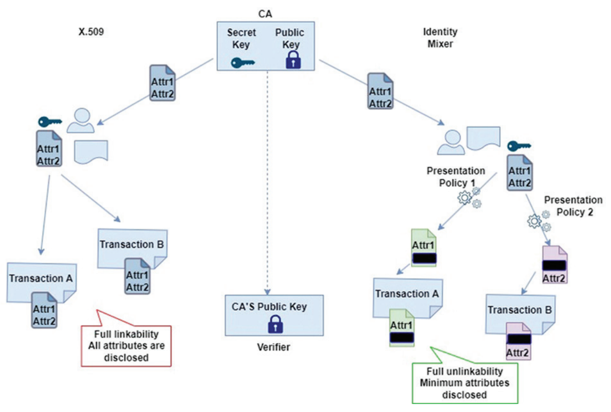

Fig. 3. Comparison with X.509

4.6 ARWEAVE DATA STORAGE

IPFS is a well-known P2P network for decentralized data exchange. Utilizing Arweave can provide permanent and tamper-proof data storage. The data is maintained and stored by a decentralized network of nodes, and they can be permanently lost if no one hosts them. We have integrated Arweave off-chain data storage technology with Hyperledger Fabric and kept encrypted data on the Arweave and the hash to the blockchain. For storing a file on Arweave using Java, read the file into a byte array and then create a transaction object with the byte array.

Example

```
Path path = Paths.get(Pathoffile);
byte[] data = Files.readAllBytes(path);

Gateway gateway = new
GatewayImpl("https://arweave.net");
Wallet wallet = Wallet.fromKeyFile(keyFilePathfromwallet);
Web3cTransaction tx = new Web3cTransaction
```

(wallet, gateway);

tx.addTag("Content-Type", mimeType);

tx.setData(data);

Upload upload = gateway.upload(tx);

5. FABRIC COMPONENTS

5.1. COMPONENTS

To develop an end-to-end blockchain system using Hyper Ledger Fabric, a complex workflow is required. The following components helped to construct the Fabric integrated permissioned EHR sharing framework [17]:

Component	Description
System	Ubuntu 18.04
Hyperledger Fabric SDK	Fabric-Java-SDK v2.4
No. of Healthcare organizations	Five organizations
No. of peers per org.	2-3
Ordering Service	Raft
World State DB	CouchDB
Concept	Docker-based
Chaincode language	Java
Front End	The web interface for the client application was created using HTML, CSS, and JavaScript
Other Integration	Hyperledger Explorer and Caliper
Access Control	It is used to restrict access to smart contract transactions based on role.
CA and MSP	X.509 certificates
Identity Mixer with ZKP	Achieve user anonymity and unlikability.
Transient field	For passing the symmetric key to chaincode through the docker as well as all the input parameters of the transaction.
Private Data Collection	For each organization to store permission lists and any private data of two or more organizations.
Chaincode	Fine-grained attribute-based access control using chaincode. 1. Create medical record 2. Save data 3. Update medical record 4. Allow doctor write 5. Update ownership (patient gives consent to another doctor) 6. Doctor fetches patient data
Encryption	Public key encryption is made possible by using Elliptical Curve Cryptography (ECC)
Docker Compose	A tool for distributing containers, which are software packages (docker-compose-ca.yaml, docker-compose-couch.yaml, docker-compose-net.yaml).

5.2 CHAINCODE SAMPLE

Create Medical Record

```
public String createMedicalRecord(Context ctx, String patientName, int age, String medicalHistory, String medications, String testResults, String owner)
{
ChaincodeStub stub = ctx.getStub();
// Generate a new ID for the medical record
String id = stub.getTxId();
// Create a new medical record object
MedicalRecord medicalRecord = new MedicalRecord(id, patientName, age, medicalHistory, medications, testResults, owner);
// Convert the medical record object to JSON
String medicalRecordJSON = gson.toJson(medicalRecord);
// Save the medical record on the blockchain
 stub.putStringState(id, medicalRecordJSON);
return id;
}
```

Fetching Patient Data

```
public String getMedicalRecord(Context ctx, String id)
{ ChaincodeStub stub = ctx.getStub();
// Retrieve the medical record from the blockchain
String medicalRecordJSON = stub.getStringState(id);
if (medicalRecordJSON == null || medicalRecordJSON.isEmpty())
{
throw new ChaincodeException("Medical record not found");
}
// Convert the medical record JSON to an object
MedicalRecord medicalRecord = gson.fromJson(medicalRecordJSON, MedicalRecord.class);
return medicalRecordJSON;
}
```

Update ownership of a patient's medical record

```
public void updateOwnership(Context ctx, String id, String newOwner) {
ChaincodeStub stub = ctx.getStub();
// Retrieve the medical record from the blockchain
String medicalRecordJSON = stub;
        if (medicalRecordJSON == null ||
medicalRecordJSON.isEmpty())
{
        throw new ChaincodeException("Medical record not found");
}
// Convert the medical record JSON to an object
MedicalRecord medicalRecord =
gson.fromJson(medicalRecordJSON, MedicalRecord.class);
// Update the medical record with the new owner
medicalRecord.owner = newOwner;
// Convert the medical record object to JSON
medicalRecordJSON = gson.toJson(medicalRecord);
// Save the updated medical record on the blockchain
stub.putStringState(id, medicalRecordJSON);
}
```

6. PROPOSED ARCHITECTURE

The proposed privacy preserving EHR architecture using Hyperledger Fabric is shown in Figure 4. Our Framework has implemented privacy-preserving mechanisms of the Hyperledger Fabric to preserve data and user privacy on the network during transaction consensus. In our system, the patient has complete ownership of their healthcare data. The study offers effective access control between patients and doctors through encryption techniques. Our approach is a tamper-resistant mechanism as we have stored only metadata (hashed data and data reference URLs) for every healthcare transaction in the blockchain. Other pieces of information will be encrypted using AES before storing it to the database. Keys are encrypted using ECC (Elliptical Curve Cryptography) to fetch information. Idemix with ZKP is stacked on top of the Hyperledger Fabric to achieve unlinkability and anonymity.

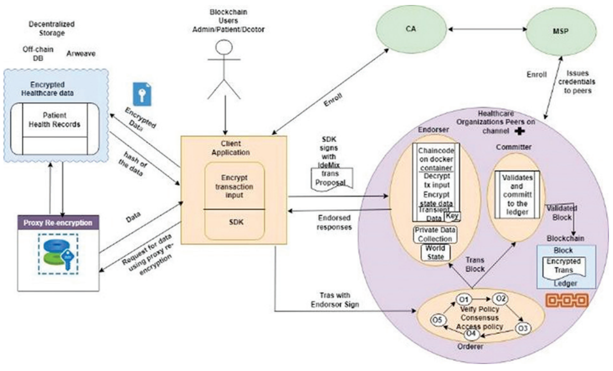

Fig 4. Hyperledger Fabric integrated Blockchain architecture for EHR sharing system

Explanation

This section describes the entire process of the client submitting the transaction to build a block on the blockchain, confirming consensus, storing EHR files to Arweave storage, and reading files from Arweave, as shown in Fig. 1. After the patient's approval, a doctor can add disease, prescription and other information. This information will be stored in the private data collection of that organization. Permission lists are stored on the peer's CouchDB and heavy files such as reports will be stored on Arweave data storage. Only hashes of the transaction will be stored on the blockchain.

The below example describes the architecture:

1. First, the client submits the transaction with encrypted input data. Suppose a patient wants to give data access consent to another doctor. This request with input data of a transaction such as patient name, doctor name, and organization name are passed through the transient field for privacy preservation purposes. Each participant will employ identity certificates associated with Idemix to carry out any action on the distributed ledger.
2. This transaction will be passed to the identified endorsers and endorsed according to the endorsement policy. The endorsement policy specifies that this transaction must be endorsed through the doctors' organization and the patient data provider organization. All Endorsers endorse transactions by executing chain code and providing consent to the doctor for a specified time request. The fetched patient data will be stored on private data collection on a particular healthcare organization's peer network. Later, the same data will be fetched using proxy re-encryption from the Arweave data storage. Endorsers apply the corresponding changes to a state database that maintains a snapshot of the current world state.
3. After endorsement, this signed transaction will be submitted to the client's application. If transient data are passed in a transaction, then only chaincode can access them, decrypt them and store them in the private data collection of the organization/s. Before sending transactions to the ordering service and adding blocks to the ledger, values within the chaincode can be encrypted using AES encryption to protect the data. So, only a person who has access to the appropriate key can decrypt encrypted data.
4. Then the client submits this transaction to the ordering service with read/write sets and signatures from the endorsers.
5. Ordering Service orders the transaction according to the channel and puts them into blocks and sends the block to peers.
6. Peers (Endorsers & Committers) validate each transaction in the block by checking whether the transaction has the necessary endorsements or not. If the transaction satisfies an endorsement policy, then the transaction will be marked as "valid" in a block and respective states will be updated in a state database by reading the write sets. If the transaction fails to satisfy the policy, then it will be marked as "invalid" in a block and there will be no state changes in the state database.

7. EVALUATION

In the last few years, only a few Hyperledger Fabric blockchain-based permissioned EHR solutions have been proposed, with each claiming that it revolutionizes the way of transaction processing along with ensuring security and privacy. However, none of them has utilized IdeMix in Fabric Java SDK. To compare our study with existing research, we analyzed four parameters: 1) Data Security and Scalability, 2) Data Privacy and GDPR, 3) Transaction Data Privacy during consensus, and 4) User Anonymity.

Data Security and Scalability

The problem with submitting confidential information to the ledger is that it will be available to everyone. Few existing studies have proposed a solution to overcome this problem. They have utilized off-chain technologies to store encrypted heavy files on a distributed network such as IPFS and put only the hash of the private data on the ledge. However, IPFS is not a truly decentralized and permanent solution. In our proposed solution, we have utilized Arweave with the Fabric-based HER system to store data permanently to make the EHR system scalable and immutable.

Data Privacy and GDPR

The existing Fabric-implemented EHR studies have neither implemented any privacy control mechanisms nor do they comply with GDPR. Few existing systems partially mitigate this limitation by storing the medical records on IPFS, which supports data deletion. However, metadata is kept on-chain. So even if the patient requests its removal, it cannot be executed. We have implemented data privacy demands using Private Data Collections and Transient Fields. By implementing this feature, organizations transact with one another on the same channel without worrying that another organization will discover their confidential information. It decreases the number of channels while enabling data privacy inside the channel. Also, we have stored data on Arweave storage where the user can delete data if no longer needed. To delete a JSON file that has been uploaded to the Arweave network, we can create and post a new transaction that references the original transaction ID and sets its data to an empty buffer.

Transaction/Data Privacy during consensus:

The notion of privacy always complicates the notion of replication and decentralization. Even though the Fabric network is permissioned, and the healthcare organization's sensitive information is stored on off-chain storage, sensitive information about a transaction can be leaked. In a Fabric-based HER system, data privacy is concerned with the privacy of input parameters of a transaction, state data privacy (output of a transaction), and smart contract privacy. While submitting a transaction, the client passes input parameters and other information. Later, the endorser executes the chain code based on the input parameters and prepares state data for the transaction. If this transaction-related information is visible to the orderer or other nodes of the channel, it may result in the leakage of private data. We have implemented chaincode to share the confidential data on-chain via a transient field and private data is passed using private data collection of the other organization.

User privacy during transaction consensus:

User privacy is concerned with user anonymity and transaction unlinkability. In Fabric, all the channel participants are aware of each other's identities, which can compromise user privacy. 1) User anonymity means hiding user-level credentials to achieve user privacy in the ledger. In Fabric, transactions are endorsed by endorsers before going through validation. During an endorsement, an endorser specifies its identity with its signature, to be later validated by other nodes. However, revealing the identity of an endorser can compromise patient privacy in transactions. Such as when patient grants read permission rights to the doctor of another organization. Revealing the identity of endorsers will leak the organization membership of their transactor, which in turn, leaks patient disease-related information through the identity of the organization. 2) Transaction unlinkability is the ability of a transactor to throw multiple transactions without being linkable. In Fabric, endorsers would know which clients had submitted transactions, and the orderers would know which endorsers had endorsed transactions. All this information could reveal sensitive and confidential information about a patient. Since the EHR system deals with patients' confidential information, this sensitive data must be kept secret for some users and hidden from other network users. For resolving these concerns, we have implemented Idemix cryptographic protocol in Hypeledger Fabric Java SDK v2.4 and achieved unlinkability and anonymity.

Below in Table 2, the proposed framework is compared to existing Fabric-integrated permissioned EHR solutions in terms of Java SDK, Raft Ordering, Transient Field for passing input parameters of a transaction, PDC for private data sharing between organizations, IdeMIX protocol for user privacy and Arweave for permeance of data. The comparison shows our study has implemented all the features to make a robust and secure EHR sharing system.

Table 2. Comparison with existing Fabric-integrated EHR systems.

Ref.	[7]	[8]	[9]	[10]	[11]	[12]	[13]	Proposed Model
Fabric Java SDK	N	N	N	N	N	N	N	Y
Raft ordering	N	N	N	N	N	N	N	Y
Transient Filed	N	N	N	N	N	N	N	Y
Private Data Collections	N	N	N	N	N	Y	N	Y
IdeMix	N	N	N	N	N	Y	N	Y
Arweave	N	N	N	N	N	N	N	Y

8. CONCLUSION

In this research study, we reviewed several existing studies that aim to address privacy of EHR data. However, we found that most of these studies failed to address user anonymity and transaction unlinkability concerns. Our proposed solution suggests a permissioned EHR that makes use of Hyperledger Fabric and its privacy-preserving features to address the critical issues regarding EHR user privacy and transaction unlinkability in the network. Our solution uses Identity Mixer using Fabric's Java SDK V2.4 to guarantee transaction anonymity and unlinkability in the network. Moreover, data confidentiality between a subset of organizations on the network using private data collections and transient fields has been implemented. Our proposed solution addresses these challenges and provides a robust method for preserving patients' data, paving the way for an improved patient experience and a digital revolution in the healthcare sector. In our future work, we will try to implement quasi-anonym endorsement to accomplish privacy and anonymity of transactions in Fabric.

9. REFERENCES

[1] C. C. Agbo, Q. H. Mahmoud, J. M. Eklund, "Blockchain technology in healthcare: a systematic review", Healthcare, Vol. 7, No. 2, 2019, p. 56.

[2] J. Kaur, R. Rani, N. Kalra, "Blockchain-based framework for secured storage, sharing, and querying of electronic healthcare records", Concurrency and Computation: Practice and Experience, Vol. 33, No. 20, 2021, p. e6369.

[3] I. Yaqoob, K. Salah, R. Jayaraman, Y. Al-Hammadi, "Blockchain for healthcare data management: opportunities, challenges, and future recommendations", Neural Computing and Applications, Vol. 34, No. 14, 2022, pp. 11475-11490.

[4] T. T. Kuo, H. E. Kim, L. Ohno-Machado, "Blockchain distributed ledger technologies for biomedical and health care applications", Journal of the American Medical Informatics Association, Vol. 24, No. 6, 2017, pp. 1211-1220.

[5] Z. Leng, Z. Tan, K. Wang, "Application of hyperledger in the hospital information systems: A survey", IEEE Access, Vol. 9, 2021, pp. 128965-128987.

[6] E. Androulaki et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains", Proceedings of the Thirteenth EuroSys Conference, Porto, Portugal, 23-26 April 2018, pp. 1-15.

[7] J. Huang, Y. W. Qi, M. R. Asghar, A. Meads, Y. C. Tu, "MedBloc: a blockchain-based secure EHR system for sharing and accessing medical data", Proceedings of the 18th IEEE International Conference on Trust, Security and Privacy In Computing And Communications / 13th IEEE International Conference on Big Data Science And Engineering, Rotorua, New Zealand, 5-8 August 2019, pp. 594-601.

[8] V. Mahore, P. Aggarwal, N. Andola, S. Venkatesan, "Secure and Privacy Focused Electronic Health Record Management System using Permissioned Blockchain", Proceedings of the IEEE Conference on Information and Communication Technology, Allahabad, India, 6-8 December 2019, pp. 1-6.

[9] A. Dubovitskaya et al. "ACTION-EHR: patient-centric blockchain-based electronic health record data management for cancer care", Journal of Medical Internet Research, Vol. 22, No. 8, 2020, p. e13598.

[10] D. Tith, J. S. Lee, H. Suzuki, W. M. A. B. Wijesundara, N. Taira, T. Obi, N. Ohyama, "Application of blockchain to maintaining patient records in electronic health record for enhanced privacy, scalability, and availability", Healthcare Informatics Research, Vol. 26, No. 1, 2020, pp. 3-12.

[11] S. Chenthara, K. Ahmed, H. Wang, F. Whittaker, Z. Chen, "Healthchain: A novel framework on privacy preservation of electronic health records using blockchain technology", Plos One, Vol. 15, No. 12, 2020, p. e0243043.

[12] C. Stamatellis, P. Papadopoulos, N. Pitropakis, S. Katsikas, W. J. Buchanan, "A privacy-preserving healthcare framework using hyperledger fabric", Sensors, Vol. 20, No. 22, 2020, p. 6587.

[13] V. Mani, P. Manickam, Y. Alotaibi, S. Alghamdi, O. I. Khalaf, "Hyperledger healthchain: patient-centric IPFS-based storage of health records", Electronics, Vol. 10, No. 23, 2021, p. 3003.

[14] Private data, https://hyperledger-fabric.readthedocs.io/en/release-2.5/private-data/private-data.html (accessed: 2023)

[15] Transient Field, https://hyperledger-fabric.readthedocs.io/en/release-2.5/private-data-arch.html#how-to-pass-private-data-in-a-chaincode-proposal (accessed: 2023)

[16] MSP Implementation with Identity Mixer, https://hyperledger-fabric.readthedocs.io/en/release-2.5/idemix.htmll (accessed: 2023)

[17] A Blockchain Platform for the Enterprise, https://hyperledger-fabric.readthedocs.io/en/release-2.2/ (accessed: 2022)