Proceedings of the
Third Workshop on Blockchain
Technologies and its Applications

February 04 – 07, 2019

TECHNICAL REPORT

Information Security Research & Development Centre (ISRDC)
Department of Computer Science and Engineering
IIT BOMBAY

Editors:
Dr. Vishwas Patil, IIT Bombay
Dr. NV Narendra Kumar, IDRBT Hyderabad
<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
<th>Speaker(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10:30 - 11:00</td>
<td>High Tea</td>
<td>Navin Gupta, Managing Director, Ripple</td>
</tr>
<tr>
<td>11:00 - 12:30</td>
<td>Technical Session 1</td>
<td>Ashwin Sekhari, NIT Rourkela, Anasuya Acharya, IIT Bombay</td>
</tr>
<tr>
<td>12:30 - 14:00</td>
<td>Lunch Break</td>
<td></td>
</tr>
<tr>
<td>14:00 - 15:30</td>
<td>Technical Session 2</td>
<td>Joji Varghese, Print2Block, Shachindra, REVOLtic Engineering</td>
</tr>
<tr>
<td>15:30 - 16:00</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>16:00 - 16:30</td>
<td>Keynote 2</td>
<td>AS Ramasastri, Director, IDRBT Hyderabad</td>
</tr>
<tr>
<td>16:30 - 17:30</td>
<td>Panel Discussion</td>
<td>Panel Members: NV Narendra, V Muthu, Bharath Bhusan, Ajay Lande, Pramod Nair, Mayank Lau, Vinay Ribeiro, Moderator: Vishwas Patil</td>
</tr>
<tr>
<td>17:30 - 18:00</td>
<td>High Tea</td>
<td></td>
</tr>
<tr>
<td>09:00 - 09:30</td>
<td>Registration</td>
<td></td>
</tr>
<tr>
<td>09:30 - 10:10</td>
<td>Keynote 1</td>
<td>Shachindra, Sagar Ganiga, Shreya Saha, Anish Mishra, Meil Maheshwari, Gaurav Kumar</td>
</tr>
<tr>
<td>10:15 - 10:45</td>
<td>Keynote 4</td>
<td>SVM Nepal, CSIRO Australia</td>
</tr>
<tr>
<td>10:45 - 11:15</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>11:15 - 13:00</td>
<td>Industry Session</td>
<td>Sarthak Gaurav, Surya Nepal, Salil Kanhere, Rajesh Chauhan, Shyamalesh Choudhury, Abhijit Singh, Moderator: NV Narendra</td>
</tr>
<tr>
<td>13:00 - 14:00</td>
<td>Lunch Break</td>
<td></td>
</tr>
<tr>
<td>14:00 - 15:30</td>
<td>Technical Session 3</td>
<td>Prachi Gupta, Wipro</td>
</tr>
<tr>
<td>15:30 - 16:00</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>16:00 - 17:00</td>
<td>AIC Meeting</td>
<td>AIC Project Collaborators</td>
</tr>
<tr>
<td>09:30 - 09:40</td>
<td>Welcome</td>
<td></td>
</tr>
<tr>
<td>09:40 - 10:30</td>
<td>Keynote 5</td>
<td>Vallipuram Muthukumarasamy, Griffith University, Australia</td>
</tr>
<tr>
<td>10:30 - 11:00</td>
<td>Invited Talk</td>
<td>Praveen Jayachandran, IBM Research Labs India</td>
</tr>
<tr>
<td>11:30 - 12:00</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>12:00 - 13:00</td>
<td>Technical Session 4</td>
<td>Jagrendra Singh, Chainworks Digital LLP, Niviera Technologies</td>
</tr>
<tr>
<td>13:30 - 14:30</td>
<td>Lunch Break</td>
<td></td>
</tr>
<tr>
<td>14:30 - 17:30</td>
<td>AIC Meeting</td>
<td>AIC Project Collaborators</td>
</tr>
</tbody>
</table>

Day 2 (February 05, 2019) @IIT Bombay

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
<th>Speaker(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00 - 09:30</td>
<td>Registration</td>
<td></td>
</tr>
<tr>
<td>09:30 - 10:15</td>
<td>Keynote 3</td>
<td>Surya Nepal, Data 61, CSIRO Australia</td>
</tr>
<tr>
<td>10:15 - 10:45</td>
<td>Keynote 4</td>
<td>Rama Vedashree, CEO, DSCI</td>
</tr>
<tr>
<td>10:45 - 11:15</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>11:15 - 13:00</td>
<td>Industry Session</td>
<td>Sarthak Gaurav, Surya Nepal, Salil Kanhere, Rajesh Chauhan, Shyamalesh Choudhury, Abhijit Singh, Moderator: NV Narendra</td>
</tr>
<tr>
<td>13:00 - 14:00</td>
<td>Lunch Break</td>
<td></td>
</tr>
<tr>
<td>14:00 - 15:30</td>
<td>Technical Session 3</td>
<td>Prachi Gupta, Wipro</td>
</tr>
<tr>
<td>15:30 - 16:00</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>16:00 - 17:00</td>
<td>AIC Meeting</td>
<td>AIC Project Collaborators</td>
</tr>
</tbody>
</table>

Day 3 (February 07, 2019) @IDRBT Hyderabad

<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
<th>Speaker(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>09:00 - 09:30</td>
<td>Registration</td>
<td></td>
</tr>
<tr>
<td>09:30 - 10:30</td>
<td>Keynote 5</td>
<td>Vallipuram Muthukumarasamy, Griffith University, Australia</td>
</tr>
<tr>
<td>10:30 - 11:00</td>
<td>Invited Talk</td>
<td>Praveen Jayachandran, IBM Research Labs India</td>
</tr>
<tr>
<td>11:30 - 12:00</td>
<td>Tea Break</td>
<td></td>
</tr>
<tr>
<td>12:00 - 13:00</td>
<td>Technical Session 4</td>
<td>Jagrendra Singh, Chainworks Digital LLP, Niviera Technologies</td>
</tr>
<tr>
<td>13:30 - 14:30</td>
<td>Lunch Break</td>
<td></td>
</tr>
<tr>
<td>14:30 - 17:30</td>
<td>AIC Meeting</td>
<td>AIC Project Collaborators</td>
</tr>
</tbody>
</table>
Enhancing Security and Privacy of Permissioned Blockchain using Intel SGX

Batsayan Das  
*TCS Innovation Labs*  
Kolkata, India  
batsayan.das@tcs.com

Srujana Kanchanapalli  
*TCS Innovation Labs*  
Bengaluru, India  
srujana.k@tcs.com

Vigneswaran R  
*TCS Innovation Labs*  
Chennai, India  
vigneswaran.r@tcs.com

Abstract—Permissioned Blockchain systems are envisioned to have immense impact in various sectors such as Finance, Supply Chain, IoT, Insurance and so on. However, the practical aspects and deployment has seen slower progress. One of the critical concerns restraining the practical deployment of blockchains is security and privacy of data and computations within the applications. Our research focuses on making Blockchain more practically deployable by enhancing privacy, security and scalability. In this direction, we leverage Trusted Execution Environment (TEE) to solve several practical challenges in Hyperledger Fabric, an open source permissioned blockchain. We employed Intel's Software Guard Extension (SGX) to improve scalability, privacy and security of the Enterprise Blockchain Network. And implemented a practical process as layer above (on the top) of Hyperledger Fabric and Intel SGX.

Index Terms—Permissioned Blockchain, Enterprise Applications, Hyperledger Fabric, Intel SGX, Intel SGX Protected Code Loader (PCL), TEE, Security, Privacy, Scalability

I. INTRODUCTION

Blockchain is a shared ledger that enables mutually distrusting parties to transact with each other without any central authority. The participants form a peer-to-peer network of nodes enforcing a common ledger of transactions. Fig. 1 shows the sample blockchain network. Blockchain, as its name implies, is a chain of blocks. Each block consists of a set of entries (financial transactions in case of cryptocurrencies) to be included in the blockchain and each new block is chained to the preceding block. All entries are appended to the ledger based on the consensus (agreement) of the involved parties. This ensures that the ledger is always consistent among all the parties.

The properties of a Blockchain are as follows:

- Decentralization: No central authority
- Transparency: Shared ledger
- Integrity: Immutable ledger
- Availability: Peer-to-peer network

A blockchain [13], as used in most cryptocurrencies, does not require any authorization for participants to join or leave the system, and hence is referred to as a permission-less blockchain. However, enterprise applications cannot operate in such models. Enterprise applications operate in a regulated and controlled environments with verified identities for all the parties involved. Hence, the permissioned blockchain models [14] have evolved over the time.

 Enterprises are keen on blockchain technology implementation on different sectors (like, supply chain, IoT, Identity management, Insurance, Healthcare and so on). These Permissioned Blockchain systems are envisioned to have immense impact in various sectors. However, the practical aspects and deployment has seen slower progress. One of the critical concerns restraining the practical deployment of blockchains is security and privacy of data and computations within the applications. Our research work focuses on making Blockchain more practically deployable by enhancing privacy, security and scalability. This paper first explores the challenges for the enterprise blockchain networks and the second part explores about Intel SGX a Trusted Execution Environment (TEE) and then our work of implementing the Blockchain with Intel SGX for enterprise application with a secure layer of communication channel.

II. SECURITY, PRIVACY AND SCALABILITY IN BLOCKCHAINS

Making blockchain which is practically deployable involves solving several challenges such as scalability, privacy and security.

A. Security of Computations

The parties (nodes) in blockchain network append entries to the ledger after agreement on validation. The validation
involves business logic that asserts if the entry is valid or invalid. Such business logic is required to guarantee that the validation is sound and secure.

The guarantee involves:
- Security of private computations- The business logic is confidential if necessary. Several enterprise applications require the business logic to remain confidential to protect their business interests.
- Correctness of computations- The correctness of business logic execution has to be guaranteed to ensure robustness of the system. A more trusted execution process embeds higher trust into the system.

B. Privacy of data

Most of the enterprise applications mandate privacy requirements even within the organizations. Besides, the blockchain applications usually involve several organizations transacting with each other and hence require more stringent privacy guarantees. The applications involve computing together on each others’ data while still preserving the privacy. This requires mechanisms to facilitate computations on private data.

C. Scalability

As noted earlier, the entries to the common ledger are appended only after the parties validate the transactions independently and communicate agreement with each other. This induces significant latency into the system, hampering the overall throughput of the applications. Hence, mechanisms to increase the throughput are necessary.

We focus on solving the above mentioned challenges by using trusted execution environment. In particular, we employ Intel’s Software Guard Extension (SGX) [8] to improve scalability, privacy and security in permissioned blockchain setting.

III. INTEL SGX

A TEE enforces trust in an untrusted environment through software and hardware. Intel’s SGX (a set of extensions to Intel’s architecture) [4] [1] provides an isolated and secure execution environment called enclave for a user program without trusting any privilege software (such as hypervisor or operating system). A TEE guarantees privacy and confidentiality (of both data/code and computation) with hardware security. The computations are performed in a secure container (enclave), which is deployed at the untrusted remote machine with the help of remote attestation. The enclave creation service runs at privilege level 0 (Ring 0) which is the most privileged level for any operating system. The deployment of instructions with Ring 0 privilege ensures higher security guarantees and more trust.

A. Application Design

Application design with Intel SGX requires that the application be divided into two components:
- Trusted component: Trusted application part of Intel SGX is the enclave. The instructions residing in the trusted enclave are isolated from the other application environment.
- Untrusted component: Untrusted component comprises of rest of the application and any of its modules. It is important to note from the standpoint of an enclave, operating system and Virtual Machine Manager (VMM) are considered untrusted components.

For better security it is desirable to structure minimal trusted-untrusted components interaction. While enclaves can leave the protected memory region and call functions in the untrusted component (through the use of a special instruction), limiting these dependencies will strengthen the enclave against potential attacks.

B. Basic Control Flow of SGX

![Fig. 2. Basic flow of control with SGX](image-url)

Basic control flow of SGX [2] as shown in the Fig. 2. The important components of the flow are described as follows:
- Untrusted: It refers to code or construct that runs in the application environment outside the enclave.
- Trusted: It refers to code or construct that runs in the TEE inside the enclave.
- ECALL: A call from the application into an interface function within the enclave.
- OCALL: A call made from within the enclave to the application.
- Untrusted Run-Time System (uRTS): Code that executes outside the enclave environment and performs functions such as loading and manipulating an enclave (For example, destroying an enclave). Making calls (ECALLs) to an enclave and receiving calls (OCALLs) from an enclave.
- Trusted Run-Time System (tRTS): Code that executes within the enclave environment and performs functions such as receiving calls (ECALLs) from the application and making calls outside (OCALLs) the enclave. It helps in managing the enclave.
C. Features

There are three main functionalities (Fig. 3) that enclaves achieve listed as follows:

- **Isolation** - Instructions and data inside the enclave protected memory cannot be read or modified by any process external to the enclave. Hence, data and computation inside the execution are isolated from the untrusted application thus preserving privacy of data and computation.

- **Sealing and Unsealing** - The enclave platform uses secret key to encrypt data to the platform (that is, sealing), or to decrypt data already on the platform (that is, unsealing). Data passed to the host environment is encrypted and authenticated with a hardware-resident key. The data can only be accessed within the enclave when necessary. This limits the exposure of private application-critical data within the trusted zone.

- **Attestation** - A special signing key and instructions are used to provide an unforgeable report attesting to instructions, static data, and (hardware-specific) meta-data of an enclave, as well as outputs of computations performed inside the enclave. The attested results from the enclave guarantee correctness of the computation.

These features of Intel SGX can be leveraged to solve the challenges mentioned in the previous section.

IV. HYPERLEDGER FABRIC WITH INTEL SGX

A. Application design

This section describes architecture and application flow of the client application interacting with enclave environment.

1) Architecture: Our design realizes leveraging Intel SGX in the Hyperledger Fabric (HLF) [12] environment through design of a security layer that enables communication between the blockchain application and the Intel SGX.

- Fabric Node is a client application (HLF BC network node executes both chaincode and fabric code) which is executed on a machine (or node).
- The application’s Untrusted part.
- The application’s Trusted part.

2) Application Flow: - Fabric chaincode (application code) sends the request to the untrusted part.
- Untrusted part checks and send the required ECALL request to the trusted part.
- Trusted part executes the request and sends the reply to the untrusted part.
- Untrusted part sends the response back to the fabric chaincode.

Note that all this communication is under secure channel.

B. Security layer implementation

This design model has an SGX application (which has untrusted and trusted parts), and a Blockchain Node (which we use as client, representing blockchain network). This section explains the secure layer design implementation.

This section explains the implementation details in two modules; first module explains the key generation and client registration process, and the second module explains the data sharing between client (Blockchain Node) and SGX (Enclave or Trusted) using session key and Rivest-Shamir-Adleman (RSA) key.

This secure layer is implemented and tested with Intel SGX Protected Code Loader (PCL) [10]. The Intel SGX PCL [11] is intended to protect Intellectual Property (IP) within the code for Intel SGX enclave applications running on the Linux OS. The enclave shared object is encrypted at build time. It is decrypted at enclave load time. So, adversaries cannot reverse engineer the binary enclave shared object.

1) Module 1: Key Generation and User Registration: In this module the following activities will be performed:
- Key Generation: RSA key pair is generated at Client and Enclave side. - User/Client Registration: In this process a Client ID and clients Pub.key is shared to the enclave. Similarly, enclaves Pub.key is shared with client for securing the subsequent communications.

Fig. 5 shows the flow of Key Generation and Client Registration process.

2) Module 2: Data sharing between Client and Enclave (SGX): In this module, client shares the payload or data with the enclave (SGX) using our secure communication design. Data is shared between enclave and client in a secure way using session key. This session key is randomly generated symmetric key. A new session key is generated for every transaction (for sharing inputs or payloads) to communicate in a secure way. After enclave receives the payload, it executes the business logic and send the result back to client in a secure way.

Fig. 6 shows the process flow of data sharing between Client and Enclave.

3) Future proposal - Design proposal for Future Implementation:

Signature will help in avoiding man in the middle attack in case of Client Registration as well as Data or payload Sharing process of our secure layer communication. This section is very similar to the Module 2. Additionally, signature will be passed to enclave for verification. The Fig. 7 shows the flow and steps involved in the process of data sharing with signature.

C. Realizing our Blockchain objectives

We now describe how the features of Intel SGX realizes our objectives of improving security, privacy and scalability in blockchains.

1. Security Of Computations: Security of computations in blockchain applications can be attributed to security of private computation and correctness of computation.

The independent parties in the blockchain network may require to preserve the confidentiality of computations. Hence, it is not possible to share such computations with other parties for transaction validation. The TEE executes the business logic and attests the result guaranteeing the correctness of the computation. Thus, the TEE ensures security of private computations and embeds trust in the correctness of business logic execution.

2. Privacy of data: Privacy of data is of utmost importance in applications involving several parties. Blockchain applications require several parties computing together on each other’s data while still preserving the privacy.

The isolation of TEE facilitates preserving the privacy of data while allowing computations on data. The involved parties can provide encrypted private data to the TEE. The enclave (TEE) decrypts the data within the trusted zone and computes necessary operation on the data. The TEE then sends attested
results that can be verified and accepted by the parties without revealing private information to each other.

3. Scalability: The entries to the common ledger are appended after the parties validate the transactions independently and communicate agreement with each other. This induces significant latency into the system hampering the overall throughput of the applications.

TEE can be leveraged to improve the scalability in network. Instead of individual parties validating the transactions, a TEE (enclave) can be employed to validate the transaction and endorse the validity. This way, all the parties need not validate transactions independently, rather just verify the endorsement of the enclave before appending the transaction into the blockchain. This decreases the communication rounds necessary for parties to reach agreement on the validity and improves the transaction throughput in the network.

V. CONCLUSION

We leverage Intel SGX, to improve the privacy, security and scalability of blockchain applications. Though there are other alternate solutions such as Zero-Knowledge proofs and verifiable computing, a trusted execution environment is the most practical solution in terms of viability and efficiency. Our security layer implementation can be utilized in any blockchain application by appropriately scheming the application to utilize our feature implementation to improve security, privacy and scalability in blockchain applications.

ACKNOWLEDGMENT

We thank Mr. Sitaram Chamarty, Dr. Rajan M A, and Mr. Nitesh Emmadi from Tata Consultancy Services for their insightful feedback. We also thank the other people who reviewed and shared their valuable comments in improving the paper.

REFERENCES

Abstract—Land is an immovable property, and to prove the ownership of the land. Ideally, one must provide a legal document that should conclusively prove it. But, land record in India is a loosely defined term. Here, one is presumed to be the owner, until proven otherwise. Land record is a generic term that could mean a record of rights, Khasra, Shajra, etc. Moreover, different documents are maintained by different departments, and any one of them could be used to stake a claim on the land. This leads to numerous disputes, and it is evident from the number of cases, which are pending in Indian courts. In this paper, we aim to solve this problem of coordination between the various departments, and multiplicity of land records, via entanglement of blockchains.

Index Terms—Blockchain, Hyperledger, Land Records, Record of Rights

I. INTRODUCTION

Indian land record system is still inspired by the one introduced by Raja Todarmal (one of the 9 gems in the court of Akbar). This land record maintenance system is gripped with many loopholes. Firstly there are numerous records, each of which could potentially be produced to have the claim on the land. Many of these records could be mutated without the consent of the owner. Secondly, different records related to land are managed by different departments and hence they are not always on the same page. This has lead to numerous disputes for the ownership of the land. Thirdly, India does not have a uniform system for records related to land. Land is a state subject, and each state have different practice related to land. Lastly, much of these records are not digitized, and create huddle in coordination across the departments

With the advancement in information technology, Some data pertaining to land record is available in digital form [1] and some is in progress. These records are going to be digitized soon which will improve the existing business process like land registration, mutation, etc. But the problem of integrity and coordination still remains. Moreover, these documents would be prone to cyber attacks.

In the last few years, There is an increase in the number of cyber attacks. As per the Breach Level Index (BLI), There are approx 3 million records which have been compromised in the first half of the year 2018 [2]. With the rise of e-governance, government services will be available to citizens using information and communication technology (ICT) which might attract the adversaries and results in an increase in number of cyber attacks [3]. This technological advancement has raised information security concerns with the land records where the integrity of the ownership of the real property is the main target which results in increased number of land disputes, social restless etc. In India, it takes several years to get a dispute resolved [4] [5].

A blockchain based framework can be a solution to maintain the integrity of record of rights, and it can be useful while detecting an insider attack [6] and to incorporate the traceability. In this paper, we will present an implementation of Blockchain on Indian Land records system. We would not only increase the security and integrity of the land records, but would also enhance coordination of the department via this technology. As depicted in fig. 1. There are mainly three departments are involved in the process of maintaining the record of rights. Land records in India have been very poorly defined, and here a person is presumed to be the owner until proven otherwise. Numerous document like Khasra, Khatauni, Shajra, Registration document, parwana could refer to the Land record. Due to this improper definition, India has a huge amount of land dispute cases. Hence we want to create a solution that could bring all these land records under one system, and at the same time maintain its integrity.

Our contribution of this article is to develop a blockchain based solution 1). To make the land records tamper-resistant, 2). To enhance the coordination across department with blockchain 3). To expedite the business process. The rest of the paper is organized as section II.
will brief the existing posture of land records. Section III will brief the proposed solution based on blockchains. Section V will conclude our work.

II. BACKGROUND AND RELATED WORK

In previous section we showed how land records in India are vague, inefficient and prone to manipulation. To tackle this issue, India has started digitizing land records, but just digitizing them and securing them cannot be fruitful. We need to ensure integrity as well as the coherence of the land records. All of the land records must be changed at the same time and records must be updated simultaneously by every department. We propose a solution based on the entanglement of the blockchains, where every department have records on their own chain, but they are entangled so that change of record by one department, would force the change of record by another department. In this section, we list the work done by governments in India for the digitization of the land records, and then we show how blockchain technology suits our need.

“Haryana land record information system (HALRIS)”, A project of Haryana government digitizing land records since 2000 [7]. Another project “Nemmadi” initiated by the government of Karnataka in 2004 has “Bhoomi” (meaning land in Kannada) program which expedites the digitization of land records and digitized around 20 million records of land ownership of 6.7 million farmers in the state [7]. Board of Revenue (Uttar Pradesh) modifies land reforms policies and implemented “BHULEKH” project in which all the Land Records of the state in each 312 tehsils of 71 districts have been computerized [7]. Chhattisgarh started “Chhattisgarh Online Information for Citizen Empowerment (CHOICE)” project to provide service like land records. More states are replicating similar projects to digitized the land records.

The Government of India has approved the National e-Governance Plan (NeGP) in 2006 which includes “Land Records” as a state project (comes under Ministry of Rural Development) to identify and automate multiple services such as integration of textual and spatial land records, integration of registration and mutation processes, automatic updating of land records providing conclusive title to land owners, and etc. mentioning a few [1] [8]. NeGP has identified Panchayat as one of the Mission Mode Projects(MMP) to overcome the challenges at the village level. One of the main objectives of this MMP is to make land records tamper-proof, which will reduce the menace of litigation and social conflicts, associated with land disputes [7].

Blockchain is one of the latest technology that could be used to create tamper proof records. Blockchain, as the name suggests, is a chain of blocks. Blockchain is a distributed ledger, where records are stored in form of transactions. Each node have it's own copy of the ledger, and hence it is immune from single point of failure. Moreover transactions are bunched in form of a block, and each block is linked to previous block by the hash of the previous block, thus forming chain of blocks. Since it is linked with hashes, Blockchain is append only ledger, and any attempt to tamper a confirmed block, is same as attempting to break hash functions (SHA256 in case of bitcoin). This is the source of integrity of these chain. Fig 2 depicts a basic blockchain.

Blockchain have been classified as permissioned and permissionless blockchains. Permissionless blockchain
allows any user to join and leave the network at his will. Identities in case of permissionless blockchain need not to be known, due to which they are prone to sybil attacks [9]. Hence, in order to reach consensus, these blockchain uses methods like proof of work, proof of stake proof of space, etc. Few of the best known permissionless blockchain are Bitcoin, Ethereum, etc.

Permissioned blockchain, on other hand, requires identities of the nodes to be known beforehand. Only certified nodes can join the network. This requires establishment of central authority that generates PKI certificates. Since, identities are well established, these permissioned blockchain are not prone to Sybil attacks, and hence they could use lighter method for consensus like, solo, kafka, Byzantine consensus [10]. One of the best known permissioned blockchain is Hyperledger. Blockchain, apart from being tamper proof ledger, can also have smart contracts or chaincode. These are the programs stored on the blockchain, that are triggered by certain action. Both Hyperledger and Ethereum supports smart contracts.

An open blockchain network cannot possibly serve our all needs. Maintenance of “Record of Rights” is an intricate task and only the trusted members should be able to interact with them. To keep a check on who is able to interact with the ledger it is essential to conceptualize using permissioned blockchains. Out of several permissioned blockchain frameworks like composer, sawtooth, Iroha, fabric, etc. We chose Hyperledger Fabric v1.0 for our prototype because we needed modularity and extensibility in code along with the ability to create separate channels, thus allowing us to group respective departments in order to create a separate ledger of transactions. As far as scalability aspects are concerned. The Fabric architecture uses Solo, Kafka consensus protocols and membership services for plug-and-play. The hyperledger fabric provides both modified and unmodified PKCS11 for key generation.

The components of the architecture are listed below:-

1) A distributed ledger that logs all transactions between the clients and users.
2) A database (encrypted) that maintains the records of all the clients, including land contracts.
3) All the three types of nodes exist in the blockchain network: A set \( \{E\} \) of endorsers that verify a transaction, set \( \{O\} \) of orderers that run a consensus algorithm to create a block and a set \( \{V\} \) of validators that validate and store the blockchain.

III. Proposed Framework

As described above, Land records in India is a generic term for records of right, Khatuuni, Khasra, Shajra, and other records. Since Land is a state subject, Land records are managed by the government of the state. For the purpose of this paper, we are using land records system as in the state of Uttar Pradesh. There are different ways for managing rural and urban land records in Uttar Pradesh. In this paper, we would use our design to tackle the rural land record problem.

In the state of UP, land records management takes place at the office of Tehsildar. This office deals with the land registration and mutation. In this paper, we would like to keep the integrity of both registration and mutation proceedings.

In UP, one of the most important document pertaining to rural land is Khasra. Each Khasra uniquely identifies a piece of land. Each Khasra contain the names of the owner of that land. Notice that since land could be divided into a smaller and smaller part, each Khasra, usually, include names of multiple owners, with their share of ownership in that Khasra. Any mutation in the ownership of land could be either transfer of land to single owner, or transfer of a part of the land to a sole owner, or transfer of different part of the land to a different owner. Notice that each such transfer is done via registration. We would like to maintain the integrity of this registration as well as the integrity of Khasra, the land records. At present, while the registration is a valid land record, there is no verification done while doing registration. All verification is done during mutation proceeding. In this paper, we do checks at the registration stage also. We generate a token, of the registration that could be later used to verify the sanctity of the original document. We propose a model based on hyperledger fabric which consists of 2 parallel blockchains to solve the issue of land registration in India. We call them Khasra chain and registration chain, for dealing with mutation and registration respectively. In our model, clients are authenticated, and a certificate authority distributes respective cryptography material generated by cryptogen and configtxgen. The Khasra chain blockchain contains all the essential information regarding that Khasra like Khasra No, CircleRate, Area, List of Owners, Their Document Hashes, etc. The Registration chain consists of Merkle root made from the hashes of the registered document. This Merkle tree is generated based on the registration done in a span of 1 minute. Based on this Merkle root, we generate tokens that could be used for the verification of the registered document later. This verification plays a major role in maintaining the integrity of the land records, and we describe this later in the paper.

A. Chaincode

Chaincode is the driving logic of the entire system and in this context, We used Chaincode to make sure that the land record mutation is genuine. The Chaincode on Khasra Blockchain does mutation in the Khasra, but that is entangled with the registration blockchain also. Any such mutation needs a valid registration. Here we would like to emphasize that registration happens via payment of stamp duties to state government. Although we have not implemented this feature in our prototype but this would act as an economical deterrent to frivolous land record mutation. Since both the blockchain are entangled, Any mutation also generates a new token that could be used for the verification of newly registered document. This entanglement is not one way. This freshly generated token is also stored as an attribute in the Khasra Blockchain. What this means that for a token to be valid, it should not only be present in the Registration chain, but also in Khasra chain, and hence
stale token cannot be used, thus preventing a double-spending attack.

We have the following chaincodes in our prototype

1) Revenue Calculation: Revenue calculation is a complex process which depends upon the type of land, circle rate, and its area. So currently our model returns area and circle rate as the outputs for the revenue function and further work is required to improve this feature.

Algorithm 1: Revenue Calculation

1 Function Revenue(KhasraNo):
2 if Khasra exists on ledger then
3   return Circle Rate, Area
4 else
5   return "Khasra Does not exist."

2) Updating Details: This is used to update basic values of a khasra like circle rate, area. Over time administration as well as land details change and as a result several values need to be updated. They may also undergo chakbandi (Consolidation).

Algorithm 2: Details Updation

1 Function Update(KhasraNo, Chakbandi(Y/N), CircleRate, Area):
2 if Khasra exists on ledger then
3   if Chakbandi == 'N' then
4     KhasraNo.CircleRate = CircleRate
5     KhasraNo.Area = Area
6   if Chakbandi == 'Y' then
7     return "Chakbandi in progress, Khasra cannot be updated."
8 else
9   return "Khasra Does not exist."

3) Querying the Database: This function takes key value as an argument and returns the current state of that Khasra.

Algorithm 3: Query for a specific object in the database based on the key value.

1 query();
2 Input : Khasra ID
3 Output: Return the details of the Khasra given by the Khasra ID.
4 1) Search for the key : “KhasraID” in the database.
5 2) Retrieve the JSONByte object and return it as the result of the query. If it does not exist, then return error.

4) Land Transactions: This is used to transfer the land from one person to another or several other persons. In our case, transactions refer to the transfer of ownership of a piece of land to a different owner(s). If a person has multiple lands in the same Khasra the document hashes (SHA-256) are appended at the end. In the pseudocode “Reciever” refers to the list of persons along with their hashes which are receiving the land.

Algorithm 4: Transaction

1 Function Transact(KhasraNo, Owner, Hash, Reciever):
2 if Khasra exists on ledger then
3   if Owner in KhasraNo.Owners then
4     if Hash in KhasraNo[Owner][Hash] then
5       Remove Hash
6     Append Reciever.Hash to Reciever(s) in ledger
7     Generate new tokens
8   else
9     Raise "Invalid Hash"
10 else
11   Raise "Invalid Owner"
12 else
13   return Raise "Khasra Does not exist."

B. Viewing the Database

This is used to get all the changes which happened for a Khasra.

Algorithm 5: View all objects in the database corresponding to a partial composite key.

1 View();
2 Input : id of owner which has registered the Khasra ID
3 Output: All the details of the Owner registered by the given proxy peer and have purchased the Khasra given by Khasra ID.
4 1) Generate a partial composite key with Owner id and Khasra ID.
5 2) Create an iterator based on the partial composite key and use it to iterate over the database.
6 3) Retrieve the JSONByte object corresponding to each Owner registered by the proxy peer and who have purchased the land given by the ID value.
7 4) Convert the byte object into string and return the entire set as the result of the query.

Other functions like adding new owners, initializing the ledger, etc are assumed to be common knowledge.

C. Security analysis of the proposed system

Our architecture brings integrity to the land records in India. We want to emphasize that our security is multifold and far better than the present bookkeeping.

a) Tokens: Our tokens are generated from the documents registered all over the state in the span of 1 minute. We calculate the hash of the uploaded documents and then generate the Merkle tree. Our token constitutes the Merkle root, hashes of the sibling nodes in the path from leaves to the Merkle root. We here claim that the uniqueness of the token is as good as the Hash collision of SHA256. Since hash inversion is computationally hard, security of the token is as strong as SHA256. Also, since each generation of the token is linked with multiple updates in Khasra chain, security
is better than in the case of single Blockchain. Since we have used tokens for the online verification of the document, we claim that this solves the problem of document forging in case of land records.

b) Security of Records: Since each Khasra is now represented in the Blockchain, and could only be changed by Chaincode. We can be sure that unauthorized mutation of a record is not possible. Even in the case of a malicious node, we want to emphasize that each variation is linked with the generation of a new token, and each token is only generated after the payment of stamp duties, which also bring security to the records. Since authorized nodes for the generation of tokens and mutation of records are different, there is no single point of failure in the system.

D. Workflow of the GUI Components

The first thing that happens while making one transaction is registering our admin user with our network’s CA. If successful, the CA will send enrollment certificates that the SDK will store for us in our local file system. When the admin wishes to make a transaction from the user interface, the SDK will create an invocation transaction. The operations (AddPeople, Updation, Revenue) get built as a proposal to invoke the Chaincode function. Transactions (via the SDK) will send this proposal to a peer for endorsement. The peer will simulate the transaction by running the Go function and record any changes it attempted to write to the ledger. If the function returns successfully the peer will endorse the proposal and send it back to the Client. Errors will also be sent back, but the proposal will not be endorsed. Client (via the SDK), will then send the endorsed proposal to the orderer. The Orderer will organize a sequence of proposals from the whole network. It will check the sequence of transactions is valid by looking for transactions that conflict with each other. Any transactions that cannot be added to the block because of conflicts will be marked as errors. The orderer will broadcast the new block to the peers of the network. Our peer will receive the new block and validate it by looking at various signatures and hashes. It is then finally committed to the peer’s ledger. At this point, the new transaction exists in our ledger and should soon exist as a record in all peer’s ledgers.

IV. USER SCALABILITY EXPERIMENTS FOR ROBUST ENVIRONMENTS

We have successfully deployed our application with a working GUI on multiple peers by using physical machines which were assigned different roles in the network.

Discrete roles and their permissions:

1) Orderer: Ordering service provides a shared communication channel to clients and peer which offers an atomic-broadcast service for transactions contained in messages. In short it implements a
delivery guarantee and maintains consistency of the ledger state.

2) Certificate Authority: Certificate Authority provides digitally signed certificates to nodes according to their roles and provide them with a digital identity.

3) Tehsildar: Tehsildar are the ones which are responsible for directly interacting with the ledger and randomly endorsing the transactions submitted by the other tehsildars.

4) Test User: Users refer to common public and can view the ledger using a state database. They can request for new tokens to the tehsildars.

In this implementation, we have not done optimization in terms of space, which is one of the major factors in terms of the scaling of blockchains. We hope that future work would also consider this aspect. We hope that methods for geospatial sharding could be easily used, as here land data is segregated along the Tehsils.

**REFERENCES**


---

**Figure 5. Scalability Test Network**

A test environment was created and real world transactions were simulated in the network.

1) Machine 1: Orderer + Certificate Authority
2) Machine 2: Tehsildar 1
3) Machine 3: Tehsildar 2
4) Machine 4: Tehsildar 3
5) Machine 5: Test User

The stakeholders pertaining to that of our model could be defined as follows:

1) Validators: They are responsible for validating the transactions and are a subset of tehsildars.
2) Orderers: They run the consensus algorithm and are responsible for block formation.
3) Endorsers: Endorsers follow the endorsement policies and verify the transactions.

V. CONCLUSION AND FUTURE SCOPE

The blockchain is currently in an evolving stage, as far as use cases like Land Registrations are concerned since the deployment levels have been increasing gradually. It is going to revolutionize the current system by eradicating numerous flaws at the same time. We have presented an Architecture in which entangled chain could be used to secure land records. In this model, we have only considered Registration Document and Khasra, and we hope that the other documents could be linked in similar ways thus create a mesh, which would be better in terms of security.
Landcoin - A Land Management Protocol

Anasuya Acharya
Information Security Research and Development Center
Department of Computer Science and Engineering
Indian Institute of Technology Bombay, Mumbai 400076, India
Email: anasuyahirai@gmail.com

Abstract—Public blockchains provide integrity, transparency, immutability, and inclusion-guarantee to the transactions they process and record in their chain of blocks. Bitcoin, Litecoin are a few examples of time-tested, reliable public blockchains that handle only one type of transaction i.e., transfer-of-value (specifically money/currency) from a user to another. Whereas, general purpose blockchains like Ethereum provide means and methods to encapsulate transfer of not only a value but any asset class that can be digitally represented. However, due to its high-level programming language to write asset transfer rules. Ethereum has encountered instances where it had to forgo the claim of immutability (e.g., the DAO attack). On contrary, the purpose-built, script-based blockchains like Litecoin have withstood the test of time and therefore are seen as reliable – an important aspect in attracting users to use a system that is built using such proven, reliable protocols. In this paper, we present an asset-transfer system for land management that borrows from Litecoin protocol its script, underlying consensus and block structure. Our resultant system is a permissioned blockchain, where only a set of pre-approved miners can append land records to the blockchain. We integrate sidechains to store transaction details that are private to the transacting parties, thus providing conditional privacy to the transactions. The mainchain stores land records that can be queried by the public, whereas sidechains are integrated to store the intricate details about the intermediate validations/operations performed by regulators, registrars, and notaries.

Index Terms—blockchain, Litecoin, land management, privacy.

I. INTRODUCTION

In countries like India, where the land records are maintained with human intervention, they are perennially marred with presumption and excessive bureaucracy – leading to malfeasance and thus into time-consuming legal disputes [9]. Land is a precious asset that can be used as a collateral or used for many other productive purposes only if the title deeds are indisputable and are derived using a transparent process. Land litigation is a huge cost to the economy and one of the reasons of financial exclusion for a large population. In the past decade, many of the states have migrated their paper based records to digital versions [1] but with only partial success in containing the malfeasance. Several state governments (since land ownership is state’s purview in the federal system of India) are exploring the blockchain approach to inherit its natural properties – integrity, transparency, and immutability; to the land records and their transactions in a land management system. Most of the implementations are based on Ethereum smart contracts and Hyperledger Fabric; where the steps of land management process are encapsulated as smart contracts (high-level programs supported by the underlying blockchain), which inherit the pros and cons of any high-level programming language [3]. Another forthcoming limitation of these implementations is their system’s state-specific scope, which may not answer queries like; what all title deeds a subject holds in India, across the states. In other words, the current approaches that we are aware of are not scalable across the states and also are not interoperable.

In this paper, we present a land management system based on Litecoin’s public blockchain codebase. We add features to modify the same to suit our purpose of having a permissioned blockchain for transfer of asset while keeping the underlying skeleton of the blockchain and its stack-based purpose specific script unchanged for the most part. Our system is scalable, interoperable, and also privacy-preserving. We achieve these desirable features by segregating the data related to a land transaction into two categories: public and private. We record the public part of the land transaction data (like, who transferred a land to whom) on mainchain and the corresponding private part of the land transaction (like, at what rate the land is sold) on a sidechain that is maintained by the state to which that land belongs. The mainchain can be queried by anyone, whereas the sidechains accept attribute-based queries only from the parties that are involved in a land transaction. In our implementation, we have introduced new transaction types to Litecoin’s codebase. Each type corresponds to a distinct operation in prevalent land management workflow. We introduce separate transaction types to the mainchain and sidechains. Each land transfer transaction on the mainchain traverses through the workflow on the sidechain before getting committed on mainchain. The first transaction/step on the sidechain takes an input from the mainchain and the last transaction/step on the sidechain inputs to the mainchain. In other words, when a subject intends to transfer her land to another subject, the transaction has to go through government verification/diligence process (which gets recorded on the sidechain) before being accepted by the miners on the mainchain. Our construction allows an audit trail to traverse from mainchain to sidechain and back to the mainchain without the auditor knowing the intricate details recorded on the sidechain, which only the seller, buyer and the land authorities can decrypt. Maintaining confidentiality of transaction details while keeping the transaction trail transparent is an important feature that our system provides.
II. BACKGROUND & MOTIVATION

The prevalent land management systems use databases for storage of land records and use cryptography for data protection. While confidentiality mechanisms can be enforced, these systems fall short in maintaining an immutable trail of operations performed on land records, because tuples in a database can be overwritten. Double-entry book-keeping is used in identifying discrepancies in records, however malicious records can only be traced with the help of a digital log management system, which in turn is susceptible to tampering [11]. Triple-entry book-keeping can be adopted, where each transaction is digitally signed by the subject of transaction; irrespective of the transaction being valid, erroneous or malicious – thus fixing accountability of actions. Most of the non-blockchain implementations of land management systems fall under this triple-entry book-keeping category. However, such centralized systems lack a real-time, transparent view of transactions in the system.

A new type of decentralized database technology (aka DLT/blockchain) appears to be a natural fit for land management system because it not only provides all of the properties of a triple-entry book-keeping approach but also offers immutability, transparency, and real-time auditability to land transactions. The transparency and auditability for all the stages in the transfer of land coupled with the inherent immutability guarantee that a blockchain provides helps solve the double-spend and prevent similar frauds prevalent during transfer of property agreements. Furthermore, blocks chained with cryptographic hashes provide a verifiable record of all the history of the transfer of land assets, as opposed to a simple database where only the current ownership status is reflected. Databases can also store transaction history but there is no guarantee that the records have not been tampered since it was appended to the logs and the trust needs to be placed onto the authorities maintaining the database for that as well.

The choice of a blockchain protocol for land management is an important design criteria because the inherent pros and cons of the protocol reflect into the system. A judicious mix of engineering tweaks need to be adopted in order to inherit the pros and mitigate the cons. In our approach, we narrowed down on the Litecoin protocol [8] due to the following criteria: i) time-tested, proven, open protocol; ii) limited set of stack-based script operations; iii) scrypt based proof-of-work consensus algorithm.

A general purpose blockchain like Ethereum could be used but its underlying programming language may open up avenues [3], [7] for serious asset loss or inconsistent asset ownership states, which is unacceptable. Therefore, we believe that it is prudent to extend a protocol (like Bitcoin [10] or Litecoin [8]) that is built for a specific purpose rather than using a general purpose protocol like Ethereum [12] or Hyperledger Fabric [6]. Between the Bitcoin and the Litecoin codebases, we opted for Litecoin for its reliance on scrypt hashing algorithm that cannot be accelerated by ASIC processors, which is the case for Bitcoin since it uses SHA256 hashing algorithm for block mining. To make the system adaptable for current land management practices, we had to tweak the default setup of Litecoin blockchain protocol. In summary, the following are the modifications we introduced in the Litecoin protocol, for which we present details in the next Section.

1) New transaction types: for mainchain and sidechain, each corresponding to a step in the current workflow of land management
2) Pre-approved miners: is a set of public-keys listed by the central government through a transaction on mainchain
3) Sidechains: allow states to compose their respective land management workflows as separate chain anchored in the mainchain; there are no coinbase operations on sidechains
4) Certified coins and mapping of coins to the landmass: is a one time operation during the bootstrapping phase in which all the pre-mined 84 million coins are transferred to the central government and then distributed to the state governments according to their proportionate landmasses.
5) Certified user addresses and signing keys: any user entity entering the system needs to have an address and corresponding signing and verification keys. Instead of generating them locally as in the case for permissionless cryptocurrencies, a Trusted Third Party (TTP) is involved in verifying identities and assigning certificates with the keys being formed jointly by both parties.

The protocol in [2] is still pseudonymous and the TTP has no control over the transactions being signed, but this helps keep track of the mapping of users in the systems against their existence as real-world entities.

III. ARCHITECTURE OF LANDCOIN PROTOCOL

This section is a guide through all the design aspects taken into consideration while creating the Landcoin Protocol.

A. From a transfer-of-value to a transfer-of-asset system

The following two challenges come up while extending a transfer-of-value system (Litecoin) to a transfer-of-asset system (Landcoin): i) a class of asset like money, which is represented by numbers alone, is different from the class of assets like land, which has identification attributes and does not have properties like fungibility or malleability; and ii) assets like land have certain legal requirements to be adhered to before any mutation or transfer occurs. These characteristics need to be taken into consideration while constructing the transfer-of-assets system.

We make the following four assumptions in our design: i) during the initialization phase of our system, all the 84 million coins are mapped to units of total land; ii) all stakeholders have unique identifiers – UIDs; iii) The title deeds of land are unambiguous and the assets represented therein have unique identifiers – URIs; and iv) the pre-approved miners (i.e., transaction validators) are honest and are always available.

B. From permissionless to permissioned setup

Since land ownership is a legal statute backed by the state, the state acts as an arbitrator for all land transactions. To
incorporate this requirement, we need to restrict the trans-
action miners who are authorized to commit transactions to
the blockchain. We introduce a role called the “Government
Authority” that is allowed to manage the chains by con-
trolling the body of miners. We introduce a special type of
transaction called “Governance Transaction” to the mainchain
and sidechain where the “Government Authority” can add or
remove a public-key from the list of pre-approved miners for
their respective chains.

Restricting the nodes who can mine the transactions does
away with the concept of incentives for transaction mining.
That is, there won’t be any transaction fees either on mainchain
or sidechain. The coinbase type of transactions that come with
the Litecoin protocol are frozen after the completion of Phase 1
in bootstrapping of Landcoin protocol as shown in Figure 1.

Furthermore, the publicly-queriable mainchain only
includes transactions that are signed off by the zonal
Registrars. This makes allows for efficient queries from data
across states, while making sure that the power remains
with the state. Furthermore, this serves as an abstraction
for any intermediate steps involved that are encapsulated as
confidential transactions on the sidechains.

In order to build a practical system, the design of the
proposed system has to resemble closely the distinct stages in
the prevalent land management practices. In the following we
enlist the steps involved in a typical land transfer transaction.

C. Steps & requirements of the land transaction workflow

While there exist several types of land-transfer/mutation
transactions: “Sale Deed”, “Gift Deed”, “Relinquishment
Deed”, “Partition/Settlement Deed” and “Inheritance/Will
Deed”; in this work we explore the “Sale Deed” as a typical
use-case. The protocol can later be extended to accommodate
the other deeds and their work-flows as well.

The “Sale Deed” is the main document by which a seller
transfers his right on the property to the purchaser, who then
acquires absolute ownership of the property. The process of
“Sale Deed” execution, in Indian context, requires involvement
of sellers, buyers, witnesses, land officers, and land registrar
at gradual stages. In the following we enumerate the entities,
their roles, and the steps in the prevalent land management
workflow. These steps are indicative only and they may vary.

1) A seller willing to sell property needs to raise an intent
to the Zonal Land Office (ZLO).

2) The ZLO processes this intent through legal checks and
verifies the eligibility of buyer/s.

3) If the intent is allowed to go through, the ZLO declares
a minimum Market Value (MV) for the piece of the land.

4) Upon mutual identity verification, both the parties may
negotiate an agreeable price, which is equal to or higher
than the MV.

5) Then seller prepares a Transfer of Ownership document
with particulars of the buyer, the land, the price, and two
witnesses who need to sign the document.

6) Revenue tax is calculated for the property and the invoice
is presented to the buyer in order to proceed with the land
transfer.

7) An invoice is prepared with the details of the parties
involved in the transaction, along with a list of conditions
that need to be honored.

8) The final invoice, along with two witnesses, is jointly
presented to the Registrar for approval of intended land
transfer leading to a valid “Sale Deed” execution.

9) As a final step the “Sale Deed” is said to be executed by
making the payment of full amount specified in the deed.
This payment amount constitutes the revenue tax.

Taking into consideration the above indicative workflow for transactions in land management, it is amply evident that several stakeholders carry out intermediate transactions leading to the actual land transaction. Therefore, it is not straightforward to use the transfer-of-value type of transaction available under Litecoin protocol. Hence, we need to introduce new transaction types to accommodate representation of intermediate transactions by respective stakeholders.

D. MAINCHAIN: Parameters and construction

We have modified the Litecoin codebase with the following parameters so that we resemble closely with the prevalent land management system and its workflow.

- Total coinbase $\rightarrow$ total landmass in India ($km^2$)
- Divisibility: 8 decimal places (smallest unit is $dm^2$)
- Block generation time on mainchain: approx. 1 hour
- Number of block confirmations: 40 (approx. 1.5 days)
- Consensus: proof-of-work (by pre-approved miners)

We start mining with lower difficulty levels and do not open the blockchain for public participation, until all blocks are mined by the appointed miners. We call this chain as Landcoin’s mainchain. Before initializing it for land transfer, all the miners send their coinbase to the “Government Authority”, represented by a self-certified public key. The “Government Authority” then transfers proportionate amounts of Landcoins to individual state Registrars, represented by addresses that are certified by the “Government Authority”. The Registrars map the coins to unique URIs of the pieces of land in their respective jurisdictions. The mapping is a transfer operation on the MAINCHAIN to the individual owners of respective URIs. The “Government Authority” invokes “Governance Transaction” (detailed in Section ??) to authorize a list of public-keys as pre-approved miners. Upon completion of the bootstrapping phase, the MAINCHAIN starts accepting asset-transfer requests.

An asset-transfer transaction, floated by a user on the MAINCHAIN, is accepted by the miners only when it has a signature of the Registrar of the sidechain for the corresponding zone. The attestation process requires the intent transaction to go through pre-defined set of steps as deemed suitable by respective states. Each transaction type on sidechain corresponds to a distinct step in the prevalent process.

E. SIDECHAIN: Placeholder for private information

Though the confirmed transactions on MAINCHAIN show the current ownership of a piece of land and its transactional history, the details about each necessary clearance, attestation, price, et al. are hidden from the public. Only the pre-approved public-keys (Registrar, land officers from ZLO) and the parties to the transaction can decrypt the content of the transactions on the sidechain. This provides the property of conditional confidentiality to the content of transactions on the sidechain. We use a CPABE scheme [5] to achieve this property, whose setup and primitive operations are depicted in Figure 2.

The setup for CPABE starts with a globally public encryption key (EK) and a master secret key (MSK) that is private to the “Government Authority”. For each user of the system, depending upon their roles, certain ‘attributes’ are defined for which each user has an assigned ‘value’. The MSK is used to derive decryption keys for each user according to their ‘attribute:value’ pairs. For encrypting a message under this scheme, EK is used along with an encoding of the ‘Policy Tree’, which is a propositional logic statement with ‘attribute:value’ pairs as their atomic parts (leaves of the tree), always evaluating to either true or false. The encryption algorithm encodes this policy into the resultant ciphertext. During decryption, the ciphertext and the user’s key is input to the algorithm and the plaintext is output only if the policy evaluates to true on the user’s attribute values.

Encryption scheme in [5] is based on bilinear pairings and its implementation is publicly available as a library and documented in [4].

The MAINCHAIN stores all the confirmed Transfer of Ownership transactions, and “Governance Transactions”; whereas, the SIDECHAINS (one for each ZLO) store the transactions involved in the intermediate steps and run verification scripts for legal compliance. The scripts for compliance check obey the governance policies and are special purpose – specific to the land resource management of a state.

F. Protocol Stakeholders, their Roles, and Transaction Types

![Fig. 3. Key management among Stakeholders](image-url)
Stakeholders and their roles:

- **Clients (buyers, sellers, witnesses):** Buyers and sellers are the end users of the system. Witnesses provide consent to a specific transaction by digitally signing it.
  - Certified Address mapped to Identification info.
  - Mainchain Visibility - all information
  - Sidechain Visibility - all transactions in which they are a buyer, seller, or witness
  - Signing key to use for certain sidechain transactions as seller or witness

- **Zonal Land Officers:** Verification, approval of transaction initiated by sellers and mining of transactions on SIDECHAIN.
  - Mainchain Visibility - all information
  - Sidechain Visibility - all sidechain info. of that office
  - Signing key to use for certain sidechain transactions, and all blocks for the sidechain

- **Registrar:** responsible for final approval of change of ownership requests.
  - Certificate Authority defining Land Officer set
  - Mainchain Visibility - all information
  - Sidechain Visibility - all sidechain info. of that office
  - Signing key to use for certain sidechain transactions, and mainchain change-of-ownership transactions

- **Government Authority:** maintenance of the system through bootstrapping and “Governance Transactions”.
  - Certificate Authority defining mainchain miners set
  - Trusted Party assigning all clients certified addresses
  - Trusted Party assigning all entities CPABE keys
  - Mainchain Visibility - all information
  - Sidechain Visibility - all transactions in all sidechains
  - Signing key to use for governance transactions

- **Pre-approved Miners:** accept and mine the “Change of Ownership” transactions emanating from SIDECHAIND.
  - Mainchain Visibility - all information
  - Verify all mainchain transactions
  - Signing key to use for blocks on the mainchain

Key management and certified addresses:
The CPABE master key MSK lies with the “Government Authority” and the encryption key EK is globally known and stored in the Software Interfaces of the Clients, ZLOs, and Registrars. Each of their decryption keys are also derived from this, as shown in Figure 3.

The signing keys and addresses are certified by a TTP that is again, the “Government Authority”. These certified addresses are created as in [2] that follows a Diffie-Hellman-like exchange between the client and TTP to generate a shared randomness used to create the Signing Key and certificate. The verification key and address is then derived from it. The certificate is also verified during the signature verification to ensure the key was certified. This computation happens in such a way that the TTP has no knowledge of the final key and so cannot abuse the signing authority of the client.

Transaction types and their composition:

1) **Change of ownership transaction:** $T_M$
   - $H_M$ - Transaction Header
   - $H_{M'}$ - Header of Source transaction on MAINCHAIN
   - $\{A_S\}$ - Certified Addresses of all sellers
   - $\{A_B\}$ - Certified Addresses of all buyers
   - $\{URI\}$ - Survey numbers and GIS co-ordinates
   - $DT$ - Effective-on date
   - $S_R$ - Signature of the Registrar

2) **Governance transaction:** $T_G$
   - $H_G$ - Transaction Header
   - $\{A, O, O_I\{\ldots\}\}$ -
     - $A$ - Action (add/delete)
     - $O$ - Object (certificate, miner, zone, ttype)
     - $O_I\{\ldots\}$ - set of objects
   - $EJ$ - Effective-on jurisdiction
   - $DT$ - Effective-on date
   - $S_G$ - Signature of Government Authority

3) **Booking transaction:** $T_B$ - Declaration by seller expressing desire to sell land to particular prospective buyer
   - $H_B$ - Transaction Header
   - $H_{M'}$ - Header of Source transaction on MAINCHAIN
   - $\{A_S\}$ - Certified Addresses of all sellers
   - $\{A_B\}$ - Certified Addresses of all buyers
   - $\{URI\}$ - Survey numbers and GIS co-ordinates
   - $\{S_S\}$ - Signatures of sellers

4) **Rejection:** $T_R$ - Abort of process “Change of Ownership”
   - $H_R$ - Transaction Header
   - $H_{M'}$ - Header of Source transaction on MAINCHAIN
   - $RR$ - Reason for rejection (optional)
   - $S_R$ - Signature of the Registrar

5) **Clearance:** $T_C$ - Permission to sell at/above declared MV
   - $H_C$ - Transaction Header
   - $H_{B'}$ - Transaction Header of $T_B$
   - $MV$ - Minimum market value evaluated by ZLO
   - $S_L$ - Signature of ZLO

6) **Pre-handover document:** $T_D$ - Document declaring final selling price decided upon and identities of witnesses signing off on the handover
   - $H_D$ - Transaction Header
   - $H_{C'}$ - Transaction Header of $T_C$
   - $\{A_S\}$ - Certified Addresses of all sellers
   - $\{A_B\}$ - Certified Addresses of all buyers
   - $\{A_W\}$ - Certified Addresses of all witnesses
   - $\{URI\}$ - Survey numbers and GIS co-ordinates
   - $SP$ - Final selling price
   - $\{S_S\}$ - Signatures of sellers

7) **Rejection of Pre-handover document:** $T_{DR}$
   - $H_{DR}$ - Transaction Header
   - $H_{D'}$ - Header of Source $T_D$
   - $RR$ - Reason for rejection (optional)
   - $S_L$ - Signature of ZLO
8) Document verification: $T_V$ - Acknowledgement after verification of legal documents by ZLO

- $H_V$ - Transaction Header
- $H_D'$ - Header of $T_D$
- $\{D_S\}$ - Hashes for doc. clearance of all sellers
- $\{D_B\}$ - Hashes for doc. clearance of all buyers
- $\{D_W\}$ - Hashes for doc. clearance of all witnesses
- $S_L$ - Signature of ZLO

9) Tax Receipt: $T_T$

- $H_T$ - Transaction Header
- $H_V'$ - Header of $T_V$
- $\{URI\}$ - Survey numbers and GIS co-ordinates
- $SP$ - Final selling price
- $Tax$ - Tax amount payable
- $\{D_T\}$ - Hashes for doc. proof of tax payment
- $S_R$ - Signature of the Registrar

10) Completion Receipt: $T_F$

- $H_F$ - Transaction Header
- $H_T'$ - Transaction Header of $T_T$
- $\{A_S\}$ - Certified Addresses of all sellers
- $\{A_B\}$ - Certified Addresses of all buyers
- $\{A_W\}$ - Certified Addresses of all witnesses
- $\{URI\}$ - Survey numbers and GIS co-ordinates
- $SP$ - Final selling price
- $\{S_S\}$ - Signatures of sellers
- $\{S_W\}$ - Signatures of witnesses
- $S_R$ - Signature of the Registrar

IV. SUMMARY OF GUARANTEES

This system aims to provide the following guarantees:

1) Authenticated yet Pseudonymous Clients: Certified Addresses provided by the Government Authority is conditioned on verification of the identity of the client as an actual and unique real-world entity. This is a one-time process and helps keep track of the user base of a system with legal implications such as this one. The addresses do not reflect user IDs on chain and so the protocol henceforth is a pseudonymous one with only the certificates being verified on chain.

2) Verify-able history of Land Transactions: MAINCHAIN transactions all originate from Government Authority address and so all the existing land is accounted for. Furthermore, all valid change-of-ownership transactions are signed by the Registrar (having gone through the entire SIDECHAIN workflow) before being put on the MAINCHAIN. This consolidated transaction trail provided by the blockchain, along with the queryable booking transactions, mitigates double spend frauds.

3) Confidentiality of Intermediate Steps: All intermediate step transactions for change of ownership are recorded on the SIDECHAIN. This maintains a consolidated record of all the actions taking place and, at the same time, keeps unnecessary details outside of the MAINCHAIN. As these transactions use confidential information like identities for verification and such, they are encrypted under CPABE for proper access control to be maintained.

4) Honesty among Mining Authorities: All mining authorities are regulated: added and evicted, if needed, by the...
Algorithm 1 Landcoin Protocol

1: Seller initiates $T_B$ on SIDECHAIN
2: \( \triangleright \) refers to previous $T_M$ that acts as an anchor
3: if Legal conditions of exchange are not met then
4: Registrar puts $T_R$ return
5: else
6: $ZLO$ puts $T_V$ with mention of $MV$
7: Seller puts $T_D$ with final price and Witnesses
8: while $ZLO$ puts $T_{DR}$ do
9: if Still interested then
10: Seller re-does $T_D$
11: else
12: $T_R$ return
13: end if
14: end while
15: while Document Verification not approved by $ZLO$ do
16: if Still interested then
17: if Valid documents can be produced then
18: $ZLO$ puts $T_V$
19: else
20: Seller re-does $T_D$
21: GOTO line 9
22: end if
23: else
24: $T_R$ return
25: end if
26: end while
27: $T_T$ is put on the SIDECHAIN
28: $T_F$ is put on the SIDECHAIN
29: $T_M$ is put on the MAINCHAIN return
30: end if

Government Authority and therefore one can expect minimal malicious activity. While competition still exists among them for any off-chain compensation the Government Authority may provide malicious miners can have their authority revoked. Therefore, minimal, if any, forking can be expected; but in the event of one, it may take as much as over a day to resolve due to the slow growth of the chain. Nevertheless, as any transaction put on the chain needs to come signed by the Registrar, a fork will not translate to a double-spend event.

5) Centralized Authority with Decentralized Auditability:
The protocol allows for as less deviation form the existing structure of authority for land record management as possible. However, it facilitates larger inclusion of both users and land charted; and ease of auditability and verification by virtue of the guarantees provided by the blockchain structure and its underlying cryptographic basis.

V. IMPLEMENTATION DETAILS

As a simplified proof-of-concept, the following is an implementation of the protocol. This implementation supports four roles: Client (Individual or Organization), Registrar (with operations of $ZLO$), Government Authority, and Miners.

A Landcoin Client has a client ID that needs to be entered into the Login Window along with a password. If a new Client registers, his/her certified keys and address is then formed by a back-end call to the Trusted Third Party. Given below are the typical client credentials:

A client can view, using the View Menu, the chain explorer of the mainchain, list of ongoing transactions he/she is involved in, list of his/her completed transactions within the system.

![Fig. 5. Login Window and Different User Roles](image5.png)

![Fig. 6. Client Dashboard](image6.png)

![Fig. 7. Client View: Ongoing Transactions of the Client](image7.png)
Furthermore, in this simplified version, creation/initiation of the transaction and all the steps after it is merged into one action that the client performs. This transaction then becomes part of the USERMEMPOOL.

As in the original protocol, the registrar also reserves the role of giving the final approval for any transaction. This transaction then is moved from the USERMEMPOOL to the MEMPOOL from where it can be picked up by a miner to be included in a block on the blockchain.

The Registrar’s Dashboard contains his credentials that are similar to that of the Client’s except it also includes the Zone of jurisdiction.

The remaining two interfaces are of the Miner and the Government Authority. A Miner, once logged in, can view the MEMPOOL, UTXO and the blockchain through the explorer. However, mining is not possible unless a special permission is granted by the Government Authority. The actions that a Miner can perform are requesting for mining permission, and mining (once the permission is granted).

This role has permissions to view all mainchain transactions through the chain explorer, ongoing transactions of the zone, completed transactions of the zone, and current zone state (current owners of land in the zone). This includes transactions in the USERMEMPOOL, the MEMPOOL, and the UTXO that belong to the zone.
The Government Authority is allowed access to the main blockchain through the explorer, and can view the list of current miners and prospective miners that have requested permission for the same. The actions this role performs include granting and revoking of mining permission.

VI. CONCLUSION

We have devised a transfer-of-asset system (Landcoin) from a transfer-of-value system (i.e., Litecoin). Landcoin extends the limited and secure opcode set of Litecoin. This set is Turing incomplete and all the scripts written can be formally verified for correctness, at the same time providing just enough functionality for our use-case. Transaction details are segregated into public and private data streams on separate but linked blockchains, helping us preserve the privacy. Mainchain records each and every successful transfer of ownership transaction, and the sidechain records the intermediate details of such transactions emanating from mainchain. Thus anyone can browse through the mainchain to verify the ownership of land but the intricate details of associated ownership provenance is available only to the stakeholders of that transaction. Confidentiality of transaction details on sidechains is enforced using CPABE (Ciphertext Policy - Attribute Based Encryption) scheme, where the “Government Authority” possesses the master key so it can read (decrypt) any transaction. This is a preliminary version of Landcoin and we are experimenting with different methods to achieve transaction confidentiality and privacy on sidechains.

ACKNOWLEDGMENT:

This work is carried out as part of research at ISRDC (Information Security Research and Development Center), supported through grant 15 DEITY00 004, funded by MeitY, Government of India.

REFERENCES

Energy Banking through Smart Contracts, Crypto-tokens and Block-chain Technology - First step in Shaping up of Full Fledged Electricity Markets in India

S.K. Jain1
Power System Operation Corporation Ltd.1
India1
skjain@posoco.in

Paresh Khandelwal2
Power System Operation Corporation Ltd.2
India2
parekhkhandelwal@posoco.in

P.K. Agarwal3
Power System Operation Corporation Ltd.3
India3
pkegarwal@posoco.in

Abstract: Power sector across the globe is experiencing huge transformation in terms of regulatory, commercial and technological arenas. To match the transformational needs, this paper proposes the ways to utilise block chain technology and smart contracts in energy banking. Energy banking is presently being done only on MW quantum basis with no price tag. However, the electricity price similar to other commodities also depends on demand supply ratio, thus varies in peak and off-peak times. To ensure energy banking transactions at real time cost and hassle-free financial settlements, this paper suggests use of Peer to Peer (P2P) model of block chain technology for executing Smart Contracts mutually agreed by both parties. This paper also recommends an online platform for carrying out traditional energy banking in an efficient and transparent manner. Validation of mutual consent before each transaction, tamper evident transactions and decentralised ledger technology could replace traditional energy banking system.

Keywords: P2P Model, Smart Contracts, block chain, Energy banking, TBCB, DLT, TIU

I. INTRODUCTION

Banking of power or Energy Banking means exchange of electricity for electricity (instead of money). In India, due to geographical and seasonal diversity the power requirement varies heavily between states. Due to this, it’s always a herculean task to devise a balance between demand and generation requirement of a state. It is worthy to mention here that, in India there are two prevalent methods of energy banking, firstly through mutual consent and secondly through competitive bidding. In India, Hydro rich states have executed Energy banking agreement on mutual consent basis through MOU (Memorandum of Understanding). Some utilities in India have adopted the competitive bidding route for Energy banking. The electricity Utility (say, Utility A) has surplus power in winter season with respect to demand, but the same generation capacity couldn’t cater to it’s demand in summer season, however other Utility B has reverse scenario. Both Utilities meet each other’s needs retrospectively for a quantum of power without any financial transactions with each other in their needy season. This is what we call energy banking It’s like Barter system where in commodity transactions happen without currency payment. In energy banking, exchange of same commodity happens, but at different timelines of the year.

1.05.19 - 30.09.19
06:00 - 24:00
Return % to be specified
RTC
Return % to be specified

TABLE-1: SUPPLY OF POWER BY UTILITY A TO UTILITY B

<table>
<thead>
<tr>
<th>Period</th>
<th>Option A</th>
<th>Option B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Duration (Hrs.)</td>
<td>Quantum (MW)</td>
</tr>
<tr>
<td>01.10.18 - 31.12.18</td>
<td>06:00 - 22:00 &amp; 24:00</td>
<td>Up to 200</td>
</tr>
</tbody>
</table>

TABLE-2: RETURN CYCLE OF POWER FROM UTILITY B TO UTILITY A

<table>
<thead>
<tr>
<th>Period</th>
<th>Option A</th>
<th>Option B</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Duration (Hrs.)</td>
<td>Quantum (MW)</td>
</tr>
<tr>
<td>01.05.19 - 30.09.19</td>
<td>06:00 - 24:00 &amp; 12:00</td>
<td>Return % to be specified</td>
</tr>
</tbody>
</table>
As seen from above tables, Energy banking seems easier way of exchange of electricity between two parties as there are no financial transactions involved. However, the actual scenario is entirely different.

Despite having few advantages and ease of execution, Energy banking poses lot many challenges as outlined hereunder:

(i) Since, these agreements are spanned for a longer time frame i.e. one party delivers power in 1st quarter and is liable to receive power in 3rd quarter, there are always chances of fall out of the contract due to political or any other issues which may occur in these time differences. This may incur heavy loss to the party who has supplied power in 1st qtr.

(ii) The power exchanged among the parties occurs at different time of the day and different quarters of the year. Price of electricity is volatile and depends on real time market conditions and corridor congestions. Since the power exchange takes place in different timelines, the cost of power will not be same; hence one of the parties in agreement is at loss which may result in auditing problem at later stages.

(iii) Involvement of many stakeholders viz. respective RLDCs/SLDCs for scheduling, RPCs for energy accounting, traders, stakeholders, etc. Hence, there are chances of failure of Energy banking agreement due to procedural lapses.

(iv) As explained earlier, some parties execute energy banking via tendering process, which itself is time consuming & also involve financial transactions in terms of Earnest Money Deposit (EMD), Bank guarantee (BG), etc. In addition to this, open access charges up-to delivery points i.e. RLDC/SLDC application fees, RLDC/SLDC Operating Charges, PoC (Point of Connection), injection/drawl charges, STU Charges, trading margin, etc. These transactions need to be ensured, monitored and settled. Thus, an entire finance mechanism needs to be put in place and that’s a cumbersome and time taking process.

(v) Once agreement executed, the parties become dependent on each other irrespective of knowing the actual demand, corridor availability, market price, etc. If there is any corridor failure or breach of generation contracts, the second party is left with no other option except to purchase power at market determined prices which obviously will be higher. This inadvertent hike in price of power is not covered in any contracts, only a minimal penalty clause is available. In the above-mentioned contract also, there exists a clause for unsupplied quantum of energy at the end of banking cycle. This clause allows the second party to settle @ Rs. 4/- or 5.25/- per KWh to first party which is again an assumed price.

From the above points, it is clear that Energy banking agreement requires lot of human intervention at various stages. It is a time-consuming process & there is always a possibility of second party defaulting while returning power as there is a gap of approximately 3 months for return cycle. Though the defaulting party is liable to pay penalty on account of default as per contract, but this penalty may not be enough for the first party to arrange the same quantum of power from another source at reasonable price. Same quantum of Electricity has different price tags at different timelines of the year; hence these agreements cannot be executed on price parity mechanism. Since, it involves public money at large, these agreements must follow sound commercial principles.

To mitigate above complexities, crypto token & smart contract based on block chain technology may be used for Energy Banking.

II. USE OF BLOCK-CHAIN TECHNOLOGY IN ENERGY BANKING

The sample distributed network in which all the stakeholders involved in Power banking i.e. RPCs, System operators (RLDCs/SLDCs), Captive power producers, Distribution Licensees, Power Exchanges, traders etc. is as shown in Fig. 2. This proposed distributed network is a permissioned network. The parties which are not registered in the network cannot access any information. The transactions will be in encrypted form and can only be decrypted by the concerned parties only. All transactions will be completed through smart contracts and saved in distributed ledger. Here, transactions do not limit to financial transactions only. It could be any document & data also.

To maintain the secrecy of documents, same will be encrypted first and then put into block chain, so that same can be decoded only by concerned user in network. For financial transactions a crypto currency, named iP6owercoin is proposed to be issued by TIU (Token Issuing Utility). RPC (Regional Power Committee) or any other authority approved by regulatory commission may be designated as TIU. The initial value of iP6owercoins will be equivalent to product of MWH and yearly average MCP (Market
Clearing price) of Energy exchange for last financial year. Proposed formula for one iPowercoin is as below

| Initial Value of One iPowercoin = 1 MWH × Yearly average MCP of Energy exchange for last financial year in Rs. |

Initially, both parties having Energy banking arrangement will be required to deposit the amount equivalent to product of contracted power (in MWH) and last financial year’s average MCP of Energy exchange. If Energy banking contract is for 200 MW, 8 hours and 3 month @ yearly average MCP being Rs. 5/- per KWH, then both parties have to deposit Rs. 200x1000x8x3x5x30 = Rs. 24,00,00,000/- with TIU as initial deposit/guarantee money. TIU will issue 2,40,000 iPowercoins for each party. These iPowercoins will be credited to accounts of both parties, but it can’t be utilized without validation by TIU. Settlement will be done by TIU based on actual energy transacted @ market (Exchange) determined price between both parties on daily basis. However, iPowercoins available with one entity can either be used in single banking agreement or in multiple banking agreements. If regulatory authority permits, then these iPowercoins can be used pan India for energy exchange or settling of any dues pertaining to power market.

III. PROPOSED METHOD OF iPOWERCOIN GENERATION

As shown in Fig. 3 the system-generated iPowercoins are saved in Block-0, known as Genesis Block. The Genesis block is the first block of block-chain generated by system.

The first transaction in block-chain is saved in Block-1 and executed by the smart contract. The block header of block-1 is hash (SHA256 or SHA 512) of all the data available in block 0 as shown by hash pointer. Similarly, the block header of block-2 has the hash of entire block of block-1. The same process is repeated in all subsequent blocks entered in block chain. The entire block-chain is distributed to all the entities connected to network. If any malicious entity wants to change any transaction/data in the block, he is required to change the entire subsequent blocks also as every succeeding block has the previous block hash. Additionally, the malicious entity is required to change these transition/data simultaneously on all the nodes connected in network. This is practically very difficult which makes these transactions tamper evident.

IV. PROPOSED BIDDING PROCESS UTILIZING SMART CONTRACTS

The bidding process can be simplified by use of Smart contracts and block-chain technology as shown in fig. 4. From the above figure, it is understood that all the processes during the bidding can be completed by smart contract in a transparent and secure manner without much human intervention.

All the bid documents are available on the network and could be accessed by all participants registered on the network. This will avoid the physical bid, which require scrutiny & analysis of many documents submitted by the bidder and there are chances of misses. Here, we are proposing that a certain quantity of iPowercoins must be deposited as EMD (Earnest Money Deposit) by the bidders. These iPowercoins shall be kept as security deposit (SD) for successful bidder and in case of unsuccessful bidder, released as soon as the bidding process is completed. The bidder can utilize the same iPowercoins for any future contracts as well. This type of flexibility is possible only if we are resorting to the smart contracts.

In the above process, the contracts are converted to computer code, stored and replicated on the system and supervised by the network of computers that run the block-chain. This would also result in ledger feedback such as transferring money and receiving the product or service. The Energy banking is the best case suited for P2P modeling using smart contracts, DLT and block-chain technology. To resolve the challenges of Energy...
banking, there is a dire need to ensure that transactions happen in smooth manner, monitored and settled amicably. To smoothen financial settlement process and carry out Energy banking with price tag (different due to different timelines of the contract), use of block-chain with iPowercoins and settlement through smart contract is proposed.

V. PROPOSED MODEL OF ENERGY BANKING

After completion of bidding process, the smart contracts will be put to real operation on the bidding platform as shown in Fig.5.

![Block diagram for proposed model of Energy Banking](image)

This model will calculate the amount of energy exchanged and it’s real time price based on the inputs received from the inputs from RPC (Account statement generation), RLDC (Scheduling part) and Energy exchange (Real time price of Energy). Based on the transactions on a day, the token stacked with RPC/designated authority will be released and deposited to beneficiary’s account. Also the open access charges i.e. RLDC/SLDC application fees, RLDC/SLDC Operating Charges, PoC injection/drawl charges, STU Charges, trading margin, etc. can also be transferred in form of iPowercoins (if these coins are given mandate by regulatory body for Pan India use) to respective beneficiary. Smart contracts will have all the predefined terms and conditions of all transactions of all involved parties. All the data received from different entities, transaction of iPowercoins and calculation of account settlement will be stored in block-chain and can be seen by all registered/involved parties and auditors with an exceptional registry key valid for auditing period only.

VI. CONCLUSION

Energy banking is synonymous to barter system and Block chain is window to future. Whenever east meets west or tradition meets technology, it’s a scintillating experience and similar will be the fusion of energy banking and block-chain. The proposed model of energy banking encrypted with block chain technology will definitely pave way for future smart contracts and carry out energy banking transaction in a more reliable, secure and assured manner. The major advantage of this method is digitally encrypted records, minimum human interface, easy financial transactions, no dependability on single party and accessibility to global markets. Baby steps gets converted in Giant steps with the passage of time, same is being envisaged by authors as this introduction of block chain in energy banking may contribute for development of full-fledged future electricity markets in India.

VII. WAY FORWARD

As of now, there are no specific regulations regarding crypto currency in India which may act as deterrent in implementation of proposed model. Since, the proposed model doesn’t suggest trading of Crypto tokens i.e. iPowercoins and it’s only P2P model, it may be accorded approval by the respective authorities. Regulatory authorities are already in process of redesigning the Indian Electricity markets and the said model of energy banking may also be proposed to be incorporated into it for future regulations. This will aid to use of decentralized ledger, smart contracts and ensure hassle free transactions with fair value of electricity in different times. The author of this paper strongly believes that no transactions of energy can occur at fixed price rather it’s price changes every second. Hence, all energy banking transactions needs to be tagged with real time price, using latest technology with minimal human intervention. Future of Electricity markets lies in adoption of latest technology and this model may act as a catalyst to develop it at a faster pace.

References:

1) Energy Banking agreement executed between Hydro rich states of India.
2) Bid document for Power Swap (Banking) of private utilities of India
3) https://powerledger.io/media/Power-Ledger-Whitepaper-v2.0.pdf
To Build a Secure platform on Blockchain to issue & verify Digital Documents

Joji Varghese
Print2Block
Chennai, India

Abstract—Our current, mostly analog system for managing and Issuing Documents or Certificates are slow, complicated, and unreliable. Relying on Single authority to verify the authenticity of the same Document are slower, complex, and impermanent even in this digital age. Document and Record Fraud is a big challenge for institutes, company and Government to tackle, which in turn results in huge financial losses. Government Inter Departmental Verification of documents consumes time and cost resulting delay in businesses process at grass root level.

Keywords—Blockchain, digital Documents, Verification, Document Fraud, Private Blockchain

I. INTRODUCTION

Our current, mostly analog system for managing and Creating Digital Documents or Certificates are slow, complicated, and unreliable. For past 20 years of technology evolution, we have been creating software’s to generate Digital Document according to the business processes and we do not have digital document Issuing system other than the existing PDF and other document formats that can only be used to send via email or file sharing methods.

Relying on Single authority to verify the authenticity of the same Issued Digital Document are slower, complex, and impermanent even in this digital age. Document and Record Fraud is a big challenge for institutes, company and Government to tackle, which in turn results in huge financial losses.

II. HOW ARE WE SOLVING IT

Print2block, has developed a Robust truthful Infrastructure platform and a Advance Human Interactive Hardware USB device for issuing Digital Documents on Private and Public Blockchain called DocChain.io

DocChain.io Make Any Documents (Eg. Academic Credentials, Professional Certifications, Official Letters & Records) truly Digital by Authentic, Verifiable, Most Secure, Immutable, Trackable and Notarized on a Private or Public blockchain.

DocChain.io Creates Verifiable Documents on the BLOCKCHAIN. -New technology of TRUST: The world needs a new system to record, house, curate, SECURE, and distribute records.

The blockchain empowers institutions with a permanent and tamper-proof infrastructure of TRUST. The blockchain acts as a NOTARY that can always attest to the authenticity of records.

The new technology is redefining the way we transact. If that sounds incredibly far-reaching, that’s because it is. Blockchain combines the openness of the internet with the security of cryptography to give everyone a faster, safer way to verify key information and establish trust.

DocChain.io enable a drag and drop to

1. Publish the Documents on a Private or Public Blockchain (with Our In-house Advance Human Interactive Hardware USB device)
2. Manage your users.
3. Revoke a document
4. Create time-bound document. (Smart/Programmable documents)
5. Charge a fee for instant document Verification on your Blockchain.


DocChain.io offers a platform to create an application easily for any document to be Secure, Verifiable, Trackable, Shareable for Hospital records, Insurance Contract, Letter of Credit, offer letters etc. Which Helps Institutions and Government body to stop Fraud document and records more perfectly without human interventions by making Blockchain as their sole Notary and verifier.

III. TECHNOLOGIES USED

Hyperledger Fabric
Nodejs
Ethereum
IPFS
In House Tools for peer and cluster Management
IV. THE IMPACT – SOCIAL, ECONOMIC, ENVIRONMENTAL OR OTHER

With DocCahin.io having Blockchain to acts as a NOTARY that can always attest to the authenticity of records, with 100% integrity cutting the middle man, saving huge time and cost for the document holder. This Truth Infrastructure can eliminate corruption at the grassroots level, where document holder do not have to serve the mercy of a human verification process. When an Institution or Government issue a valid document (like any Certificate) to its member, it builds the trust of receiver that they will verify the same document seamlessly, which can be achieved by Docchain.io. Document and Record Fraud is a big challenge for institutes, company and Government to tackle, which in turn results in huge financial losses, which can be stopped using the best security of cryptography gives everyone faster, safer way to verify key information & establish trust.

DEMO

https://drive.google.com/open?id=1meR4xurP1naFs-I1MEZOmgY5VKJXg5j84
Secure and Decentralized Live Streaming using Blockchain and IPFS

Shachindra
Blockchain Architect
REVOtic Engineering
New Delhi, India
shachindra@revoticengineering.com

Sagar Ganiga
Application Developer
REVOtic Engineering
New Delhi, India
sagar@revoticengineering.com

Shreya Saha
Application Developer
REVOtic Engineering
New Delhi, India
shreya@revoticengineering.com

Anish Mishra
Infrastructure & Security Architect
REVOtic Engineering
New Delhi, India
anish@revoticengineering.com

Meit Maheshwari
Blockchain Developer
REVOtic Engineering
New Delhi, India
meit@revoticengineering.com

Gaurav Kumar
Full Stack Developer
REVOtic Engineering
New Delhi, India
gk@revoticengineering.com

Abstract — Decentralized cloud systems are proving to be much more advantageous than centralized cloud systems. They distribute power away from a central authority, cut down operation cost, have greater fault tolerance, fewer trust requirements between storage providers and data owners and is less prone to attacks. InterPlanetary File System, a protocol to create a content-addressable, peer-to-peer method of storing and sharing hypermedia in a distributed file system can revolutionize how we share the media content over the internet. We provide an overview of the current systems to stream media over the internet and describe various problems that these systems face with regards to media delivery, governance, and distribution. We exhibit, how with the help of IPFS, Blockchain based Smart Contracts and HTTP Live Streaming (HLS), it is possible to minimize, avoid and diminish the problems associated with the traditional media delivery system and how we can improve the overall efficiency of media delivery systems. We explain how the conventional framework of media delivery can be transformed by IPFS based delivery network supported by HLS streaming for all kinds of distribution model (live or on-demand). We also propose a novel method to decentralize the cloud storage system using a separate server and client-side applications.

Keywords—IPFS, Internet media delivery networks, HLS, Streaming with IPFS, Distributed Ledger Technologies

I. INTRODUCTION

Video streaming over the Internet is one of the most popular activities worldwide. Streaming media accounts for over two-thirds of all internet traffic and it is estimated to jump 82 percent by 2020. To handle this ever increasing traffic of media, the internet has matured over the last 3 decades and the current scenario demand one more revolutionary change in the backbone of the internet. Today’s media delivery networks are experiencing problems related to limited bandwidth, quality of streaming, security and censorship concerns. In this paper, we explain how these issues can be addressed with the help of emerging technologies like IPFS supported by HLS.

II. EVOLUTION OF STREAMING MEDIA TECHNOLOGIES

The vision of streaming media occurred at a point when primary multimedia technologies were prominent among desktop users. They ran unique software created to decompress and render these files on the screen. The initial and key annex of this prototype was the conviction of downloadable media on the Internet. Nevertheless, this was not an adequate action for consumers with scant amounts of storage, slow internet connection, and limited tolerance. This emphasized the creation of streaming media, a technology that facilitates the user to experience a media content on-the-fly over the internet.

This technological evolution includes the following standards

- HTTP-Based Streaming
- First Servers and Protocols for Streaming Media
- First Video Codecs for Streaming Media
- RealSystem G2
- Distributed Media Delivery Networks

III. CURRENT SCENARIO OF STREAMING OVER INTERNET

In the last few years, video-supported applications, and especially video streaming applications, have become quite popular. Many providers started publishing their content such as news, series, and movies on their dedicated Web sites or for this purpose used dedicated sharing services such as YouTube, Hulu in the US, and many others.

In the above mentioned services, video streaming is usually based on the HTTP and TCP, and the video player is embedded in a web browser. TCP is currently the most widely used transport protocol on the Internet but it is commonly considered to be unsuitable for multimedia streaming. The HTTP and TCP are general purpose protocols and were not specifically designed or optimized for streaming media delivery. Bandwidth is precious. When we have a lot of people requesting the media over the internet, there is a lot of bandwidth that travels across the internet. These connections are uni-cast and they stream to each individual person. If there are 500 people watching, each person has to get a copy of that media. In this case, bandwidth increases with the number of people. For 500 people we need 500 times the bandwidth. To tackle this
issue, companies like YouTube are dependent on CDNs. Basically, they are fixing this issue by adding lot more of computers and by moving the distribution centers nearer to the mass users. To make matters more complicated, the difference between yearly price drop in storage (40%) and bandwidth (26%) is almost double. All this means that we will have more people, streaming more data on channels that are not scaling as fast. This creates congestion that at one point in time cannot be solved by just adding more hardware.

Imagine you are in a lecture together with 100 people and you all watch the same video. What happens is that this video has to be fetched from the closest nodes of Google, streamed to each of the attendee’s laptops and repeated 100 times. Instead of students, who have an identical copy of the video, sharing it with each other, we propagate large amounts of data, long distances, multiple times. Inefficient, but this is how HTTP works and it is creating large congestion problems on the backbone of the Internet.

The internet is not 100% centralized since no single corporation owns the entire internet. But relatively few large corporations are responsible for hosting essential elements of what we consider the internet. This kind of system naturally has a single point of failure. This point of failure can be misused to disable access to the content for an entire country. Also, these traditional streaming services are under the authority of one or few big organizations. Quality of service and censorship rules are decided by them and the users need to follow that without any questions. Making it a centralized system of power and authority. Current cloud architectures are completely reliant on large storage providers such as Google, Amazon or Microsoft acting as trusted third parties that store and transfer data. Encryption is not widely adopted and the current architecture is susceptible to many security vulnerabilities. Many storage devices rely on the same infrastructure which is why failures are often correlated across systems or files.

Decentralization can be defined as distributing power away from a central authority or location. In a decentralized storage network, the stored information is distributed across decentralized clients, and each client node encrypts the data to ensure data security and data integrity is maintained using a proof of retrievability. Putting data on an open peer to peer market drives down cost for storage devices. The data is also resistant to unauthorized access, tampering, censorship and data failures. A decentralized storage network has providers of storage capacity that are economically rewarded from the users renting storage using smart contracts. Decentralized storage networks allow for micropayments where payments are directly tied to audits of how the files have been stored. This minimizes how much trust is needed between storage providers and data owners. Decentralized networks cannot be controlled by one authority figure or government, but they are logically centralized as in there is one commonly agreed state and the entire system behaves like one supercomputer.

IV. Proposed Solution

Our system uses the following components of modern tech to build a safe and efficient method for media streaming over the internet.

A. InterPlanetary File System (IPFS)

IPFS or InterPlanetary File System is an open-source protocol and network designed to create a content-addressable, peer-to-peer method of storing and sharing hypermedia in a distributed file system. It aims to make the web faster, safer, and more open. IPFS works by connecting all devices on the network to the same file structure. This file structure is a Merkle DAG, which combines Merkle trees, and Directed Acyclic Graphs (used in Git version control, which also allows users to see the versions of content on IPFS).

This is the process of adding and retrieving files from IPFS.

1) Each file and all of the blocks within it are given a unique fingerprint called a cryptographic hash.
2) IPFS removes duplicates across the network.
3) Each network node stores only content it is interested in, and some indexing information that helps figure out who is storing what.
4) When looking up files, you're asking the network to find nodes storing the content behind a unique hash.
5) Every file can be found by human-readable names using a decentralized naming system called IPNS.

HTTP is inefficient and expensive. HTTP downloads a file from a single computer at a time, instead of getting pieces from multiple computers simultaneously. With video delivery, a P2P approach could save 60% in bandwidth costs. IPFS makes it possible to distribute high volumes of data with high efficiency. And zero duplication means savings in storage. IPFS keeps every version of your files and makes it simple to set up resilient networks for mirroring of data. The webs centralization limits the opportunity. IPFS aims to replace HTTP. IPFS is becoming a new major subsystem of the internet. If built right, it could complement or replace HTTP. In IPFS streaming we don't need to push the content to every user. All you have to do is, push the content to the IPFS gateways. Anyone who wants that content can pick it up from there. IPFS gateway caches the content locally. So a number of gateways, more content sources. Each file and all of the blocks within it are given a unique fingerprint called a cryptographic hash. IPFS removes duplication across the network. Each network node stores only content it is interested in, and some indexing information that helps figure out who is storing what. Each network node stores only content it is interested in, and some indexing information that helps figure out who is storing what. When looking up files, you’re asking the network to find nodes storing the content behind a unique hash.

IPFS provides persistence storage of those live streamed media content so that if someone misses the live streaming, they can always come back to find a saved copy of it. The network automatically deletes duplicates and tracks version
history. With our interface we also allow users to schedule the content for live streaming and broadcasting. Users can record the media and automate the system to stream that media on the given time for the specified duration. IPFS helps to resolve congestion and overly controlling governments by distribution. Instead of locations, IPFS addresses point directly to the resources and it makes sure that this data comes from the closest sources. This means that if a classroom full of students would watch the same video, they would fetch it from each other instead of any central location. This would make streaming a 4k video bufferless.

Another advantage of IPFS is that the user can download parts of a file from various sources at once and combine it at their side rather than downloading the whole file from a single source.

B. HTTP Live Streaming (HLS)

Streaming performance over IPFS can be increased by optimizing and compressing the video in right formats. HLS or HTTP Live Streaming is a video streaming format first introduced by Apple. It breaks streams into small file-based segments which are widely supported format for viewing streams in almost real time. HLS is designed for reliability and dynamically adapts to network conditions by optimizing playback. HLS can be easily integrated with HTML 5. HLS is quickly making its way up the technology ladder, thanks to its superior features and versatility. HLS divides the video chunk into fragments of equal length, kept as .ts files. It also creates an index file that contains references of the fragmented files, saved as .M3U8.

When an HLS video stream is initiated, the first file to download is the manifest. This file has the extension M3U8 and provides the video player with information about the various bit rates available for streaming.

C. How IPFS Helps in Streaming

There are 2 parts to this system, one who publishes the data and one who subscribes. We are calling the publisher section as “media_publisher” and subscriber as “media_subscriber”. The architecture below describes how each component works and communicates with each other.

1) Getting the Content
   a) WebRTC provides web browsers and mobile applications with real-time communication via simple application programming interfaces.
   b) With WebRTC we transfer the webcam content to node server as WebM with VP9 codec every 10 seconds.

2) Transferring this data to Server
   a) From Publisher, we send this WebM data over Socket.io to our server
   b) We create different room/folder for each session with the ID of Publisher.
   c) While doing this, Publishers can select the target node they want to save this data. Either they can store the chunks on their local gateway or on the cloud-based gateway.

3) Converting to HLS
   a) Once we receive the content chunk at server side, using FFmpeg we convert that chunk into HLS format.
   b) It creates an m3u8 file and ts files. We add these files into a folder specifically dedicated to this chunk inside the folder dedicated for this session of the live stream.

4) Adding the files to IPFS
a) We add the newly created chunk folder to IPFS.
b) In this way, we only add a small chunk of data to IPFS every 10 seconds and send this hash of it to subscribed viewers.
5) Sending this data to subscribers
   a) Subscribers receive the hash for chunk as soon as it is added to the IPFS
   b) This transfer of chunk happens over Socket.io
   c) Each session has a different socket session and this hash is shared over that session.
6) Play it on the Subscriber Side
   a) As we get the hash on the subscriber side, HLS.js picks hash from the queue and starts playing it.
   b) HLS.js is connected to IPFS and gets the data for the given hash directly.
7) The stored version of the live session
   a) Once the live session is over, the publisher has the ability to add the recorded version of this live session to IPFS and share its hash over the socket.
   b) This will have the entire session recorded during the live session.
8) Pin the content locally
   a) We provide a separate server and client-side API (discussed in more detail in 1) to pin the media content locally to the local IPFS node of the user for faster and efficient access.
   b) We are able to remove files from the main server that is already being hosted by a large number of clients. Hence, the storage is not wasted.
   c) This also leads to decentralization of the entire system.

This is one cycle of chunk recording, conversion to HLS, adding to IPFS, transfer to the subscriber and playing it on HLS.js. This operation happens continuously for various concurrent live streams.

D. FFmpeg

We are using FFmpeg to convert the incoming chunks into HLS. FFmpeg is a free and open-source utility for converting, recording, splicing, editing, playing, encoding, muxing, demuxing, and streaming multimedia files. It can work with audio, images, and video in basically any codec or format used in the past 20 years. With FFmpeg, we can convert between different file formats and codecs, adjust bitrate (both audio and video) and broadcast a live stream video feed. It can effortlessly parse a file to any format, convert it to a different format and even transmit it through a network via any protocol. Storing incoming chunk on SSD will further improve the performance and conversion time.

E. Smart Contract

1) Authentication
   With the increasing threat to the data leak in the current architecture. Data reliability is 0%. As anyone can manipulate personal data and steal or pretend to be that person.
   a) With the help of blockchain, we can authenticate whether user claims to be is the same person or not.
   b) Manipulating the data or reviewing someone else data, is next to impossible. As the only user has access to those personal data.
   c) As mentioned above, it becomes easy to authenticate the ownership of the streaming data. And it cannot be changed at any given point in the system.
   d) This authenticity is achieved, in the system with the help of blockchain. As two address cannot be the same in the system. These nature of the blockchain provide the authenticity of the streaming data in IPFS.

2) Authorization
   a) After the streaming gets complete over the blockchain, a file hash is generated and stored in the blockchain mapped with the publisher address. As this process will help the users to authorize and authenticate the use of the streaming data by other users.
   b) Streamer can put the validation as to who can view or download the data. This process puts the extra layer of security on the data streamed over the IPFS network.
   c) Three types of function generator paid streaming, private streaming, and public streaming. All the functionality have a different way of working. Please find the brief details of the same below:
      i) Paid Streaming: User will have to pay tokens to watch the streaming
      ii) Private Streaming: User will invite the other users or a group
      iii) Public Streaming: Every user can view the streaming without spending tokens

![Diagram](image_url)
d) As mentioned above, it becomes easy to authenticate the ownership of the streaming. And it can not be changed by anyone. This authenticity is achieved, as no two address can be the same in the system.

3) Accessibility
   a) Being the nature of decentralization, Blockchain offers a wide range of accessibility to the data.
   b) The publisher can set the access level to the data added in the IPFS node, knowing the level and importance of the data.
   c) As blockchain is easily accessible by connecting the node. With the inclusion of IPFS, it becomes very easy to access the data even without internet.
   d) Whenever system is connected to internet and node gets sync with the network, Data’s that are accessible can be view at any given point of time.
   e) After adding the accessible level, the publisher can also validate whether the same person has accessed the data or not.

4) Security
   a) Data leak, data hack, duplication, and many other threat levels which today’s generation facing, the integrity of data is lost.
   b) Security to the data added in any web application can be stolen at any given point making it vulnerable.
   c) Blockchain uses many different algorithms to create the hash. And all the data is stored in the hash. It is practically impossible to crack the hash.
   d) Once the transaction is done in the blockchain, and validators, validate the data. It becomes next to impossible to break the security. Making data very secure and reliable.
   e) The source of the data is always known and it helps to track the authenticity of the data. Fake data or spam data can be blocked at any given time if the regulation is followed.
   f) Connecting the IPFS node to blockchain makes it very secure, as no one can steal the data. The only authorized person can view or edit the data.

5) Token Transaction
   a) Uploading data into IPFS or streaming data using IPFS can be used to generate the revenue for both publisher or viewer.
   b) The publisher can set the particular token amount to view or subscribe to the data. Making it easy to earn medium.
   c) Transaction of token takes very less amount of time as blockchain uses the concept of remittance free world.

3) Secure and controlled Chat Rooms
4) Decentralized Television Network
5) Decentralized subscription based on-demand video
6) The decentralized and secure surveillance system

G. Scalability & Reliability

Above proposed solution is scalable due to the nature of IPFS and private POA blockchain that is being used. Having private blockchain in place gives us easy access to scale our this platform up to 120 transactions per second.

The native web application that gives access to the platform could be highly distributed and decentralized by enabling it to be part of IPFS itself.

A lightweight application can sit on the client side to serve dynamic content as well. This enables us to scale and support an increasing number of concurrent live streams.

Having Distributed Ledger Technologies and IPFS along with lightweight client-side application installed by the user can benefit in terms of performance and reliability as well.

H. Security

May it be peer to peer stream or broadcast, it is highly secure and streamer only can control who can have access to their stream(s). Each live stream sessions are recorded on blockchain using unique session token and is controlled by specialized smart contract and it holds all necessary metadata to identify and associate the stream with streamer and viewers.
Streamer can allow specific viewers/wallet address on blockchain and authorize them to view the stream while viewers with valid wallet & private key can only subscribe to the stream and have access to the stream. This approach is highly secure & reliable harnessing the power of decentralization.

This gives tremendous flexibility to control the accessibility of data or stream allowing streamer to be the only identity-defining, managing and maintaining the authorization and security. Only the owner of data can define and control the security parameters associated with their stream, which is way more practical, reliable and secure compared to other means of centralized data delivery.

I. Proposal to decentralize Cloud Storage System

We create a cloud server which stores pictures, audios, videos, and documents. Each of these files is added to IPFS and are assigned a unique hash value. The user has the option to pin any of these files. JSON arrays are maintained unique to each user which contain the hashes of the file pinned by each user, i.e., whenever a user pins a file, its hash value is added to the respective user’s JSON array. A desktop application is running on the machines of each user, which checks the JSON array of the user at regular intervals and pins the file hashes to the local IPFS node of the user. When more than a certain number of clients pin the same file to their local node, it is removed from the cloud-based IPFS node, making its primary use as an IPFS Gateway.

A. Desktop Application

This is in the form of a web app running locally on the user's machine. The pre-requirement of the application is that ipfs.exe should be preinstalled on the client's computer. This application dynamically pins files that the client wants to host from a server cloud to the local IPFS node of the client. The server-less cloud maintains a list of the file hashes that a client wants to host in the form of a JSON string which is updated regularly. The desktop application checks the JSON string at regular intervals and pins files to the local IPFS node of the client. The desktop application also has features where the user can start the IPFS daemon, check if the daemon is running or not and manually pin files to the local IPFS node.

Incentive Model: The users who pin the content locally on their computers are rewarded the token based on certain factors as the duration of storage (i.e. pinning of the hash), size of storage, bandwidth etc. This helps strengthen the IPFS network and allows the users to get an incentive for renting out their hard disk drive. This model is similar to Filecoin except the fact that the platform decides which users would store specific files and issue tokens periodically.

B. Web Application

A lightweight, interactive and secure web application will be used to provide native functionality of authorizing users. A new user can create their account to use IPFS streaming services. They can monitor the stats of IPFS network. Create API key to communicate between a client application and streaming server. Over web application, a user can also check documentation and guides to effectively use IPFS streaming technology.

C. API Server

It primarily performs two major functions. It creates a blockchain wallet for a user taking username and password as the input. A public-private key pair is created for every username. The public key is returned to the user and the private key is encrypted with the entered password and stored. It is also responsible for adding files to IPFS and pinning them to the server IPFS node. The files are categorized into images, audios, videos, and documents. The API also maintains a record using JSON arrays (for every user) that contains the files hashes pinned by each individual user. A file is unpinned from the server IPFS node in case it is pinned by six or more users to their local systems. It basically is a lightweight, interactive and secure web application that will be used to provide functionality to the users.
REFERENCES


IoT Based Blockchain Solution To Endorse Positive Human Behaviour

Annant Vijay Kushwaha  
Wipro Limited  
Bangalore, India  
annant.kushwaha@wipro.com

Harihara Vinayakaram Natarajan  
Wipro Limited  
Bangalore, India  
harihara.natarajan@wipro.com

Keerthana Jayakumaran  
Wipro Limited  
Bangalore, India  
keerthana.jayakumaran@wipro.com

Prachi Gupta  
Wipro Limited  
Bangalore, India  
prachi.gupta4@wipro.com

Raksha M P  
Wipro Limited  
Bangalore, India  
raksha.p82@wipro.com

Sahana Krishna Karki  
Wipro Limited  
Bangalore, India  
sahana.karki1@wipro.com

Abstract—In many enterprises, automation is unevenly spread. There are patches of automation. It is hard and costly to establish a network among sparsely connected devices since they are spread unevenly. By introducing mobile phones of humans (like a cobot) for data collection in the automation process, there is a possibility to increase efficiency, reduce fraud and more importantly create happy employees by removing the fear of automation and significantly reduces resistance to automation and also decreases the deployment time.

A fracking oil well is an example of a typical enterprise and the process of wastewater management is a typical use case scenario that can benefit from such automation. The current mechanism is a time-consuming process and lacks transparency. The payment takes 7-8 weeks due to problems like intentional delays, verification, significant back-office processing and false data/claims(measurements). Additionally, even with full automation and networking, the collection of sensor data for IoT applications will not be reliable and requires significant investments to create a reliable resilient infrastructure. This paper explores a blockchain based smart contract solution as a holistic solution to the above. We have created an experimental platform that would allow validation of the various ideas and measure how they perform in real-world scenarios.

Index Terms—Blockchain, IoT, Game Theory, Human Behavior, Reputation Mechanism, MicroService, Smart Contracts

I. INTRODUCTION

Waste management consists of all the tasks from gathering the waste from the source and transferring it to the disposal sites for harmless disposal after proper treatment. Disposal of waste has to be monitored carefully. If not handled with care, they can cause harmful effects on the environment. The use case that we have considered is wastewater management.

The process includes the disposal of wastewater generated at the oilfields after hydrofracturing. As the wastewater generated has chemicals present, it has to be disposed of properly. An instance of the complete process flow will be called as a trip. The wastewater management lifecycle still follows the traditional methods for the disposal of the wastewater. The process has 3 main stakeholders:

- Truck drivers
- Fleet Owners
- Oil field Owners

The Fleet Owners own multiple trucks and Oil field Owners own the oilfields. When the wastewater is produced as a byproduct of fracking they are stored in the storage tanks and the level of the wastewater is measured manually. On filling up, the water is disposed at a disposal site by a truck. The drivers wait at the oilfields and thus waste a lot of time. Since the process of fracking will pause until the water in the tanks is disposed of this is preferred by the oilfield owners. This invoice generation method is prone to manual errors and false claims since the trucks are typically paid by the amount of water disposed off/mile basis. After the completion of the process, it takes around 7 to 8 weeks to get the invoice and pay the drivers. There are complaints of inappropriate disposal (e.g.) at the roadsides rather than at the disposal site. Inappropriate disposal of wastewater leads to hazardous effects. The water produced can create a bad impact on the environment by contaminating the groundwater if not disposed of properly. The oil companies are responsible for the wastewater management and in cases of inappropriate disposal, the companies have to face legal consequences.

To automate the manual process, we have proposed a solution which comprises of IoT sensors for measuring tank levels and a blockchain smart contracts to orchestrate the entire process, a web/mobile application which enables users to have various options such as selecting a trip, viewing current status, uploading images of tank level measurements, etc. and game theory to promote positive human behavior.

A central database can be used as a data store. But this requires the other stakeholders in the system to trust the entity running the database. Using a blockchain solves the trust issue since it provides a tamper-evident mechanism and also provides a decentralized resilient mechanism.

The blockchain solution additionally provides a mechanism of cryptocurrencies and trusted marketplaces that can be audited. The cryptocurrencies provide a new way of incentives for humans in real-world scenarios leading to different economic models.
The paper is divided into three parts. The first part describes the design solution to the problems mentioned above. It describes the experimental system, data generation, and measurement. The second part details the construction of smart contracts, coin distribution, the economic model, reimbursement and the game theory aspects of choosing drivers while the third part discussed the results, future work and lessons learned.

II. RELATED WORK

The mobile application would be used by drivers and there was also a need for proof of Water tank readings. We decided on using the camera of the driver to click a picture of the meter reading at the well. The application automatically captured the date/time and the location of the picture. We used the “Amazon Rekognition” service to convert the measurement readings into text. Amazon Rekognition[1] is highly scalable and it uses deep learning technology to analyze the image. It provides a simple API that can analyze images and extract text, human faces, known shapes etc. Interestingly the accuracy of detection significantly increases when there are additional objects in the frame. Thus we mandated the use of selfie (a human face). The user feedback on the correctness of the reading is used as a mechanism to rate the effectiveness of the Image to Text conversion process.

We had considered the use of Tangle[2] from IOTA as blockchain layer since there was an element of IoT in the solution. Tangle uses Directed Acyclic Graph (DAG) for its infrastructure that makes it block-less. It has no transaction fees and no limits on scaling that make the IOTA scalable. The network transaction speed increases with new activity and it works around the transaction, not the miners. So there is no miner hence no consensus is required. It also has a token mechanism. Even with all the advantages, we decided to experiment with Ethereum primarily due to the fact of better development tools (ganache, truffle etc.) and the fact that we wanted a trusted permissioned blockchain. Tangle would be a credible choice to be considered for future developments.

There are three ways to incentivize in peer to peer application-

A-Reputation System- In reputation system[3][4] we follow the scoring system where the score is interpreted as the probability of an entity behaving honestly. This system provides us with information regarding the honesty of the peers so this can be used to find the misbehaving users.

B-Tit-for-Tat Schemes- This scheme is also called Barter scheme[5][6]. This scheme encourages mobile users to cooperate by exchanging equal services based on the contribution they have done for others. In [6], Buttyan et al. proposed the use of Tit-for-tat incentivize mechanism where each user estimates the contribution levels of its neighbors based on their behavior in history, and then forward as much traffic to its neighbor in accordance with their contribution levels.

C-Credit-Based Scheme- In a credit based system[7][8], a central authority assigns certain virtual money to each user. When a user wants help from some other user then he has to pay the helper with a certain amount of virtual money.

III. MODEL

We did an eventstorming[15] session with field experts and stakeholders on our use case to decompose it into functional units based on business capabilities. The event storming sessions and the format helped in nailing the requirements of the services such as which services are needed, defining the boundaries/functions of a service, how the interservice communication happens and how to make the services loosely coupled etc. Since our design is based on functional decomposition, it will have the benefits of scaling, accelerate development, reduces the service load, facilitate testing, service dependency knowledge, etc. Below is the architecture diagram which shows the significant components involved such as user/driver/trucker, various types of services and blockchain. To establish communication between the services, we are building a message queue/message broker/in-memory database. We used Redis for pubsub(publisher-subscriber) system. The publisher will publish the data to channel, the subscriber will listen to the sent data.

A. Architecture Diagram

![Model-Architecture Diagram](image-url)
Trucker or operator by providing the respective UI/map region containing assets.

- Asset service: It has complete data of all three assets (oilfield, disposal well, and truck). For example, data will be location(lat,lon) details, unique id, place etc. It will maintain the status of all the assets.
- Location service: It will have the current location of the trucks.
- Storage Tank Service: It will get data related to oilfield tank that needs to be emptied, which will in-turn contact blockchain to do a task and update the state of the smart contract instance.
- Disposal Tank Service: It will get data related to dumping wells or disposal tanks, which will in-turn contact blockchain to do a task and update the state of the smart contract instance.
- Oilfield Perimeter Service: It will have the data about the time of arrival and departure at oilfield site.
- Disposal Perimeter Service: It will have the data about the time of arrival and departure at the disposal site.
- Rating Service: It will provide a rating for the user.
- Optimization Service: It will optimize the given data. For example, a trucker/driver near to an oilfield tank should be selected.
- Waste Management Service: It is the main service which will communicate with blockchain. It will contact other services like optimization, perimeter and rating services for updates. It will have all the details about the wastewater storage tanks which need to be emptied at that particular time, the truckers/drivers available near to that tank, which disposal sites are available to dump, payment details etc. With that, it is responsible to create a trip(data of the process) in the blockchain using contracts(business logic).
- Identity Service: This maintains unique identities of users(i.e, driver / oilfield operator). This provides identities for the stakeholders involved in it.
- Controller Service: It will control/take desired actions regarding the various services appropriately.
- Trip Monitoring Service: All the trips will be monitored through this service.

This model does not account for missing / dropped events and repeated events. We have assumed a reliable event sub-structure (Redis).

2) Blockchain : The solution is built on the ethereum platform and has several contracts deployed. Every event in the life cycle of the trip i.e from the event of wastewater disposal request being raised to wastewater being dumped in the dumping well(every possible event) will be an entry in the blockchain.

One of the problems the solution focuses on is to reduce the invoice generation time for a trip for a driver. Having the entry of each event of the trip lifecycle on the blockchain, the invoice generation will be instantaneous as there is no need for any manual inputs of events to prove the occurrence of the events. After the final event, i.e disposing the wastewater at a disposal well invoice is generated as part of contract logic.

a) Ethereum
1. Ethereum blockchain provides a platform for private network setup.
2. Ethereum allows the use of smart contracts to model any complex business logic.

The Oil field operator, the fleet owner, and the services present represent the nodes of the network deployed using ganache. We also have a 12 node setup on the cloud platform using Ethereum private network.

Different Contracts used in the solution:
- The business logic of the use case is coded by means of a smart contract using solidity language[10][11]. The various contracts deployed for the solution are:
  - Service contract: This contract gives permissions for the services to contact the blockchain. In the solution, not all the services associated has the permission to talk to or contact the blockchain. The services which are given the permission to contact blockchain are Waste Management Service, Storage Tank Service, and Disposal Tank Service.
  - Request contract: Contract schedules the trip, by creating a trip request for a wastewater storage tank filled in an oilfield. wastewater management service is the only service which has the permission to invoke the function to create the trip request.
  - Trip contract :This contract manages the trip end-to-end(trip cycle). Events from a “truck accepting a trip request” to “trip being completed by dumping the wastewater at dumping well” and “invoice being generated” are managed by this contract.
  - Reputation Contract: This contract maintains the reputation score of the participants involved in the trip, i.e reputation score of the trucker/driver.
  - Coin Contract: This contract is used to create ERC-20 tokens called “AQUA COINS” used in the solution.

b) Coin Distribution
Every user onboard in the solution is assigned with these AQUA COINS which are used as a transaction fee, payment and even rewards. Every trip created will have the token equivalent to the cost of the trip locked with the request and on successful completion of the trip, these locked tokens are released to the driver who completes the trip. Initially, we generated 100 thousand coins. The driver who provides the accurate readings of the oilfield data will also be rewarded with these tokens based on his reputation score. The driver who cancels the ongoing trip will lose some tokens as some gas was used to execute the contracts. More trip cancellations will reduce the reputation of the driver and the tokens will be deducted accordingly. The Users who are assigned with these ERC-20 tokens on onboarding are :
  - Oilfield Operators
  - Truckers/Drivers
  - Disposal Site Operators

Note: The services(Waste Management Service, Storage Tank Service and Disposal Tank Service) have strict access control
over the events they are responsible for, i.e in a contract, other services can’t invoke the functions of events they are not responsible for.

b) **User Identification** We are using Indy as the identity of the account rather than the default Ethereum account identification mechanism. During the implementation phase, we realized that the integration of Indy with Ethereum is in the preliminary stages. But we feel using an Identity provider like Indy is the way to proceed in the future

B. Data Collection

1) **IOT based data collection**: The use case we are dealing with involves a sparsely connected geographical area where deploying sensors and establishing communication between them is one of the major concerns. The sensors at the oil fields measure the level of water which comes out as a by-product in the oil drilling process and is to be disposed. The traditional process involves a truck to act as a carrier for disposing wastewater at the dumping wells. Deploying an IoT infrastructure to monitor the whole process is very expensive and power consuming so we will install IoT device where the manual data can lead to tampering of data. Moreover, if IoT devices are not secured properly or if any one of the IoT nodes in a network is faulty, it is vulnerable to various attacks which are not desirable from the business perspective. The integration of IoT with blockchain helps us to overcome few of these challenges. All the sensor data at the oilfields are collected and stored in the blockchain. Records are created for loading the truck and its identity and the trip details are also recorded in the services. This service communicates with the blockchain and thereafter all the events are recorded in the blockchain which is tamper-evident and tracks the end to end process. We expect this solution to be a cost-effective solution than deploying the network connecting all sensors.

2) **Human reliable data collection using image recognition**: We also devised a human reliable data collection based on image recognition. This also allows the driver to confirm the recognized values. Any corrections done are also recorded and we expect to use these for further training and enhancement. It has been observed that the accuracy of text detection improves significantly if it includes other objects as well along with the text. These readings are provided as an input to predict the level of the tank at a given point in time. The data collected in this way is recorded on the blockchain.

We are working on an algorithm to detect faulty data. For a given oilfield, suppose 10 drivers are collecting the data and one of the drivers has recorded faulty data and it doesn’t match with the data collected by other drivers, then the algorithm detects the difference in the data provided by different drivers and the reputation of the driver who provided faulty data goes down. Since it is a repetitive mechanism, repeatedly providing false data will result in a low reputation score which will create problems for the driver in the future. This compels him to be honest. We also have the proposal to use this mechanism to endorse positive human behavior by providing incentives to the person with ERC tokens or rewards. The criteria for incentives will consider various factors like accuracy, previous ratings. The data collected by the sensors and human reliable data will be compared to check for accuracy. All these criteria will be considered in Game theory, based on which reputation will be provided to the person.

C. **Game Theory**

Automation creates the fear of unemployment in people. It is important to analyze the human psychology, rational thinking, etc to have a better and stable ecosystem. There are different types of theories like behavioral economics, rational economic, etc based on the factors affects human behavior. Game theory[12] is one among them which is a mathematical science strategy used to model the behavior of an individual or among competitors. We can apply the same to understand how an individual can behave, how it affects the ecosystem involved in it and also how to influence the behavior, etc. We are using the concepts to influence the decision making/outcome of humans.

a) How to endorse positive behavior?

- If an oilfield tank readings have to be updated to the service, any users/drivers who are near to that oilfield can help to do by taking a photo of tank level(maybe shows tank level and also the driver).
- If a user/driver is not completing the trip within the allocated time. How to influence the driver behavior to complete on time? Another way around if a user/driver is completing the trip before the allocated time. How to encourage the user to do the same?
- Whoever is reading the data will earn some money.
- A user is getting a good rating for the long term. Apart from assigning more trips to him/her in the future.In what way to motivate to do more?

By providing ERC tokens as incentives or rewards for their work done in the above cases, this, in turn, will endorse positive behavior for an individual. Basically trying to reduce the choices human makes(ex: deciding to behave honestly).

b) How to improve data reliability or accuracy?

- Besides IOT device data, human intervention in collecting the same data will be used to create a better predictive algorithm.
- For example, assume drivers are providing the tank readings at times t1,t2,t3..., this will be an intake for the algorithm to predict the time at which oilfield storage tank gets full even before sensor,etc..

c) How it helps in performing strategic decision-making?

- It also plays a major role in strategic decision-making. A control system based on this can make better decisions. For instance, When an error/improper event happens in the event-driven flow, what actions needs to be taken. Different cases such as,
- when a driver accepts the trip and because of some issues(maybe breakdown) could not continue the trip. So the control system needs to schedule the trip to another driver.
When two drivers accept the same trip, how to resolve the conflict.

A wide collection of use-cases can be modeled using game theory.

IV. WORK FLOW DISCUSSION

The workflow of the project involves the user/driver requests, services and contracts.

Below flow will discuss for a trip cycle from contract/blockchain perspective.

- First, the services of stakeholders(oilfield operators, fleet owners, truckers/truck drivers) need to be part of the blockchain. In our case waste-management service(wmg service), Storage tank service and disposal tank service should be onboard in the blockchain.
- After onboarding those services, it is ready to do their respective functions.
- As part of wmg service, it will deploy an instance of the request, service, reputation contracts.
- The user now sends a “create trip” request to the waste-management-service wmg service). Then the service with aid of request contract methods(a deployed instance of it), it will create an instance of trip contract with the trip details and deployed the same in an address. In the same manner for every unique trip details, an instance of the trip contract will be created and deployed in an address. If there is more than one request with the same details, then the request contract will reject it. If the trip contract instance created successfully, then the wmg service will respond to the user with the transaction hash. Note: tripid(unique id for the trip details) will be shared in another way which will be discussed later.
- Then onwards, every other request for that trip should be sent with the trip id which will be used by the contracts to identify the specific deployed instances in the blockchain.
- “Acceptance trip” request to wmg service and the service will make sure the contract details updated with the help of contract methods.
- Next request for ‘Truck reaches the oilfield site’, will be sent to storage tank service. Then the service will get the address of that trip contract instance from request contract using trip id. And call the respective function in trip contract to update the trip details( for example, Trip status,etc..)
- And other subsequent requests to storage tank service will be “water-filled” and “leaving oilfield site”. Then the service will get contract address and called respective functions to bring the trip contract instances with the latest data.
- Remaining consecutive requests “Pumping water at dumping well”, “water dumped at a disposal site” and “leave disposal site” will be taken care by disposal tank service. Then service will obtain deployed contract address and invokes respective methods in trip contract to keep in engage with latest data.
- After “leave disposal site” request happened, invoice generation will happen as the next step. And then the trip contract will determine the score for the driver/user and invoke the reputation contract method with score details.

To keep track of contract instance states, will make the contract/blockchain to emit events after specific task completed. Sending the blockchain events using Redis channels which are created based on different events. It will be effective for the services which are acting based on events. The trip id which is a crucial element will be shared as part of event data.

A. Simulation Results

It is a microservice based architecture where multiple services interact with each other to complete the task. We are going to experiment the framework which has discussed. The major elements involved are Data generator service, driver/user, UI service, Location service, Asset service, Waste Management service(WMG service), Storage tank service, Disposal tank service and Blockchain.

The figure-2 shows events state-diagram. To simulate the real-time environment, we created the data-generator service(or event-generation service) to produce unreal events and sending them to different services(wmg service, storage tank, and disposal tank). The data generator service has a finite state machine which keeps track of the state of a particular trip and helps to generate the next appropriate event. When the data generator sends a request/event data to the WMG service, the WMG service will create contracts and do a specified task and it sends transaction hash back to the data generator.

From a user perspective, we proposed two types of views. Driver view which will be a truck driver/trucker point of view. Figure-3 represents the driver view. It provides the map with assets(oilfields and dumping wells) around his/her truck.
location. The driver can accept any trips available to him/her, then the selected trip status will be tracked. It shows the below driver-oriented details such as ongoing trips (active trips), completed trips, to be selected trips (open trips), directions (directions for the selected trip event list) and current status of a driver. The driver view will change with respect to drivers. Sometimes the assets associated with driver view will be the same when they approximately share the same location. Oilfield view for oilfield operators. It shows all the oilfields and trucks in a particular zone. Figure-4 represents the oilfield view. It reveals the details of all the trips related to that zone. It will be facilitated for operators to monitor as a unit. Besides event list (shows all the record and status of the oilfields in a zone) and open trips, this will also show the statistic analysis of an oilfield. The oilfield view will be based on oilfield operators. So these views will be radically distinctive.

When a driver asks for the view to UI-service which will respond back with driver view. For the view, UI service will request other services such as location, asset, waste-management for truck location, assets around his zone, open trips respectively. It queries open trip details and event list periodically to track current status. According to events, changes will happen in the status of the assets (i.e., when an oilfield gets full, its icon will change to red in color, etc.).

In the same manner, when an oilfield operator demands to UI-service for a view. It will provide oilfield view. To prepare the view, UI-service will contact asset service for the assets. Other data like open trips and event list are being queried periodically. UI-service querying the open trips from wmg-service and event list from redis channels. UI-service almost listens to all channels (for all events) published by blockchain. And asset-service and location service listen to related events through channels to update assets status. When the blockchain generates the events, it publishes them to the Redis channel. When new events happen, the UI-service listens to the events through the subscribed Redis channels and updates the corresponding view.

Let us consider the following scenarios for better understanding.

Scenario 1: A successful trip
Assume the driver gets his/her view. When the storage tank is full, EV-2 (event 2) gets generated according to the figure 2 event flow and it creates an instance of trip contract per trip. Then, the list of available trips will get updated. The driver selects one of the available trips and it sends a request to the Waste management service and the driver will be assigned to that particular trip after EV-3 gets generated. After that, the accepted trip status will be tracked according to the Events generated. The EV-14 (event-14) will get generated, once the truck gets allocated and disposal well is available. When truck reaches oilfield site and starts pumping in, Ev-7 and EV-8 get produced respectively. As per the event flow, the storage tank status will get normal (Not full). The same flow details will get updated in the blockchain. Once the truck reaches the disposal site and starts pumping out then Ev-11 gets generated. When it leaves the site, then Ev-12 is generated and the trip will end. The state will get updated in the blockchain. Then the blockchain will emit EV-13, which corresponds to invoice generation.

Scenario 2: Two drivers accept the same trip
Assume the two drivers get their views. The list of available trips gets updated on the driver view. Suppose two drivers book the same trip, the same will try to get updated on the blockchain. But the blockchain will pick one trip among that based on the proximity and the time at which the drivers booked the trip and reject the other driver’s request.

Another scenario “driver cancels the trip” is explained in the future work.

V. CHALLENGES
Confronted problems like circuit-breaker, socket hangups (communication), Timeout, jitter (sleep module), etc.

- Privacy issues i.e. anyone who is part of the system can view the entire data which is there on the blockchain even though it is a private blockchain. No one can try to enter the blockchain network without verifying the ownership (using identity service). But then the stakeholder’s data will be available to everyone in the network. So privacy is still an issue.
- We used Indy to create identities for the assets and we had to verify the signatures that were done by Indy to verify
the identity of the assets on Ethereum. Indy used BLS signatures which was not supported directly in Ethereum.

- Query slowness. The Ethereum smart contracts need to be queried (i.e.) executed. A typical solution is to provide a relational database that can be used to provide fast queries. This leads to issues of data consistency. The solution of the database also does not work in our case since we are looking at real-time data display. For now, this has been solved by some standard caching techniques within the services.

VI. FUTURE WORK

- We are looking at scenarios where two trucks book the same trip and one of the trips has to be canceled. We are framing a contract which takes care of the gas payments by the system or the driver whoever canceled it.
- In cases where a truck breaks down or if the driver cancels the trip while the trip is going on, then there should be some way to complete the trip by booking another truck. Then it should be decided if the driver has to pay the gas fee for the contract execution or the system itself will pay it.
- Use Tangle as the Blockchain platform of choice since it provides a more flexible permissionless model.
- The current mechanism does not lend itself to a new workflow very easily since the services are aware of the event names. Ideally, the workflow should be determined by writing a new smart contract. The Blockchain should also be leveraged to handle the case of missing / dropped events etc.

VII. CONCLUSION

The proposed solution can be used to monitor the waste management lifecycle by fastening the process of payments and also reduces the time consumed to complete the trip. It also makes the process transparent to avoid false claims. It promotes positive human behavior by incentivizing them with tokens. This solution overcomes all the shortcomings of the current methodology by using the blockchain and makes the process more reliable and accurate.

REFERENCES

[10] https://solidity.readthedocs.io/en/v0.5.1/
[12] https://medium.com/blockchannel/cryptoeconomic-theory-game-theory-basics-fb3a49a9ab1a
[14] https://blog.mapbox.com/a-dive-into-spatial-search-algorithms-ebd0c5e39d2a
A Decentralized, Distributed Approach for Unsolicited Commercial Communication

Divya Pradeepkumar Taori Harihara Vinayakaram Natarajan Prachi Gupta Raksha MP
Wipro Limited Wipro Limited Wipro Limited Wipro Limited
Bengaluru, India Bengaluru, India Bengaluru, India Bengaluru, India
divya.taori@wipro.com harihara.natarajan@wipro.com prachi.gupta4@wipro.com raksha.p82@wipro.com

Abstract—An "Unsolicited Commercial Communication" (UCC) means a commercial communication which a Subscriber opts not to receive. For a long time, Telecom Regulatory Authority of India (TRAI) has been a centralized regulating body for any Commercial Communication and now has a requirement for a DLT based blockchain solution to overcome problems of Subscriber Data leak, Spam Messages, Huge penalization, Law violations etc.

A DLT though helps in creating a decentralized distributed system of trust, it doesn’t focus on privacy concerns of the emerging world. The Gartner Survey claims - "By 2021, 75 percent of public blockchains will suffer "privacy poisoning" - inserted personal data that renders the blockchain noncompliant with privacy laws". [1] and Justice Srikrishna committee report on data protection claims that “Protection of personal data holds the key to empowerment, progress, and innovation.” [2].

Keeping the above claims in consideration, in this paper we are presenting a DLT solution to regulate the current eco-system of Commercial Communication which respects Consumer Consent along with a Zero Knowledge Proof based blockchain solution addressing the Consumer Privacy concerns. This Paper also describes an approach to efficiently store the Consumer Preferences and Consent at their respective registered Access Providers by implementing the concept of Sparse Merkle Trees as a Data Structure, to store the said preferences and consent (>10 million Consumer Data). The solution focuses on ensuring privacy (Subscriber Preference and Consent) across the Telecom Service Providers.

Additionally, we have an AI Module like SimHash, which is used to identify an Unregistered TeleMarketer (Spam Messages) and a Reputation AI model, to implement an incentivization scheme which in turn ensures discipline in Communication. And as a future enhancement AI can be employed towards learning-based automatic detection of Consumer Preference.

Index Terms—ZKP, Merkle tree, DLT, Blockchain

I. INTRODUCTION

For a Subscriber, Unsolicited calls and SMS are a serious problem. TRAI (Telecom Regulatory of India), a centralized regulatory body has come up with initiatives to try and protect subscribers from these telemarketing calls and messages and has set up a regulatory framework for subscribers for not getting calls and SMS from telemarketers if one doesn’t prefer for it.

A Telemarketer register’s with TRAI and enters a signed agreement with the Service Providers to access their resources to send messages to the Subscribers. These are the RTM’s (Registered TeleMarketers) which can be traced by the TRAI. But there might be unregistered telemarketers which cannot be traced by the TRAI and the messages from these UTM’s (Unregistered Telemarketers) are blocked to a Subscriber only when he/she raises a complaint against it and in most cases these UTM’s becomes the main source of spam messages.

The current system of registry and preferences is a case of regulator being involved in a day to day job and currently does not have traceability, enforceability for violations etc. The newer system would envisage the regulator only enforcing regulations.

To overcome the above problems, TRAI has requested for a new regulation to curb spam calls and messages and this includes the use of blockchain technology to ensure regulatory checks to control the flow of commercial communication.

A. Blockchain/Distributed Ledger Technology as a solution

DLT(Distributed Ledger Technology) is considered as a solution for the present problem because:

• It establishes trust without a single authorized, centralized entity.
• It ensures all the necessary pre-checks required to control the flow of commercial messages in an effective and efficient way.
• It records the data and actions in a immutable, non-repudiable and confidential manner.
• The preference registered by a subscriber at his/her access provider can be made safe and secure during exchange and processing.

B. Need for Privacy

In the current ecosystem, the user preferences stored in Do not Disturb (DND) register are exposed to all the
registered entities, leading to a possibility that user’s data can be compromised with third parties. With the advent of Artificial Intelligence and Data Analytics there has been a rise in the data privacy issues. The data analytic run over the history of user preferences can reveal sensitive information like health records based on the health care categories he subscribed to or tracking the user location based on the travel packages he is subscribing for and so on. These issues calls for the necessity to keep the user preferences private. In our proposal, the user’s access provider are entrusted with the storage and maintenance of user’s preferences and is kept private.

According to the new regulation, all the telecom service providers will now adopt DLT as a regulatory technology to develop a UCC ecosystem and solutions around it. Each TSP will own, operate and maintain its own blockchain network or can operate in a network built in collaboration with other service providers.

Unlike the present, the subscriber has more control over the type of messages they receive. In the proposed regulatory, the service providers, will have the provision for a Subscriber to register his preference for commercial messages and a ledger to record the consent of the subscriber.

In this paper we have devised a zero knowledge (for data privacy) oriented, AI (Artificial Intelligence) and a DLT based regulation for UCC. Regulating Unsolicited Commercial Communication/Spam Messages and calls originating from registered telemarketer’s as well as Unregistered Telemarketers and maintaining the subscriber/user preference and consent private for the said message is the main aim of our proposed solution.

II. CURRENT ECOSYSTEM

A. Consumer Preference Registration Portal

Currently, TRAI acts as a centralized authority to regulate the commercial communication though it doesn’t intervene in the process directly. It maintains a registration portal for the telemarketers where it enters into a signed agreement with the access providers who provide its resources. All registered telemarketers are obliged to follow the set regulations by TRAI. As per 'Telecom Commercial Communication Customer Preference Regulation, 2010', to respect the consumer preferences it provides the subscribers to register with their access providers for two options i.e fully blocked category and partially blocked category. The consumer preferences are maintained by National Do Not Call Registry (NDNC) [3]. This portal also facilitates the registered telemarketers to download the preference of users and send the transactional or promotional messages or calls as per preferences.

The consumers register their preferences through one of the facilities provided by the access providers which can be through voice call to 1909, through SMS in a specified format to 1909 or through IVRS by dialing on 1909 as shown in Fig 1. The users are then provided with unique registration number and their request takes seven days from the date of request to come into effect. If the user gets spammed even after registration he can complain with its TSP using Customer Complain Registration Facility (CCRF) mentioning the source which is generally header for registered telemarketer or number for unregistered telemarketer. For detecting unregistered telemarketers, signature solution is used currently which matches the resemblance pattern of the messages with the registered telemarketers.

B. Limitations of the Current system

- Present Regulatory does not stop the Subscriber preference registered with TRAI from being leaked to a registered or an unregistered telemarketer.
- Even though customers opted out of the calls or messages, RTM have succeeded to get the consent of the users by other means surreptitiously.
- There is no robust mechanism to keep the Subscriber preference/Consent private.
- For a complaint raised against a RTM (registered telemarketer) by a subscriber, both the RTM and the Access provider they are registered with are subjected to financial penalization.
- Present regulation, does not have necessary ways and means to protect the data that are made available to the registered telemarketers. Protecting this data may limit the exploitation by UTMs to certain extent.

III. PROPOSED SOLUTION

Blockchain / DLT is a fundamental technology and would make a huge difference in the TSP network and would enable newer business models. The TRAI regulation is a great first step to kick-start the adoption of DLT in the TSP ecosystem and also curb the menace of Unsolicited
Commercial Communication.

Few points to be noted in our proposed solution:

- User Preference data should not be public. And have use techniques like Merkle Proofs and Zero Knowledge Proofs to achieve it.
- Mechanisms other than crowd sourced data (e.g.) True Caller to be used to classify data. The existing 140 numbering series for Tele marketers should be augmented with the protocols like STIR / SHAKEN, CDR analytics, DLT based reputation systems.
- Subscriber onboarding to use a KYC style checking and the DLT based reputation systems.
- Dealing effectively with messaging from RTM’s would also need an effective way to deal with messaging from UTM’s using AI / matching techniques.
- The Scrubber entity (entities ) would be responsible for delivering messages from the RTM and also maintain a record in the blockchain. The tools for auditing these blockchains would also be provided by the Scrubber services.
- The existing TSP’s should also store specific preference data rather than just categories to allow for implicit consent. This requires huge amounts of data and requires efficient storage of data (e.g.) encoding/columnar storage etc for querying.

Solution proposed is in agreement with the TRAI’s requirement but have few following differences from the UCC draft put up by TRAI [4].

1. Preferences of Users are private and is known only to the TSP a user is registered with.
2. Scrubber functionality.
   - Filter RTM messages ( for self ) and deliver RTM messages to peer scrubbers.
   - Deliver report of number of blocked messages / delivered messages.
   - User preference API’s can be embedded in RTM applications allowing users to modify and validate preference changes.
   - Scrubber API can request delivery responses for a set of random numbers.
3. Change in incentive model
   - RTM’s incentivized for not flooding network.
   - Cost of RTM’s flooding the network prohibitively increases.
4. DLT store.
   - For each RTM message store the Merkle Path, Merkle Proof and Delivery Receipt ( for random numbers ).
   - A separate ZKP node providing proof for the blocks in the DLT.
5. Complaint system.
   - An AI based system automatically classifies UTM messages as spam and logs complaints based on rules and reputation system.

The paper focuses on solution proposal of many requirements.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTM Text Messages not respecting preferences</td>
<td>AI based solution in Scrubber</td>
</tr>
<tr>
<td>RTM Text Messages misusing headers</td>
<td>• AI based solution for detecting appropriate classification in Scrubber</td>
</tr>
<tr>
<td></td>
<td>• Reputation of RTM and incentivization mechanisms.</td>
</tr>
<tr>
<td>UTM Text Messages</td>
<td>AI based solution for detecting similarity in SMSC(Small Message Service Center)</td>
</tr>
<tr>
<td>Auditing messages by TRAI / other TSP’s</td>
<td>• Merkle Proofs.</td>
</tr>
<tr>
<td></td>
<td>• Zero Knowledge Proofs.</td>
</tr>
<tr>
<td>Implicit Consent for Subscriber Preferences</td>
<td>• AI/rule based systems.</td>
</tr>
<tr>
<td></td>
<td>• Storage of individual preferences.</td>
</tr>
<tr>
<td>Privacy of User choices</td>
<td>• Merkle Proof storage</td>
</tr>
<tr>
<td></td>
<td>• API in Scrubber</td>
</tr>
</tbody>
</table>

A. Our Solution goals

- Subscriber preference should not be discoverable. A preference registered by a Subscriber should not be discovered by any other Access Provider’s or RTM’s.
- Given above constraints,we should be able to prove that any Access Provider, respects his Subscriber’s preference and delivers messages as per the preference.

B. Our Solution Approach

- A DLT based solution involving all the Access Providers.
- To use the concepts of Merkle trees and Zero Knowledge Proof for the auditing of messages by TRAI and other access providers.
- A separate ZKP (Zero Knowledge Proof) node providing proof for the blocks in the DLT.
- Message delivery to be as per user Preference and Consent.
- Subscriber preference and Subscriber Consent private across TSP’s.
- To use AI Modules to identify and block the messages from UTM’s.
C. Architecture

The DLT network comprises of nodes of various entities like Originating access provider (OAP), Terminating Access Provider (TAP), TRAI as shown in Fig 2. User registers his preferences with its respective access providers using the API for it. Access providers calls for preference storage preference service which generates the Merkle tree for preferences which is an efficient way of storage and then makes an entry of the root in the Blockchain. RTM gets the consent of the users by the exposed API and generates the subscriber list for its clients. Every message originating from RTM passes through scrubber of its respective access provider i.e. OAP and thereafter it is routed through the other TAPs by SMSCs. Scrubber is the node of access providers which is responsible for filtering out the messages from the RTMs as per user preferences. On the contrary, since UTMs are not part of service agreement, it is treated like normal message and goes to Mobile switching centers. To detect the UCC from UTMs we have AI module like SimHash integrated with SMSCs to estimate the set of resemblance of spam messages. The complete flow for our solution will be discussed in the work flow section. Our proposed solution employs the concepts of Sparse Merkle tree, ZKP, AI which we will be discussing in detail in further sections.

The solution regulates the current eco-system of commercial communication which respects subscriber preference with an approach to efficiently store the subscriber preference at their respective registered access providers.

The solution employs the concept of 'Sparse Merkle Tree' as a data structure to store subscriber preference (>10M subscriber data) and delivery receipts of the messages. The merkle root of the tree will be made as an entry in the DLT. The solution uses merkle trees for 2 purpose:

- Preference Merkle Tree - for subscriber preference store.
- Proof Tree - for storing delivery proofs of messages.

D. Sparse Merkle Trees

A sparse Merkle tree is a data structure similar to a standard merkle tree, except that the data contained in it will be indexed and each datapoint is placed at the leaf that corresponds to the datapoint’s index. Similar to standard merkle tree, sparse merkle tree uses merkle proof to prove that a data is part of the tree. Sparse merkle tree proves the non-inclusion of an element in the tree and best part about them is that unlike the standard merkle tree, they represent the key-value store inside the merkle tree [5].

We use Merkle trees as an effective store for User preference because,

1. Merkle tree is a tree of hashes which allows the audit of data much faster.
2. Merkle tree traversal requires only logarithmic space and time.
3. Merkle tree are storage and computation efficient.

In the solution, the merkle tree will have intermediate nodes of a 4 digit prefix (the 4 digit of the phone number which corresponds to the access provider) and the lowest nodes(leaves) will have store of 1000 subscriber preferences. These subscriber preference will be stored as a TLV (Type length value) array. This is because the preference will be sparse Fig 3. Whenever a subscriber registers his preference for a message at his/her access provider, based on the access provider (4 digit prefix) it will be stored in the respective leaf of the merkle tree.

An access provider, does not store the subscriber preference directly into a merkle tree instantly but instead after every certain period of time, say 'X' hours, a new root for the Merkle tree is calculated and this updated tree root will be put on DLT and this mirrors the latest changes in the subscriber preference. This way a subscriber’s preference will be reflected across all the access providers only after the said 'X' hours.

Also every access providers maintains a private database, which will have store of the subscriber preference (same as one store as TLV in merkle tree).

E. Zero Knowledge Proof

Keeping user preference private is one of the important aspect that we are focusing on in our solution. A zero-knowledge proof or protocol is a method by which allows a "prover" to assure a "verifier" that they have knowledge of a secret or statement without revealing the secret itself. It can be built on top of blockchain to ensure privacy in the DLT. In our solution we are using ZK module to ensure privacy of user preferences at the time auditing the
delivery of commercial messages. Proofs can be generated on demand by the proving authority (Access Provider) which can be asked by any other entity in the DLT either TRAI or Access providers. zk-SNARKs are the variant of zero-knowledge proofs in which no interaction is necessary between prover and verifier. Proof computation requires a trusted setup which generates the proving key and verification key to be used for non interactive proof verification [6]. The proof generation requires this proving key, a public input, and private input which are user preferences in our case. The public input is provided by the blockchain stored as tuple which consist of hash of preference, Delivery proof, Merkle path. Zero knowledge proofs helps us in proving the validity of the message delivery as per user preference without revealing the said preferences.

F. AI Module

Detecting UTM’s spam who are not bound by the standard agreement was one of the major challenge. Our solution address this challenge by integrating an AI module at SMSC with the DLT solution. Every incoming or outgoing message irrespective of its source (RTM or UTM) is hashed and stored. The sender-receiver graph is computed for each of the SMS clusters. Algorithms like SimHash is used for quickly estimating the resemblance of the two graphs i.e. from RTMs and UTMs. Based on the similarity the message is listed as originated from UTM.

Additionally, we use Artificial Intelligence to solve the problem of the use of transactional pipeline for sending promotional messages by Registered telemarketers. It can be a rule based mechanism which can be extended to match the patterns of promotional or transactional messages.

We also have the provision for providing reputation to the RTMs. The reputation can be computed offchain and referred on blockchain [7]. In the context of number of messages sent as per the user preferences to the scrubber by RTMs and the statistics of the discarded messages by the terminating access providers, reputation may be awarded to the RTMs. In case of any unsolicited commercial communication or the misuse of transactional channel, their reputation may be degraded by contextual anamoly detection mechanism over a period of time.

IV. Work Flow

A. RTM Registration

- Every TSP, maintains a ledger called ‘RTM REGISTRY’, which maintains a list of RTM’s registered with it, and exposes an end point for new RTM Registration.
- On successful registration, a header will be assigned to the newly registered RTM specific to a category and this assignment is recorded in a HEADER REGISTRY maintained at an OAP’s (Originating access provider) end.
- The RTM with a valid HEADER assigned, registers it’s message template with the TEMPLATE REGISTRY maintained at the OAP, which records the active template registered by the RTM. This is useful in filtering of unwanted/unregistered message format form RTM, from reaching the Subscriber.
- When a RTM wants to send the message to its Subscriber, it sends the intended message with Subscriber list to its OAP which then filters out the list and routes the message to its respective TAP’s (Terminating Access provider) through its SMSC.
- SMSC stores the hash of every message sent from it because it helps in identification of UTM message of similar sort.

B. Delivery of RTM Messages

Refer Fig 4 to get the graphical sequence of the flow:

- User registers his preferences with his registered TSP which will be stored in a private Database maintained by Customer Preference Service of the Access Provider.
- The RTM which wants to send out a Commercial Message, sends a list of Subscriber along with the intended message to the Scrubber of its registered OAP.
- Every Access Provider will have a Scrubber which is responsible for filtering out the messages as per user preferences at its end. Every Scrubber Fig 5 has multiple services embedded into them such as.
  - Customer Preference Service: to register Subscriber Preference and store it in a private Database.

1 OAP is the Access provider a RTM is registered with, and is responsible for the RTM message delivery.
2 TAP is the access provider, a Subscriber is registered with and it is the one which holds the subscriber preference and is responsible for filtering the message for its subscriber based on their preference.
3 In cases, for a message both its OAP and TAP might be same.
Merle Tree Service: to construct Preference tree and Proof tree.
Blockchain Service: node to connect to a Blockchain.
SMSC Adapter Service: to route the message with the Subscriber list to SMSC which delivers the message to the respective Subscriber in the list.
QED-it Service: QED-it node used to build a zero knowledge proof for delivery as per preference for a random number 'N'.

Scrubber of an OAP, on receiving a message along with the Subscriber List from the RTM, identifies the TAP's of the numbers in the Subscriber List and routes the message to the Scrubber's of those TAP's.

While routing the message to the Scrubber of a TAP, an array of random number will also be sent along with the Subscriber List and message, for delivery verification of message.

And the Blockchain will contain 1 entry per RTM Message sent. The format of the entry will be:

\[
\text{hash ( preferences (0 - n-1) }, \text{ hash (preference (n) ), delivery proof (n),hash( preferences (n+1 - m-1) ), hash ( preference (m) ), delivery proof (m),hash( preference (m+1,999))), Merkle Path}
\]

From a Scrubber after filtering, the message is routed to a SMSC along with a Subscriber List who has preferred for the said message which then delivers the message to the Subscriber.

There will be entry of preferences, delivery proof, Merkle path for each of the RTM message in the DLT. And the delivery receipts for a message will not be instantly generated, but after every 'X' hours, the delivery receipts will be collected from the SMSC and a Proof tree will be generated and the Merkle root of the proof tree will be put on the DLT.

C. Auditing of Message Delivery
Refer Fig 6

D. Detection of UTM Messages

To preserve user preference as private entity, we have used the concept of "Zero Knowledge Proofs" which helps us in proving the validity of a message delivery for a user preference or in common, the validity of user preference registered at his/her TAP, without having to reveal the said preference.

The TAP, which stores the User Preference is the Prover who has to prove to a Verifier, who might be another Scrubber of a TSP or even an auditing entity like TRAI, the validity of the User Preference and User Consent without revealing the subscriber preference stored at its end.

A regulator/verifier takes the entry of a message from the blockchain and requests for a proof of delivery as per Subscriber preference. A prover must be able to construct a zero knowledge proof, by taking the public inputs from the blockchain which are nothing but entries of deliveries and preferences which are private to the prover (private input).

In a Scrubber of a TAP, after receiving the Delivery Receipt of a number 'N', it refers to the preference root and preference path stored in the DLT for the number 'N'. By knowing the root and the path for a number, the Scrubber has the knowledge of the preference of a Subscriber with number 'N'.

Based on the Subscriber preference and Delivery Receipt of a number, zero knowledge proof for delivery of the message is constructed and provided to the verifier who requested for the proof.

The verifier then checks if the proof is valid. And if the proof is not valid, it calls the reputation service which then reduces the reputation score of the TAP and if the reputation score is below a threshold, will be penalized.

D. Detection of UTM Messages

An AI Module employed at SMSC's of the service provider's, is used to detect UTM source.
• Every message from the RTM’s routed through SMSC (both at TAP and OAP end) will be hashed and stored in it.
• When a SMSC receives a message from an unregistered source, it hashes the message from the source and verifies it against the stored hashed message of the RTM's.
• If the hash of the message is similar to the stored hash of RTM message, it is classified as an UTM message and is made as an entry in the Complaint registry and further blocks the message from reaching the subscriber.
• We use the concept of hash algorithm like SimHash to estimate the similarity in the hashes of the messages.

E. Complaint Model
• The complaint model allows user to complain against any particular RTM/UTM which will be blocked at its TAP’s end respecting customer’s preferences.
• When a RTM /UTM sends a message, every message irrespective of RTM or UTM incoming and outgoing from SMSC will be hashed and stored.
• AI module analyses these messages with the RTM message graph and detects UTM based on resemblance.
• If the user complaint is against a RTM source, the preference of the user is checked and if the message is against the subscriber preference, the Process/Reputation module employed will regulate the RTMs. And their reputation will decrease accordingly based on defined rules.
• If the reputation score of the RTM is below a fixed threshold, it notifies the RTM of the violation and blocks any future message of RTM to the Subscriber.
• If a UTM is detected, it is made as an entry of complaint against the Sender in the Complaint Registry and blocks the message from reaching the subscriber.

V. CHALLENGES
The major challenges we encountered in deploying our solution at such a large scale are:
• Initially while deploying the DLT, there can be cases where every access providers may not be part of DLT and thereby not all the RTMs will be part of RTM registry. In that case we are proposed to treat RTM as UTM and apply the rule to regulate it and incentivizing it for any UCC.
• Registration of closed RTMs are feasible. Handling law violators RTM who spams and are closed and then re-registers with other access providers. To address this we have our reputation model such that they send a message only if they have a reputation and the reputation can only be gained if people endorse / accept it.
• In sparse merkle tree user preference data will be indexed. When the user ports his number, its operator prefix will still be the same but it should not be accessed by the previous operator. Operators will need an access key to decrypt the private user preferences for proof generation even though its index in its operator prefix level.

VI. CONCLUSION
The DLT based solution proposed, regulates the Unsolicited Commercial Communication/Spam Message and calls originating from Registered Telemarketer’s as well as Unregistered Telemarketers by employing concepts of Merkle trees for efficient storage of preferences, zero knowledge proof to ensure privacy, AI and hash algorithms to solve the problems of subscriber preference data store, privacy, unregistered source detection, anomaly detection etc. This solution is expected to solve the issues in the present system like de-registration of subscriber preference after mobile number portability, leaking of user preferences etc.

The system designed is decentralized and distributed without a single point of trust, and aims at preserving subscriber privacy.

VII. UNSOLVED PROBLEMS
• The RTM’s currently spam customers and can easily set up another RTM in the same Access provider or another access provider. We should provide a mechanism to ensure that is costly to switch. We could provide for RTM’s to begin operation only if they get a certain amount of reputation and this could prevent spam.
• The reputation system as currently envisioned is threshold based and hence is vulnerable to established players violating the system. We need to factor in a reputation system that has the recency factor involved and also frequency of violations.
• Trusting Access providers with the preferences of the users stored at its end is still a issue and we need to find out the ways of to store the preferences without trusting access providers in a trust less system.
• One of the requirements is to ensure that the access provider does not get any undue advantage (over other access providers) to harvest the data available with it.

VIII. FUTURE WORK
• As part of future work we are suggesting to use AI for detecting the user preferences implicitly. We are proposing a rule based approach for this.
• To incorporate Mobile Number Portability in the same ecosystem as UCC by designing a Receiver originated (RO) model with zero knowledge proof ensuring that the subscriber has all his dues cleared with the previous operator(Donor Operator) and validating the reasons in case of any delays.
REFERENCES

Addressing Blockchain Scaling by Chain Splitting

Kazim Rıfat Özyılmaz
Department of Computer Engineering
Bogazici University
Istanbul, Turkey
kazim@monolytic.com

Harsh Patel
Ahmedabad, India
reach@harshpatel.space

Ankit Malik
New Delhi, India
ankit@ankitmalik.in

Abstract—The Bitcoin protocol is a significant milestone in the history of money. However, its adoption is currently constrained by the transaction limits of the system. As the chief problem of blockchain technology, the scaling issue has attracted many valuable solutions both on-chain and off-chain.

In this paper, our goal is to explore the notion of unspent transaction outputs (UTXOs) to propose an augmented Bitcoin protocol that can scale gracefully. Our proposal aims to increase the transaction throughput by partitioning the UTXO space and splitting the blockchain. In addition, a new type of Bitcoin node is introduced to preserve the capability to run validating nodes in low-bandwidth environments, despite the increased transaction throughput.

Keywords—Bitcoin, blockchain, consensus, UTXO, scaling, block size

I. INTRODUCTION

Scalability is one of the most important aspects affecting Bitcoin’s adoption. Limits of scalability express themselves as high transaction fees, which affects usability and adoption negatively. To improve transaction throughput, various proposals have been made, starting with directly increasing block size. However, the most effective scaling improvement already integrated to Bitcoin is 'Segregated Witness' [1]. This increased the block capacity by introducing the block weight metric. Another notable attempt to solve the scalability problem is the Bitcoin-NG protocol [2] which introduced an additional mining process where miners gain the capability to mine microblocks by mining a regular Bitcoin block.

In this paper, we propose a solution that increases the transaction throughput of the Bitcoin network without hurting network decentralization in terms of bandwidth requirements. By partitioning the UTXO space and splitting the blockchain into a tree structure, independently operating sub-chains will be created at every split event. As a result, a new block from all sub-chains will be mined at every block interval, increasing the transaction throughput exponentially. Moreover, in order to preserve the capability to run a node in this increasing bandwidth requirement, a new type of Bitcoin node (the half node) is introduced. Although this new node type does not store the complete blockchain, it can independently verify the transactions on the sub-chain it is tracking, which gives it the capability to operate in low-bandwidth environments.

In the next section (Section II), an overview of the core concepts is presented. Then, the general mechanics and technical details (Section III) of the proposal are described. Effects of the proposal to mining (Section IV) and network organization (Section V) are discussed in the following sections. Next, the transactions discussion (Section VI) provides insights on transactions in the proposed system. A section dedicated to comparison of the split-scale proposal to other major Bitcoin scaling solutions (Section VII) is included afterwards.

II. CORE CONCEPTS

A. Unspent Transaction Output (UTXO)

Bitcoin does not use the concept of 'account balance' as Ethereum does. Instead, total balance of a Bitcoin account is the accumulated amount of the transaction outputs that are claimable but not yet spent. These unspent transaction outputs or UTXOs for short, are used as the inputs of the transactions. They are referred using the source transaction hash and index of the output within that source transaction (Listing 1).

The receiving party will have at least one transaction output after the transaction is validated and added to the blockchain. If a UTXO is used as a transaction input and the transaction is already a part of the blockchain, then it is considered spent and thus can not be used a second time as a transaction input.

The UTXO set is stored by nodes in a database called chainstate.db outside the blockchain, which provides persistent key-value storage. As of Bitcoin 0.15.0, the chainstate database

has been changed from a per-transaction model to a per-output model which added benefits like faster serialization, predictable memory usage and better caching [6]. This change may provide a smooth transition for the chain splitting mechanism that is proposed in this paper.

The \texttt{CTxIn} class (Listing 1, line 2-9) is a simplified version of the transaction input. It contains the location of the previous transaction’s output that it claims and a signature that matches the output’s public key. The \texttt{COutPoint} class (Listing 1, line 11-16) in the transaction input shows how UTXOs are actually referred by the input. It contains both the \texttt{transaction hash} and the \texttt{index} of its output. Lastly, the \texttt{CTxOut} class (Listing 1, line 18-23) presents the anatomy of a transaction output in a simplified form. It contains the amount and the script \texttt{scriptPubKey} to claim the output. Below is the \texttt{scriptPubKey} for a standard Pay-to-PubkeyHash (P2PKH) transaction [7]:

\begin{verbatim}
OP_DUP OP_HASH160 pubKeyHash OP_EQUALVERIFY OP_CHECKSIG
\end{verbatim}

On a side note, as of 26th of October 2017, 82% of the Bitcoin transactions are Pay-to-PubkeyHash (P2PKH) [8] so \texttt{scriptPubKey} of these transactions are directly tied to a single receiving address.

B. Memory Pool (mempool)

The memory pool, or mempool, is the memory area reserved by Bitcoin clients to store unconfirmed transactions. Unconfirmed transactions are accumulated in the mempool until they are picked by a miner, mined, and added to the blockchain. Currently, each node maintains its own mempool, having the complete view of all unconfirmed transactions in the Bitcoin network. The amount of memory reserved for mempool varies greatly, the peak point being around 140MB for the last two years [9].

III. SPLIT-SCALE

The idea is to split the Bitcoin blockchain (Figure 1), known here as split events, into multiple sub-chains in order to:

- create independently operating multiple sub-chains, therefore creating multiple blocks instead of one block for every block creation interval (an interval lasts 10 minutes on average).
- provide the flexibility of operating home nodes with less bandwidth and storage requirements. These nodes will have the option to track only a subset of chains without losing any verification capability.

The mechanics of such a split and how UTXO database, mempool or mining operations will be affected will be presented in the following subsections.

A. Split Event

A split event is a deterministic (and repeatable) action that will be triggered as a result of a decision made by the governing authority of the platform and/or certain performance metrics showing that the system is pushing its boundaries in terms of transaction throughput. Details on when a split event will occur or who is going to decide for it is a separate topic that will not be addressed in the scope of this paper. At the split event, following changes will happen on Bitcoin clients:

- UTXO database will be divided based on their \texttt{scriptPubKey} hashes.
- the UTXO split may be implemented logically (UTXO hashes in binary form that start with 0 or 1 for the first split) or economically (find the 256bit number that divides the Bitcoin supply in half).
- the mempool will be divided based on which sub-chain they are tracking.
- miners must create a block for each sub-chain, and a separate block containing these block headers to claim a block reward.

B. Dividing the Chainstate Database

Based on the information given on the UTXO structure, it is possible to create a 256-bit hash value from the \texttt{scriptPubKey} of any given UTXO. The hash function is selected as double SHA256() and a hash value representing all UTXOs with the same \texttt{scriptPubKey} can be given as follows:

\[
\text{hash}_{\text{UTXO}} = \text{SHA256} (\text{SHA256}(\text{CTxOut.scriptPubKey}));
\]

This approach will work for all the script types including Pay-to-PubkeyHash (P2PKH), Pay-to-ScriptHash (P2SH), Pay-to-multisig (P2MS) or Pay-to-Pubkey (P2PK) outputs.

At the split event, the Bitcoin client will calculate 256-bit hash values for all UTXOs and decide in which chainstate database a UTXO will end up. At that point, the Bitcoin client will create \texttt{chainstate0} and \texttt{chainstate1} databases and remove the original \texttt{chainstate.db}, in case of a two-way split.

From the split event onward, sub-chains start to add their own blocks. After the first valid blocks are added, two sub-chains behave just like mini-Bitcoin networks independent of each other.

C. Dividing Mempool

At the split event, Bitcoin memory pool will flush, create \texttt{mempool0} and \texttt{mempool1} and remove the original mempool. Starting with the next block, only transactions that belong
to a specific sub-chain will be considered valid. Each sub-chain will transmit its unconfirmed transactions to a different mempool, and they will be picked up by miners separately for each sub-chain.

D. After the Split

Dividing a UTXO set and mempool will create mini-Bitcoins effectively (Figure 1). These mini-Bitcoins will function independently by conforming the rules below:

- every full node can divide UTXO set and mempool, and keep track of the divided sets independently based on the guidelines (as additional consensus parameters coded in the Bitcoin client).
- mempools will detect (using the new consensus parameters) and won’t accept mixed transactions, e.g. users can’t mix up UTXOs belonging to different sub-chains in a single transaction.
- the UTXO set will be tracked in multiple databases, on a per-subchain basis.

A side effect of this is the continued presence of regular Bitcoin addresses in all sub-chains. However, there may be a different number of UTXOs attached to those accounts on every sub-chain. The effects of such an event on mining and network organization will be elaborated on in the following sections.

IV. Mining

Bitcoin uses Proof-of-Work (PoW) consensus that utilizes double SHA256() as its hash function. Every new block should contain the hash of the previous block in its header and the checksum of its own header should be lower than the 256-bit difficulty value that is updated every week. The proposed scaling solution will not attempt to change the hash function or propose a new consensus function. The aim of this paper is to adapt the current mining approach to a multi-chain setup.

As described in previous section, after the split event there are multiple sub-chains that can operate independently. This means every sub-chain may separately mine their own blocks and append to the blockchain, assuming all mining parties are honest. However, to keep the Bitcoin system robust and trustless in a multi-chain setup, some form of super-mining should be enforced, instead of making assumptions about the honesty of the other parties. Otherwise, miners with great computing power will jump through sub-chains depending on the difficulty values, which will make the system more vulnerable to instantaneous attacks [10].

A. Eigenchain

In order to keep to mining power in check, adding a new block should happen atomically across all sub-chains. This means the block count will be the same across all sub-chains, all the time. However, to verify the newly added blocks and detect double-spend attacks, there should be a separate blockchain that keeps track of the all the blocks added to their respective sub-chains. This new blockchain that stores the block header hashes of sub-chain blocks is called eigenchain.

The mining will works as follows:
1) Miners have to listen to all sub-chains and pick transactions from all mempools.
2) Miners have to mine a block for every sub-chain (the difficulty levels will be adjusted after the split events).
3) Miners have to create a new eigenchain block by using sub-chain block headers. The difficulty of the eigenchain block should be higher than the sub-chain blocks, almost having the same difficulty in Bitcoin network at the time of the split event.

In Bitcoin, a newly mined block is serialized [11] and transmitted using the block message [12]. In this proposal, however, the block message will transmit a single serialized block similar to Bitcoin but that block will contain both the new eigenchain block and all the other sub-chain blocks.

Although the proposed approach seems like a glorified block size increase at the moment, the changes in the network organization and introduction of the half node will show the benefits of the approach. For an explicit comparison, refer to the Section VII: "Bitcoin Proposal Comparison." The network organization will be discussed in the following section.

V. Network

The Bitcoin network consists of multiple types of peers: miners, full nodes and lightweight nodes. Miners are the peers that create and transmit new blocks to the network, full nodes are the verifiers that store the complete blockchain and lightweight nodes are the relatively weak ones that use Simple Payment Verification (SPV) to only verify particular transactions [14].

A. Full Nodes

In the regular Bitcoin network, full nodes store the complete blockchain and execute block and transaction verifications all the time to keep the system secure. Similarly, in our proposal, full nodes will keep in sync with all sub-chains plus the eigenchain, therefore it will be able to verify a specific sub-chain in itself and cross-reference it with the eigenchain. Miners and full nodes are connected in a way similar to the current Bitcoin network formation and new block messages are only sent to full nodes. Full nodes will verify and update the newly mined blocks, and will then re-transmit the sub-chain blocks (a serialized eigenchain block and appended sub-chain block) to the relevant networks formed by sub-chain nodes. In short, full nodes are interconnected to full nodes and half nodes. Not all messages are sent to sub-chain networks, however. Only the relevant ones are propagated to minimize the bandwidth requirements.

B. Half Nodes

With the proposed scheme, an additional type of node called half node is added to the system. Half nodes keep track of one sub-chain and the eigenchain. A half node is able to verify both the tracked sub-chain and eigenchain blocks by using block header hashes and is able to cross-check and validate sub-chain blocks using information contained in eigenchain. Half nodes
TABLE I
SPLIT-Scale COMPARISON: MINER PERSPECTIVE

<table>
<thead>
<tr>
<th>Miner Features</th>
<th>Bitcoin (SegWit)</th>
<th>SegWit2x</th>
<th>Bitcoin-NG</th>
<th>Split-Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale Factor</td>
<td>1x</td>
<td>2x</td>
<td>60x</td>
<td>Nx (scales exponentially with split count)</td>
</tr>
<tr>
<td>Block Count</td>
<td>mine one block</td>
<td>mine one block</td>
<td>mine one key block plus microblocks (every 10s)</td>
<td>mine one block on all sub-chains plus one eigenchain block</td>
</tr>
<tr>
<td>Block Size</td>
<td>~1MB on average</td>
<td>~2MB on average</td>
<td>same as Bitcoin (SegWit)</td>
<td>same as Bitcoin (SegWit)</td>
</tr>
<tr>
<td>Transaction Fees</td>
<td>from one block</td>
<td>from one block</td>
<td>from all key and microblocks</td>
<td>from all sub-chain blocks</td>
</tr>
</tbody>
</table>

TABLE II
SPLIT-Scale COMPARISON: NODE PERSPECTIVE

<table>
<thead>
<tr>
<th>Node Requirements</th>
<th>Bitcoin (SegWit)</th>
<th>SegWit2x</th>
<th>Bitcoin-NG</th>
<th>Split-Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storage Requirements</td>
<td>whole blockchain</td>
<td>whole blockchain</td>
<td>whole blockchain</td>
<td>full nodes store the whole blockchain half nodes store only one sub-chain and eigenchain</td>
</tr>
<tr>
<td>Bandwidth Requirements</td>
<td>at least ~700Kb [13]</td>
<td>at least ~1.4Mb</td>
<td>full node bandwidth requirements increase linearly with scaling factor (60x)</td>
<td>full node bandwidth requirements increase linearly with scaling factor (Nx) half node requirements will be similar to Bitcoin (SegWit)</td>
</tr>
</tbody>
</table>

only keep track of one mempool and one chainstate database (UTXO set) depending on which sub-chain they select. In addition, half nodes do not get new block messages. New blocks targeting sub-chains are transmitted by using a new type of message: a block-n message, which contains only the serialized eigenchain and sub-chain block. This way both the storage and bandwidth requirements of half node will be significantly lower compared to full nodes.

VI. TRANSACTIONS

After the split event, all the UTXOs of a specific script-PubKey will be accumulated in one sub-chain. Basically, users will be able to create transactions using only UTXOs from a specific sub-chain and be able to transact without knowing the remaining sub-chains. However, in time, users will receive payments from multiple parties in different sub-chains. Therefore, total account balance of a user will more or less reside in multiple sub-chains with different UTXOs attached to it. If a user wants to spend more than the total amount of Bitcoin in one of his sub-chains then multiple transactions should be made.

A. Hashed Time-Lock Contract (HTLC)

Hashed Time-Lock Contract, or HTLC in short, is defined as: 'a class of payments that use hashlocks and timelocks to require that the receiver of a payment either acknowledge receiving the payment prior to a deadline by generating cryptographic proof of payment or forfeit the ability to claim the payment, returning it to the payer' [15]. Lightning Networks use HTLC to be able to construct secure transfers using a network of channels across multiple hops to the final destination [16].

In the proposed system, HTLCs are used to ensure the atomicity of the payment, even the payment consists of multiple transactions on multiple sub-chains. Assuming that the sender does not have enough balance on one sub-chain to cover the complete payment, then the sender has to create multiple transactions on multiple sub-chains respectively. The receiver may claim each transaction on a different sub-chain, but it is preferred to finalize the payment in a single step. In such cases, senders (therefore the underlying Bitcoin wallet implementations) should utilize HTLCs to ensure atomicity.

The sending process should be as follows:

1) the sender creates random data.
2) the hash of that random data is calculated.
3) the hash value is added to all transactions (scriptPubKey) and transactions are sent on their respective sub-chains.
4) after all the transactions are mined, the total payment amount may be claimed by the receiver, after complete random data is shared by the sender.
5) if any of the transactions fails in a predefined time interval, funds may be claimed by the sender again.

B. Eigentransactions

An eigentransaction is a failsafe mechanism which may be added to the system to make fund transfer possible between the sub-chains. However, these transactions are special and limited to sending funds only between the same addresses in multiple sub-chains, so the private key of sending and claiming address should be the same. This is to provide an easy way for transferring the total account balance into a single sub-chain.

Eigentransactions should have a separate global pool called the eigenpool similar to the current Bitcoin mempool, and eigentransactions are mined and included into the eigenchain.
This way all sub-chains will be able to track fund transfers of the same account across sub-chains and will be able to add the UTXO (if sent to that specific sub-chain) to their balance. With the addition of eigentransactions, the block size of the eigenchain will be increased. However, activation of this feature can be easily controlled, even enabled/disabled between certain block numbers.

VII. BITCOIN PROPOSAL COMPARISON

Two forms of decentralization are at the heart of the Bitcoin scaling debate. The first form is mining decentralization, which is the problem of accumulation of high hash rates at the hands of a limited number of mining cartels. The second one is decentralization of the network, which is the decreasing amount of full nodes due the increasing bandwidth requirements. Our proposal aims to scale the Bitcoin network without decreasing network decentralization. In Table I and Table II, the split-scale proposal is compared to the other valuable on-chain scaling proposals in terms of miners and network node features.

Split-scale provides a framework for scaling and gives the opportunity to scale exponentially with every split event. For the miner, our proposal will provide better economic incentives, because although the block reward is the same, the transaction fees will be collected from all sub-chain blocks. As a result, transaction fee gains for miners will even surpass Bitcoin-NG at the sixth split event (64 sub-chains) (Table I).

Finally, our solution is clearly efficient in terms of bandwidth and storage requirements (Table II). In all the other proposals transaction throughput increase (scaling) is directly translated to bandwidth and storage increase for Bitcoin nodes. As a result, due to the increasing requirements, the number of Bitcoin full nodes will decrease and network decentralization will suffer in all the other proposals. Split-scale introduces a new kind of Bitcoin node which is called 'half-node' that eliminates these restrictions and provides capability to run a node tracking only one sub-chain and eigenchain.

VIII. CONCLUSION

Scalability is important for expanding adoption of Bitcoin. In this paper, we address the scalability problem by partitioning the UTXO space, therefore splitting the Bitcoin blockchain into multiple sub-chains. Our approach facilitates a block creation increase due to the mining taking place on all sub-chains and it proposes a way to still maintain nodes operating in low-bandwidth conditions. Compared with prominent Bitcoin scaling proposals, "split-scale" offers scalability while preserving network decentralization.

REFERENCES

Bringing Transparency in Capital Markets using Enterprise DLT

Abstract—Blockchain, the technology underpinning Bitcoin has attracted capital market participants globally with its potential to bring greater transparency, manage risk efficiently, and enable greater operational efficiency, disintermediation, and instantaneous trade settlement thereby bringing operational and business transformation. Multiple proof of concepts, pilots and solutions were developed with Blockchain 1.0 technology using Ethereum & Bitcoin for solving capital markets, which while yielding encouraging results also highlighted the severe limitations of leveraging public Blockchain networks for enterprise use. This resulted in advent of Enterprise DLT platforms coming to existence including Consortium led flavors such as Hyperledger Fabric, Sawtooth, Enterprise Ethereum Alliance’s Quorum and Proprietary technology such as Digital Asset & R3 Corda. Our paper focuses on how Enterprise DLT can bring value to Capital Market industry in certain niche areas specifically where greater transparency is needed and risk management is of extreme importance. This paper will describe real-life DLT solutions developed by Broadridge Financial Solutions.

I. INTRODUCTION

Broadridge Financial Solutions is a global leader in providing technology enabled solutions to global markets. Broadridge processes $5+ Trillion in equity and fixed income trades daily and processes 80% of proxy meetings in North America and serves 200+ markets [1]. Broadridge India is GIC of Broadridge globally and acts as center of excellence for emerging technologies including DLT. Broadridge’s Blockchain Center of Excellence has been operational since 2014 and has built multiple applications using various DLT platforms including Quorum, Fabric, DAH, R3 Corda etc. Broadridge launched a DLT solution for Global proxy solution for Spain in 2018[2][3] and has further extended the solution as a pilot in APAC region[4]. Broadridge has also developed a DLT based Fixed income repo solution for North American markets which provides ability for firms to trade in Repo contracts using tokenization[5].
II. BENEFITS OF DLT TO CAPITAL MARKETS

A. Abbreviations and Acronyms
- DLT – Distributed Ledger Technology
- APAC – Asia Pacific
- GIC – Global Insourcing Center
- PBFT – Practical Byzantine Fault Tolerance
- IDM – Identity Management Systems
- OTC – Over the Counter markets

B. Key Features of Enterprise DLT Systems
- Decentralization of networks is achieved in DLT through availability of multiple nodes which shared a “distributed ledger” eliminating need for an “intermediary” to maintain the ledger. Enterprise DLT solutions are often Quasi de-centralized with certain nodes acting as “validator” or “notary” nodes
- Simpler consensus mechanisms relative to public blockchain networks with implementation of variations of Byzantine Fault Tolerance algorithms (PBFT, Istanbul etc.) , Raft or Paxos algorithms
- Implementation of privacy and confidentiality through encrypted payload transmissions and private transactions in Quorum[6], Private Smart contracts[7] and Store Channels in Hyperledger Fabric[8], Rights and Obligations in Digital Asset platform and Flows in R3 Corda which entitle only participants with rights to view and act on data that they are entitled to
- Implementation of Enterprise Identity Management solutions with integration to existing IDM systems and X.509 standard certificates
- Support of off-chain data stores to ensure DLT systems can scale to support storage and speed requirements of enterprises in financial services industry
- Consensus driven Smart Contracts that could automate complex contractual obligations and bring high level of workflow automation without need of human interaction and interact with external world through Smart oracles

C. Benefits of DLT to Capital Markets
- More Efficient settlement of transactions and processing as all market participants see same data as long as they are entitled to and updates are circulated instantaneously. Widely mentioned as Single Source of Truth
- Increased transparency, with all participants receiving a unified view of real-time golden source data. Shared across all nodes / participants in the network through means of “Global State”
- Significant cost savings and operational efficiency by reduction in reconciliation systems and operations as all participants see same data reducing need for reconciliation
- Risk Mitigation and management in bilateral and OTC markets as Smart contracts enable capture of complex derivative contracts terms, eligibility and processing workflow automation thereby eliminating manual processes and errors
- Digitization of Know-Your-Customer (KYC) and Anti-Money Laundering (AML) in Capital markets industry eliminates disparate and fragmented capture of Shareholder information
- Streamlining of Security Issuance directly on DLT through concept of Security Tokens and Asset backed tokens which will accelerate trading of real world assets in DLT based networks
- Improved Corporate governance activities by having DLT driven Proxy voting which gives an end to end transparency from AGM being announced to shareholder positions and voting details captured real time
- Instantaneous securities vs cash settlement in DVP fashion through security tokens being settled against digital currency or crypto or fiat currency backed tokens
- Elimination of discrepancies and disputes related to securities reference data by having golden copy of information
- Payments of dividends, income distribution through Digital / Crypto and Fiat currency backed tokens
- Fungible usage of assets held in DLT as collateral enabling faster collateral management and handling of margin money

Many pilots have already been conducted in capital markets industry related to benefits described [2][4][5]. Use of DLT in Capital markets can be broadly categorized in to:
- Shared Record Keeping
- Enhanced smart value transfer of assets
- Streamlining automation through smart contracts

III. BROADRIDGE’S DLT SOLUTIONS

Broadridge’s DLT strategy is focused on bringing “greater transparency” and improved “risk management” to capital markets industry by co-innovating with clients in collaborative model. Broadridge is agnostic to underlying enterprise DLT platforms. Broadridge has built multiple DLT solutions but this paper will focus on:
- Enhancing greater transparency in global proxy markets using DLT
- Enabling bilateral repo contracts to be traded on a DLT market place through tokenization

A. When to use DLT Solution

The biggest benefits of “DLT” Solutions is when it brings a “network effect” and involve multiple participants interacting with each other in a complex workflow and exchanging data.
We recommend that DLT solution to be considered in Capital markets if it involves following conditions:

1. Network of participants who have intense interactions with each other
2. A monopolistic intermediary is currently servicing the market with observed limitations
3. High level of business risks, operational and cost efficiency, manual intense processing
4. Need for large levels of reconciliation to handle errors and disputes
5. There is a need for security, privacy and confidentiality
6. Need for real time transparency

IV. BROADRIDGE’S DLT BASED GLOBAL PROXY SOLUTION

A. Overview

Broadridge’s Global proxy solution provides ability for Institutional investors to cast their vote electronically across 150+ markets when corporate issuers hold their annual or extraordinary general meetings. Broadridge provides services from setting up the meetings, notifying the shareholders and collecting the votes. Post collection, Broadridge transmits the votes to the local market custodians and Issuer agents.

B. Problem Statement

In today’s world, when an Issuer holds their annual meeting or extraordinary general meeting in a country X, meeting information is shared globally with all respective shareholders of issuer. An institutional shareholder, say resident in Country Y, submits their votes through a chain of custodial intermediaries located in country X and Y. Votes are finally registered with issuer or transfer agent maintaining recordkeeping aspects of issuer. The process is not real time, prone to errors and discrepancies, operationally inefficient, has reconciliation overhead and in many markets there is no vote confirmation.

- Lack of transparency in end to end processing of proxy markets, with Shareholder unsure if their votes have been registered and counted, as there is no vote confirmation
- Operationally inefficient and error prone as chain of intermediaries across proxy lifecycle could have different records related to shareholders at any point of time
- Due to time-zone difference there is a shortened window available for shareholders to vote
- Over-voting or under-voting could result if shareholder records are not consistent across chain of intermediaries
- Reputational risk to all involved if errors in proxy processes

C. Goals of DLT based proxy solution

We have built a DLT based proxy solution that brings together various market participants including issuer, issuer agent, custodians, institutional shareholders and other intermediaries in a DLT network with following objectives:

- Elimination of discrepancies in meeting and agenda information by having details captured directly in DLT
- Real time viewing and correction of shareholder entitlements and positions by providing ability for custodians to set up positional information directly in DLT
- Enable direct voting by shareholders from multichannel (Web, Mobile etc.) directly into DLT
- Real time meeting quorum tracking for Issuers

Figure 1 - Considerations for Choosing Enterprise DLT
• Elimination or shareholder entitlements and voting data discrepancies between various market participants
• Real time voting tracking
• Vote confirmation and publishing of meeting results directly in DLT
• DLT super-highway network enabling institutional shareholders to vote on DLT and Non-DLT markets

D. Key Design Considerations

We have adopted following key design considerations when leveraging DLT technology for the global proxy solution. These design consideration are important to ensure legal and regulatory compliance

• Each market participant should be able to view only data they are entitled to in compliance with legal, commercial and regulatory rules
• Shareholder identification should ensure privacy and confidentiality of data and all PII data should be protected under the laws of jurisdiction including European Union’s General Data Protection Regulation.
• Ability for participants to join the DLT network as Direct participants running nodes (or) indirect participant through a client application
• Integrate with existing infrastructure
• DLT solution should be generic enough to support market specific rules and regulations through Smart Contracts
• Ability for regulators to view data related to their Jurisdiction without any data viewing restrictions
• DLT solution should be in a permissioned network with only trusted participants able to join the network
• Business Network Operator should not have any access to cryptographic identifies of network participants

E. DLT Proxy Solution

1) DLT Platform
The chosen DLT platform is an enterprise adoption of Ethereum known as “Quorum”. Smart contracts are written in Solidity[9]. DLT platform provides JSON RPC APIs that consume and publish JSON based messaging content to integrate with external systems

2) Entity and Identity Management
Market participants are set up as entities mapping to their legal identification and provided with multiple level of access depending on their access rights. Each entity is provided with appropriate Quorum credentials to ensure rightful access to node data. Entities are provided read/write or read only access depending on their business rules. Entities are allowed to setup their nodes in on-premise or hosted or cloud infrastructure if directly participating. Indirect participants access DLT through Javascript based Client applications. Nodes could be set up with view only or read/write privileges.

Only registered and approved users from entity firms have access to DLT. Users are setup through a US Patented (US 9967238, Dated May 8 2018)[10] cryptographically secure process developed by Broadridge DLT team. User credentials can be revoked or altered when a real world event impacts user & entity relationship happen such as user leaving firm, transfer to another department etc. Real world user identity is mapped to Quorum identity, verified and validated, a cryptographically secure key pair generated without Broadridge or any other market participant having any information related to keys. User identity keys are not stored in any central database

3) Meeting and Agenda Setup

Meeting and agenda information is considered as “Public” data and is available to everyone in DLT network. Meeting and agenda information is setup by either issuer or issuer agent. Meeting and agenda information is entered directly to DLT (or) ingested from any other external source and digitally signed. Smart contracts ensure that only issuer or any entity authorized by issuer such as Issuer agent are allowed to setup meeting information. Data is available from DLT to any downstream system through API mechanisms

4) Shareholder Position and Entitlements

Shareholder position and entitlements refer to position or shares held by shareholder. It is a dynamic data and can change often. It is also private and confidential information. Data should be visible only to shareholder and custodian holding assets on their behalf, i.e. – If Shareholder X serviced by Custodian C, then only participants X and C can view shareholder’s position information and entitlements. Only X and C can view shareholder identification and no one in the network can view this information. This poses an interesting DLT requirement that this information should be stored in DLT and be part of consensus but data should not be seen by any other participants except X and C. We have achieved this in Quorum by using “Private Transactions” and private contract store. Data is stored in global ledger in an encrypted manner and actual data is stored in only X and C’s nodes. Rest of nodes has only encrypted data, decryption keys are present only with X and C. Consensus is performed using Istanbul algorithm on encrypted data. Position and shareholder information is ingested from various systems in JSON format and converted to Solidity smart contracts. Downstream systems can request data using APIs. One of the unique features we have implemented and patented as part of US Patent 9967238[10] is ability to encrypt at “data attribute” level to ensure highest level of PII protection. Data seen by X and C differ based on their level of access. Other participants in the network such as local market custodians and
issuer agents can see positions and entitlements without Shareholder identity being revealed to them. Data attribute level encryption is not available in Quorum off the shelf.

5) Shareholder Voting

Shareholders can vote directly into DLT and establish their identity using cryptogenic keys. **Vote data is digitally signed by shareholder using his/her private key.** Vote data is also private data and can be seen only by shareholder, custodian servicing shareholder and other participants are able to use vote data for tabulation without knowing identity of vote.

6) Vote Tabulation

Vote tabulation is executed in real-time on DLT by a smart contract. As votes are lodged, a vote tabulator smart contact tallies votes and tabulates outcomes in real-time. Voting data is private and shared only with issuer and issuer agent in real time. Results are published publicly to all market participants and external systems only after completion of meeting. Issuer can also track vote quorum without knowing underlying shareholder identity in real time during the time meeting is live.

7) Vote Confirmation

Shareholder receives confirmation that his vote was registered during meeting by a private transaction in DLT. This vote confirmation is not visible to any other market participants.

8) Technology Stack

**DLT Platform:** Quorum  
**Smart Contracts:** Solidity  
**Server Side Runtime:** Node JS  
**Middleware:** Express JS  
**Offchain Datastore:** Couchbase  
**Data Digest Algorithm:** SHA3-KECCAK  
**Message Integrity:** ECDSA  
**Key Exchange:** ECDH  
**Encryption:** AES-256

![Figure 2 – High Level Architecture](image)

9) Benefits Observed

- Increased shareholder participation with real time quorum tracking enabling issuers to send reminders
- Lengthened voting window due to elimination of individual voting windows by each intermediary
- Ease of reconciliation with local market systems
V. BILATERAL REPO USING TOKENIZATION

A. Overview

A repo transaction in fixed income markets is a short-term lending of a bond or debit instrument from a lender to a borrower. In capital markets repos can be short-term with fixed term date (or) a rolling transaction. Repo transaction between two parties is conducted either through a central party or in a bilateral mode. Bilateral mode carries counterparty risk, operational risk and settlement risk. It is also operationally inefficient with manually intensive processes and reconciliation costs.

B. Problem Statement

A bilateral repo transaction between a Buy Side institution (Asset Managers) and a Sell Side institution (Broker-Dealers) is typically non-centrally cleared. Due to lack of a central counterparty, risks associated with bilateral repo transactions are quite high. Added to risks are associated costs involved with Custodial support and reconciliation.

- Operationally inefficient and manually intensive processes as multiple systems are needed to maintain repo transaction details and reconciliation needed on a daily basis
- Custodial delivery costs for clearing and settlement
- Counterparty, Operational and Settlement risks if a counterparty was to have balance sheet problems
- Expensive process for firms to move collateral cross border between two different geographical entities
- Disputes in Collateral valuation and management process

C. Goals of the DLT solution

- Mitigate Operational, Settlement and Counterparty Risk
- Improve operational efficiency, better workflow automation and reduce costs
- Compliance with regulations related to Repo transaction processing, repo life cycle management and margin
- Legal and regulatory compliance related to Security tokens
- Ensure actual bond represented by security token is held at a trusted place
- Ensure data privacy and confidentiality about repo transaction

D. Key Design Considerations

We have adopted following key considerations when designing DLT solution:

- Repo transaction details should be visible only to bilateral parties and their custodians involved within the network
- Security Token representation of actual bond should be fungible
- Security Token should be backed by a real world bond held at a trusted party
- Pricing of security token should be performed through a smart oracle fetching prices from an agreed upon data vendor
- Collateral eligibility rules for Repo transaction should be captured in DLT
- Security token settlement should happen post cash settlement of transaction

E. DLT Repo Solution

1) DLT Platform
The chosen DLT platform is Hyperledger Fabric. Smart contracts are written in Go and Node.js. DLT platform provides RESTful APIs that consume and publish JSON based messaging content to integrate with external systems.

2) Entity and Identity Management
The market participants are set up as entities mapping to their legal identification and provided with multiple levels of access depending on their access rights. Each entity is provided with appropriate Fabric Certificates to ensure rightful access to node data. Entities are provided read/write or read only access depending on their business rules. Entities are allowed to set up their nodes in on-premise or hosted or cloud infrastructure if directly participating. Indirect participants access DLT through Java based Client application nodes could be set up as read/write nodes or read only nodes.

Only registered and approved users from entity firms have access to DLT. Users are set up through US Patented (US 9967238, Dated May 8 2018)[10] cryptographically secure process developed by Broadridge DLT team. User credentials can be revoked or altered when a real world event impacts user & entity relationship happen such as user leaving firm, transfer to another department etc. Real world user identity is mapped to Hyperledger Fabric identity, verified and validated, a cryptographically secure key pair generated without Broadridge or any other market participant having any information related to keys. User identity keys are not stored in any central database.
3) **Security Tokenization**

The security token is a representation of actual bond held by one of bilateral trading parties. We have developed a smart contract that tokenizes asset held by trading party into security token. Token comes into existence when a trusted party holding asset of trading party agrees to tokenization. Token thus created is fungible and is traded on DLT network. Redemption of token happens through smart contract.

4) **Repo Contract Capture**

**Bilateral counterparties say A and B** pre-agree eligible tokens that can be traded. This is captured as a collateral eligibility rules smart contract. When an actual repo is traded between parties, repo smart contract checks with eligibility smart contract to ensure only allowed tokens are traded. It further checks that lending party is in possession of tokens and underlying asset with an actual trusted party – **Say Custodian C.** Once smart contract verifies all rules, repo contract is captured and obligations generated against involved parties. Token is now auto settled in DLT post an event indicating that cash settlement has happened between the trading parties. Cash settlement happens outside of DLT.

5) **Repo Mid Life Events Processing**

Smart contracts automate repo lifecycle management by executing following functionality

- Collateral Valuation
- Collateral Substitution
- Margin Call Management
- Repo Contract Amendment
- Repo Contact Termination

6) **Technology Stack**

- **DLT Platform:** Hyperledger Fabric
- **Smart Contracts:** GoLang
- **Server Runtime:** Node JS
- **Middleware:** Express JS
- **Offchain datastore:** Couchbase
- **Certificate Management:** Fabric Proprietary
- **Private Contract Store:** CouchDB

7) **Technology Architecture**

8) **Benefits Observed**

- Operational Efficiency and Cost Reduction in Clearing of Repo Transactions
- Instantaneous settlement of Repo Transactions through Tokenization
- Cost Savings by avoiding deliveries of actual collateral to counterparty
VI. CONCLUSION

We have conducted 10+ proof of concepts and pilots over the last 4 years using Enterprise DLT solutions in Capital markets including the 2 solutions that have been described in the document. We have additionally explored and continue to explore DLT in the areas of trade finance, securities reference data, equities settlement etc. on varied DLT platforms.

DLT platforms continue to mature and bring multiple stated benefits including risk mitigation, operational efficiencies, cost and capital reduction, business and operational model transformation. We also observe significant focus in DLT technologies to integrate with existing ecosystem to build a “network effect”. The DLT industry is working towards scalability & Interoperability that is desired in capital markets.

We consider following challenges to be addressed if DLT is to be able to achieve large scale adoption in Capital Markets

A. Challenges in Enterprise readiness

- DLT Systems suffer from relative lack of speed and scale compared to existing technologies
- DLT Systems have not yet achieved integration with existing IDMs, Production monitoring and logging instrumentation etc.
- Lack of enterprise scale DR/BCP infrastructure
- Lack of Interoperability between multiple flavors of DLT solutions

B. Non availability of large scale talent

Most of smart contracts are written in languages which are not widely used in capital markets industry

C. Business Model Transformation impacts incumbents

Embracing emerging technologies which have potential to disintermediate incumbents results in challenges for industry participants to collaborate leading to projects not moving forward to large scale adoption

D. Application of Enterprise DLT in Niche Areas

The Blockchain/DLT hype has created unrealistic expectations in applying DLT for all solutions. We recommend judicious use of DLT in niche areas where there is an existing network and current challenges are hard to overcome with prevalent technologies

E. DLT is still in infancy

Based on multiple industry surveys, we anticipate that enterprise DLT adoption in mainstream capital markets is still 5-10 years away. Niche solutions will go live in DLT in next 2-3 years

F. DLT does not always mean security

DLT solutions do not always mean solution is extremely secure. We recommend existing information security policies to be adopted and leveraged for maximum security. We also recommend exploration pioneering technology components like Intel’s Software Guard Extensions (SGX)[11], Zero Knowledge Proofs. Security can currently be enhanced with Air gapped Enclave computing.

G. DLT Solutions testing and Quality Assurance is not always easy

Testing tools and techniques for DLT solutions need a mindset shift. Testing network solutions where ledger is distributed and data is available only to trusted participants mean one has to rethink test strategies

VII. REFERENCES

[1] https://www.broadridge.com/about/
[2] Santander and Broadridge complete a first practical use of blockchain for investor voting at an annual general meeting
[4] ICJ and Broadridge execute the first blockchain based interoperable proxy voting process in Japan
[5] Broadridge successfully completes pilot of blockchain based bilateral repo solution
[6] Quorum by JP Morgan
https://www.jpmorgan.com/global/Quorum
https://github.com/jpmorganchase/quotum
[7] Constellation – Peer to peer encrypted message exchange
https://github.com/jpmorganchase/constellation
[8] Hyperledger Fabric
https://github.com/hyperledger/fabric
[9] Solidity – Documentation
https://solidity.readthedocs.io/
http://patft1.uspto.gov/netacgi/nph-Parser?patentnumber=9967238
https://patents.google.com/patent/US9967238B1
[11] Intel’s Software Guard Extension (SGX)
https://software.intel.com/en-us/sgx/academic-research

Broadridge Financial Solutions India Pvt. Ltd
Know Your Customer - Decentralized Secure Sharing Protocol on Quorum

Chainworks Digital LLP
Mumbai 400076 Maharashtra India
Email: admin@chainworks.io, https://www.chainworks.io
Demo Url: https://youtu.be/n4Zdz_l8Sc0

Abstract—The advent of blockchains ushered in a new paradigm of secure interoperability between different organizations in a given industry or even across industries. Several start-ups are working diligently to discover and implement new use-cases in various sectors. One of the challenges faced by the implementers in this nascent and fast-evolving field is the absence of standard protocols for common use-cases. In this paper, we propose a protocol to manage multiple KYC related functionalities on Quorum blockchain. The protocol is abstracted from any specific industry so that it can be utilized in any company where KYC sharing is carried out. We have developed a prototype of the system for shipping industry and it is being experimented with industry in real-time.

Index Terms—Blockchain, KYC, Ethereum, Quorum, Smart Contract.

I. INTRODUCTION

The invention of the internet and the world wide web is arguably one of a handful of man-made inventions which led to a paradigm shift in the way many functions are conducted on a societal level across multiple industries on all scales. Even though, on a functional level, the essence of the technology is not much more than an efficient communication mechanism, the use-cases built around it currently are responsible for the smooth functioning and existence of multiple businesses and even governments. In recent times, one such influential technology which is prophesied to create yet another paradigm shift centered around the simple idea of trust is blockchain and the applications built on it. The invention of Bitcoin in 2009 and its successful implementation for the past nine years without any security compromise serve as a proof-of-concept for the viability of the technology for other industries.

From the standpoint of modern technology standards, the current implementations of blockchain applications can be perceived to be naive and counterproductive - not dissimilar to the simple communication facilitated by the early internet - but it can be argued that it is only after blockchain first gets implementation across enterprises and industries, that the actual innovations and benefits of the trust layer would begin to appear. Continuing the contrast with the internet, one innovation that played a huge role in internet’s proliferation from the labs of Department of Defense to the common populace is the introduction of standards and protocols like TCP/IP. Similarly, for the widespread acceptance of blockchain, there is a necessity of standardization of different procedures.

There are several efforts from the academic sector as well as from the industry to come up with a viable model for standardizing operations on blockchain in multiple domains. Medrec is one such proposal by the MIT Media Lab to facilitate easy sharing of a patient’s health record between different hospitals while guarantying security and the autonomy of the patient over the data. OriginTrail is an industry-first innovation which proposes standard operating procedures for supply chain tracking. To add to the list of proposals, we have been working with an industry partner to devise a way to decentralize the KYC sharing protocol called dKYC - shorthand for decentralized KYC - between two different parties. The protocol is currently based on the Quorum blockchain. Some insights and pointers about the features which can be incorporated in devising the current implementation protocol are obtained from the contemporary research on KYC on the Fabric blockchain. We propose here the details of the same along with a strategy to help accelerate the adoption of the protocol and blockchains in general.

II. QUORUM PLATFORM

Bitcoin, despite being the forebearer of all the new-generation blockchain platforms, has limited applications apart from cryptocurrencies when compared against its alternatives. After the initial implementations of singular purpose cryptocurrencies like Peercoin and Namecoin, the general trend has been towards development of blockchains which can support Turing-complete smartcontracts. Among the public alternatives, the biggest platform in terms of number of D-Apps and developer adoption is Ethereum. Even with the large developer base, the public Ethereum chain is not ideal for industry deployments due to its scalability and privacy challenges. Though these shortcomings are being resolved in the current roadmap, the platform with its existing capabilities and a few modifications can be used to fruitfully setup an industry consortium either as a open private network, or as a permissioned network with private contracts support.

Quorum is a permissioned implementation of Ethereum supporting data privacy. The Quorum architecture is ideal for use-cases which require 1. Consortium networks 2. Private transactions and 3. Small consensus groups.
A. Consortium networks

Due to the default open and public nature of the Ethereum blockchain, there are no assumptions about the identity of users/nodes joining the network. Any individual can own more than one node. Due to this, mapping the details of the nodes to its owners is a non-trivial task. This is not necessarily the case in Enterprise Consortiums where the necessity is to carefully guard the parties that can access the network. To support this requirement, Quorum implements a permission feature where the parties can enter into the consortium only by submitting their node identity (public key) to the existing participants.

B. Private transactions

By default, the details of all transactions submitted to the blockchain is accessible to all the parties in the network. This leads to a data-privacy challenge where sensitive data is exposed to unintended viewers. To mitigate this, Quorum uses an additional privacy layer, called Tessera, to compliment the Ethereum blockchain. The benefits offered by this feature can be illustrated by considering a network consisting of fictional participants - Acme, Globex, Hooli and Initech. Acme can use the private contract feature to deploy a smartcontract private to only three participants - itself, Globex and Initech - and all transactions and data on this private contract would be visible to all the three parties while Hooli would not have access to any of the transactions. This selective privacy is helpful in networks where part of the data can be public knowledge and part has to be maintained confidential.

C. Small consensus group

A consensus algorithm is the way the nodes in the network come to an agreement about the transactions and their order in the blockchain. Choosing Proof of Work as the consensus algorithm in the Bitcoin network is one of its main sources of effectiveness due to its ability to withstand attacks in public networks. But this comes at a cost of slow transaction confirmation time and wasted processing power. Fortunately this level of resiliency is not a necessity in a private network since all the participants are known beforehand, and therefore, the attack surface is smaller. Quorum takes advantage of this and provides two extra consensus options - IBFT and RAFT. IBFT is a consensus mechanism useful in scenarios where the involved parties do not trust each other, whereas RAFT is a faster alternative which is relevant for cases where the involved parties have no concerns of mistrust.

More details about the architecture of Quorum can be found in the project’s wiki [7]. These features, along with other considerations like the platform’s rapid evolution and the economic viability to deploy a node quickly make Quorum a good platform for SMEs.

III. Know Your Customer

Know Your Customer (KYC) procedure is one of the most common and important part of customer verification done by many services during the on-boarding process. Typically, the documents required are government issued identities like the National Id, Business Permits, Trade License etc. In financial institutions, KYC is done to avoid any counterfeit and AML activities; while in an exporting business, the transit service verifies the KYC documents of the exporter to validate authenticity.

In India, to simplify the KYC processes, there are two systems in place: cKYC and eKYC. These systems are described in brief below:

A. cKYC

cKYC is a centralized repository of KYC records of customers in the financial sector with uniform KYC norms for inter-usability of the KYC records across the sector to reduce the burden of getting those verified every time when the customer creates a new relationship with a financial entity [9]. Financial Institutions (FIs) register onto the cKYC platform and the KYC records of the FI’s customers are uploaded to it. This service is maintained by Central Registry of Securitisation Asset Reconstruction and Security Interest of India and is expected to overcome the shortcomings of the fragmented traditional KYC processes done by multiple KYC Registration Agencies (KRAs).

B. eKYC

eKYC is a paperless KYC process, wherein the identity and address of the subscribers are verified electronically through Aadhaar Authentication [10]. To complete an eKYC process, the data is obtained from Aadhaar’s CIDR database which can be done only by special authorized authorities called AUAs and KUAs. The data obtained from an eKYC process is stored in a KRA’s database to ease future uses.

The dKYC protocol tries to build upon the KYC processes handled by KRAs where one set of intermediaries (typically FIs) register onto the platform and the KYC of any customer of those intermediaries is uploaded to the KRA’s database.

IV. Terminology

The dKYC protocol builds up on traditional KYC routines and makes them more secure by complimenting the processes with a few extra steps performed on the blockchain. The definitions used in describing the process are noted below

ENTITY is any party that is a participant of the dKYC platform. In the protocol, there are three types of entities - Regulator, Intermediary and Company. These are expanded upon in further definitions.

REGULATOR is the entity which is responsible to verify the other participants’ KYC documents. In a three party KYC system, while there can be multiple attestors for different participant documents, one entity takes the role of managing the documents and performing other admin related tasks. The entity which is assigned this management task is labelled as a regulator on the platform. In India’s KYC scenario, a KRA plays the role of the platform regulator.
**INTERMEDIARIES** are the entities which provide services to companies and need to check their KYC of those in order to process their services. In a general example, a Bank can be an intermediary which needs to check the KYC of a company to provide services like loan.

**COMPANY** is any participant which wishes to avail the services provided by the intermediaries. This can be either an individual or a non-individual entity. In a normal scenario, this is an enterprise which tries to avail a loan from a banking intermediary.

**PROOF OF EXISTENCE** is a mechanism to demonstrate data integrity and ownership without revealing the actual data. To prove the correctness of a company’s data at a particular point in time, the details are timestamped, arranged in a standard data format like JSON or XML and then hashed with a cryptographically secure algorithm like SHA-256 or RIPEMD. The digest thus created is uploaded to a blockchain. In case of any contention about the data, the proving party can re-construct the data in the chosen format along with the timestamp, hash this re-constructed data, then compare the digest stored in the blockchain with the re-created digest. Since the hash in the blockchain is stored not in a single system, but in a distributed system owned by all the parties in the network, there is no scope to illegitimately modify the data without compromising the network.

**SMART CONTRACTS** are computer protocols intended to digitally facilitate, verify, or enforce negotiations or legal contracts [21]. In the context of Ethereum, these are written in Solidity language and can be used to code the terms of agreement onto the distributed ledger. Access control is one of the major applications of Ethereum’s smartcontracts which dictates the entities which have access to a particular data and function.

**EVENT LOGS** are constructs of the Solidity language which are used to broadcast any sample of data in the blockchain’s logs. All data that has been broadcasted with the event logs is permanently stored on the blockchain and is immutable. A typical use-case of event logs is to record any important changes happening in the smartcontracts.

**BLOCKCHAIN EXPLORER** is a data visualization tool which provides an easy-to-use interface to query the transactions and view the details of the blocks in the blockchain. While there can be many blockchain explorers for a given chain, [1] [11] are popular explorers for the public Bitcoin and Ethereum networks respectively.

V. DECENTRALIZED KYC

The dKYC protocol merges these functionalities provided by the blockchain with the KYC processes to 1. Track company updates, 2. Manage view access permissions, 3. Track view attempts and 4. Manage view access requests

The workflow of the dKYC processes are described in more detail in the sections below.
B. Access Permissions

In addition to the details’ updates tracking, the protocol also manages the intermediaries’ permissions to view a company’s KYC data. This is achieved by the following steps:

- During the signup process, each entity is assigned a unique address to interact with the blockchain.
- The company’s profile has a page which lists all intermediaries registered on the system. Against each intermediary, the intermediary’s access status for the company’s data is shown. If the access status switch is on, it signifies that the intermediary has the access to view the company’s data, and if the status is off, any attempts by the intermediary to view the company’s data is dismissed (This process is described in more detail in the next subsection).
- During the sign up process, a smartcontract is deployed for each company to store the company’s state on the blockchain. This smart contract maintains, among other things, a list of intermediary addresses which are allowed to view the company’s data and the company details. When the company changes any intermediary’s access, the allowed intermediary in the blockchain is modified directly by the company without sending it to the regulator.
- Every time the company makes a modification to the allowed viewers list in the blockchain, the smartcontract emits an event containing the intermediary’s details.

The rights to modify the company’s allowed viewer list is restricted to the company and any attempt from any other account to change the list would be immediately rejected.

C. View Attempts

While, on one end, the company’s allowed intermediaries list is maintained on the blockchain, on the other, the intermediary’s attempts to view company’s data are also recorded on the distributed ledger. This view attempt tracking feature works as below:

- When an intermediary attempts to view a company’s data, it sends the request to the server. For the request to be accepted by the server it is required to have a signature field in its body. This value of this field is a piece of text with a timestamp which is signed by the intermediary’s private key. The piece of text being signed can be anything, but a JSON with the company’s name and the requested time would be a good candidate for this purpose.
- The server, upon receiving the request, verifies the authenticity of the signature, queries the blockchain to check if that particular intermediary’s address is in the allowed list of the requested company’s smartcontract.
- If the intermediary is present in the list, the server responds with the company’s KYC details for the intermediary to see. In the case where the intermediary is not allowed by the company, the server responds with a http 403 not authorized response.
- In the company’s smartcontract, an event is emitted with the address of the requesting intermediary along with the success status every time a request is made.

This feature allows to provably track all requests made by intermediaries to get a company’s KYC data. In the future versions of this process, the KYC documents of the companies can be allowed to be stored in the companies’ private databases with an accompanying server to return the data. The URL of the company server’s authenticated APIs would be stored in its smartcontract and can be retrieved by the intermediary. When the intermediary requests the company’s server for data, the server would rely on the permission status dictated on the blockchain to decide if it needs to revert with the details. This process can be fine tuned further upto the document level by following a ERC-1643 like standard.

D. Access Requests

To facilitate all the features necessary in a KYC sharing process, the protocol also has an option to allow the intermediary request access to the company’s data. Since this could also be a feature which needs transparency, all the access requests are tracked on the blockchain. This is a continuation of the View Attempts feature and gets triggered in the case where the intermediary tries to access the company’s data and is denied. It works as follows:

- When an intermediary tries to view a company’s data, the request with a signature is sent to the server as described
in the *View Attempts* section. If the intermediary’s address is not in the company’s allowed list, the server responds with a http 403 response.

- After an unsuccessful attempt, the intermediary is given an option to request access to the company to view its data by displaying a *Request Access* button.
- Upon clicking the button, a request is sent to the company’s smartcontract which keeps a record of all such requests from all intermediaries.
- When the company logs in into its profile and opens the *Requests* page, all the active requests are populated onto its screen from the blockchain.
- Each request also has *Grant* and *Deny* options against them. If the company decides to grant the request, the work-flow described in the *View Access permissions* gets triggered and the request gets archived. In case the company denies the request, the request gets archived and no further processes are triggered.

As with the other functions, the function to request access is limited to entities with an *Intermediary* role and the function to Grant/Deny Access is limited to *Company* entities.

**E. Access Revocation**

On being granted access, an intermediary can view the granting company’s full profile without restrictions on the number of requests. Each time the intermediary views the company’s data, an event containing the details of the request - like requested time and the view attempt status - is emitted. The intermediary’s access is valid for so long as the view status is not changed by the company in the blockchain. As and when the company needs to revoke the intermediary’s access, it can follow the workflow described in *Access Permissions* section and modify the status from its profile page.

![Diagram of Future decentralized architecture with company maintained database](image)

**VI. NETWORK, PRIVACY AND CONSENSUS**

The dKYC protocol is currently implemented on a Quorum network to take advantage of the platform’s privacy and permissioning features. Since the protocol’s operation does not depend on any particular mode of privacy, it can be implemented with various levels of confidentiality. One possible scheme would be to store the entities’ data and updates in a public contract visible to all the participants in the network, while tracking the view related permissions on a contract private to the regulator, company and the intermediary. In case more privacy is desired, the entity updates would also be stored in a private contract and the platform would play the role of a simple inter-connected private database.

Apart from the privacy policy, another choice to be made on the platform is the selection of the consensus protocol. Since the chosen platform is not an open public network, choosing Proof of Work as the consensus mechanism results in unnecessary wastage of computing resources. Instead, a more consortium friendly consensus alternative like Clique, RAFT, or Istanbul BFT can be chosen. The choice of the implementation can be made based on the nature and parameters of the system like the number of trusted nodes, block time requirements etc [12].

**VII. CKYC AND EKYC VS DKYC**

The introduction of cKYC and eKYC processes have undoubtedly improved the convenience of businesses for doing KYC. These processes are ideal for verifying individuals since typically the consequences of data leakage of an individual are not as critical as data leakage of a company’s information. However, there is another class of KYC in which a central party maintains the data and allows specific requesting entities to get access to the data of the requested entities. In scenarios like this, especially when the damage done upon leaking the KYC data is high, the existing systems cannot provide the necessary security for the registering entities. In these multi-party data access scenarios, implementing dKYC and decentralizing the storage and permissions can provide the additional security that the system demands. By distributing the data storage to the companies’ databases, dKYC limits the damage due to a data compromise since such an event would at most affect one company while the data of the others remain safe. The other major benefit of dKYC is the immutable tracking of all important events in the system. In this system, in case of any dispute, contention about a intermediary accessing a company’s data can be resolved by using the provably immutable view attempt records stored on the blockchain.

**VIII. STATUS AND FUTURE ROLL-OUT PLAN**

The protocol is currently being implemented in the shipping domain in which a managing party handles the KYC processes for shipping line intermediaries and exporting companies. For any export operation, the shipping lines are required to see the KYC of the exporting company. This process is currently being ported onto blockchain and the exporting companies are given the autonomy to manage their KYC viewing permissions. The current implementation uses RAFT as the consensus mechanism and is built on a MERK stack.

To facilitate the gradual adoption of the technology, the managing company currently maintains nodes on behalf of the shipping line and the exporter entities. When more security and trust is desired by the companies, they can host a blockchain node internally and get involved as a consortium member. This process of starting with a central provider maintaining multiple
nodes, then gradual rolling-out the nodes to the participants can be a model for accelerating the adoption of blockchain in other industries as well.

IX. CONCLUSION

The current deployment is operational in initial stages with limited entities registered. Future performance and acceptance of the technology remains to be seen. Upon stabilizing the system, we plan to define and implement metrics to measure different parameters like transactions per second and the network up-time. The usefulness and limitations can be more thoroughly understood once the system gets implemented in other domains as well. Further improvements to make the platform blockchain-independent remains in line.

Apart from the metrics, one other area of improvement is the data storage. The current deployment depends solely on a single admin document database. Future work on decentralizing the storage mechanism into company-hosted databases and other decentralized storage options like a private IPFS cluster would be beneficial to the system. A potential pitfall of the current mechanism is the scenario in which the company refuses to reveal the documents. The authors are currently working on mechanisms to counter this scenario. Some early proposals include staking and slashing protocols, Secure multi-party computation mechanisms [22] and proxy re-encryption schemes [20]. Other interest areas with open questions to solve are standardization of the data format and the possibility of including a consent mechanism to make it GDPR compliant.

We hope that the results of the current deployment and the insights gained on implementing relevant metrics can add value to the best-practices for adoption of the system, and blockchain in general, by enterprises.

REFERENCES

Human Milk Distribution - Blockchain Bolsters Transparency and Traceability

Rushi Kumar Talupuru
Research and Development
Nviera Technologies Pvt Ltd
Bengaluru India
rushi@nviera.com

Kishore V Karlapalem
Operations
Nviera Technologies Pvt Ltd
Bengaluru India
kishore.kv@nviera.com

Ravindhranath Mopparthy
Research and Development
Nviera Technologies Pvt Ltd
Bengaluru India
rv@nviera.com

Ravi Karlapalem
Operations
Nviera Technologies Pvt Ltd
Bengaluru India
ravi.k@nviera.com

Abstract— Traditional Human Milk Bank systems face challenges in maintaining data integrity, data transparency and data security. Hence, there is a lack of trust in traditional distribution system. To address this challenge, we have come up with a blockchain based solution for Human Milk banks.

Keywords— donor, pasteurization, serology, transparency, security

1. INTRODUCTION

Human Milk distribution (HMD) system is a blockchain based application to track the administration of human milk bank. The process of Human milk distribution chain deals with highly sensitive data such as Donor serological clearance, Pasteurization, Microbiology and several other records. Blockchain ensures storage of these records in a secure and transparent manner. It also enables effective end to end traceability.

2. TRADITIONAL HUMAN MILK DISTRIBUTION

A. Necessity of Human Milk Banks (HMB)

Breastfeeding is the best method of infant feeding because human milk continues to be the only milk which is tailor-made and uniquely suited to the human infant. Human Milk Banks cater to the need for pasteurized donor human milk (PDHM) when mother’s own milk is unavailable or insufficient. Human milk is of utmost importance in following cases

- Low birth weight babies, preterm and sick babies

B. Risks Involved in Traditional Human Milk Bank Management

Human milk bank deals with various donors and milk processing stages. There are many stages where risks are involved and some of them are mentioned below.

- Donor Tests – Human milk is a carrier of many serious diseases from mother to child. So tests are conducted periodically for various conditions. Few of which are listed below:
  - HIV 1 & 2
  - Hep B
  - Hep C
  - Syphilis

- Microbiological screening of donor milk is done before and after pasteurization.
  - A bacterial count of 105 CFU/mL or more in raw breast milk can be considered as an indicator of the poor quality of milk.
  - No growth is acceptable in post-pasteurization microbiology cultures. Whole batch of culture positive pasteurized milk should be discarded.

3. PROPOSED SYSTEM – BLOCKCHAIN BASED HUMAN MILK DISTRIBUTION

The proposed system employs distributed ledger, which makes data immutable and transparent. The following are the stages involved in milk bank (MB) process:

A. Donor Registration

In this process, donor details and her eligibility criteria will be captured. The following diagram explains the steps involved in donor registration.

Fig: Donor registration process

B. Donor Milk Collection

The donor collects raw breast milk and stores it either in empty bottles provided by the Milk Bank (MB) or their own containers. The milk is attached with a unique code to track throughout the process.
C. Milk Processing

Raw Milk (Received Raw units) is batched into standard units and is labelled with QR labels before being processed. During pasteurisation, the temperatures are measured and monitored. Post which, the samples are tested and the results are recorded in blockchain database. Below are the stages of milk processing and the appropriated data is captured:

A. Thawing raw milk – The thawing process takes place in three ways: 37° Oscillating Incubator, Refrigerator and Room temperature.

B. Aliquots division - Raw milk batches are divided into aliquots (bottles) of 50 ml, 130ml or 250ml. Each aliquot will be labelled with the respective QR code for traceability.

C. Pasteurisation - This process can be done for a pool of aliquots or single aliquot. During this process, a report will be generated which contains start time, end time and temperature logged periodically. The same is stored into our system through which it can be audited to comply with the suggested safety guidelines.

D. Microbiology control - Pre and Post pasteurisation tests measure the microorganisms which are present in the milk before and after pasteurisation. The same is stored into our system for further monitoring.

E. Aliquot storage - Aliquots which are microbiologically cleared will be stored in the freezer

D. Recipient Registration

Details like hospital number, date of birth, age, gender, gestational age, birth weight will be captured during registration. After registration, milk can be ordered for the baby for immediate consumption or reserve the milk for future consumption. One can verify the donor details, raw milk collection, pasteurization reports and other details by scanning the QR code.

E. Application Flow

4. Benefits of this Application

Milk bank administration desires to keep crucial data of Donors, pasteurisation process, microbiological control tests, making it immutable and transparent to the end user. Blockchain is best suited technology for this use case. The following are the few advantages for various users in the system.

A. Donor can schedule appointments for Donations

B. Serological and various other clearance details are meticulously captured and tracked till the end.

C. Application is developed using HIPAA standards, so that it is suitable for use by milk banks all over the world

D. High degree of confidence in integrity of the donor serological reports, pasteurization parameters, pre and post pasteurisation results, expiry date etc.

5. Samvaadak – Our Blockchain Library

Samvaadak powered by Nviera is a leader based consensus algorithm. It avoids many overheads that are present in traditional leader based algorithms

Traditional Leader based algorithms have many demerits such as:

- Election for Leader: In many scenarios, due to the network latency, the leader node may fail to respond to any of the other nodes in the network causing
unnecessary polling for leader. This causes huge network traffic and congestion.

- DDoS Attacks: Leader based algorithms are vulnerable to large delays due to denial of service attacks on the leader. This may compromise the whole system.

- Single point of failure: In many leader based algorithms, the designated leader node is responsible to conduct voting and commit phases for all the transactions. This may lead to a single point of failure since leader node experiences extensive load from the network.

In Samvaadak the merits are:

- Leader is chosen, not elected: Election causes unnecessary traffic in the network. However, in Samvaadak, the Leader is chosen by the initiator of the transaction. This avoids much traffic, which helps in reduction of network latency and improves performance.

- Leader is chosen in round robin fashion: For each transaction a different node will be chosen as a leader. The leader is chosen in a round robin fashion which helps in distribution of load evenly and minimizes the single point of failures.

- Nodes are tightly coupled: Backend nodes will not accept any other connections expect from their family members (Known nodes). This is achieved by using a handshake protocol among the nodes. Hence the chances for DDoS are reduced. Moreover, since the leader keeps changing for each transaction, it is even harder to target any one particular node for DDoS attack. Also Nodes which are exposed to Internet are will be protect by firewall.

**Consensus Mechanism:**

The nodes are classified in two categories as mentioned below

- **Front-End Nodes:** These nodes receive user requests and responsible for initiating consensus. It will also choose the leader to execute the consensus.

- **Back-End Nodes:** Responsible to perform voting and committing phases for a transaction.

![Components of each miner/nodes](image)

- Front-end node will receive a HTTP request and validates the payload to ensure that the presented data is correct and adequate

- The Front-end node chooses a node from the Backend pool as a leader and submits the request for consensus execution and eventual commit

- The selected Back-end node (also called as the Leader) creates a new Consensus transaction for the request

- Leader sends out messages for Voting on the subject of the request

- The rest of the nodes process the voting request sent by the leader node. Based on the business rules defined, they perform the validations and submit their vote responses back to the leader

- Leader node computes the outcome of the voting and if positive, performs the phase 2 (commit phase) of the operation. This involves creating a new block in the Blockchain

6. **BLOCKCHAIN DATABASE**

Blockchain Database contains threads of transactions called Strands. Each transaction is considered as block and linked using strand ID. Each committed transaction will create a new Block. Every block contains three parts which are listed below

- **Header:** Contains the current block hash, previous block hash & the timestamp

- **Meta:** Contains the Node IDs, which are voted for this transaction to be committed

- **Payload:** This is related to business use case, on which the consensus rules are applied to take decision at the time of voting
Every block is linked to previous block of same strand ID using SHA256.

7. TRANSACTION FLOW

The following are classification of data captured in this solution.

1. Donor Blockchain
2. Milk Bank Blockchain
3. Recipient transactions Blockchain.

A. Donor Blockchain

The above diagram show the data captured in blockchain for Donor registration

1. Donor registration - For every new donor a new strand is created with genesis block carrying the following data
   a. Unique ID
   b. Donor Name
   c. KYA Document Hash – Document stored in file storage and the hash of the document is stored in the block. While retrieving the document is verified against the hash stored in the block.
   d. Date of Birth
   e. Nationality
   f. Email Address
   g. Contact Number

2. Donor serology clearance - A block subsequent to donor registration block will be created. Serology test is repeated every 3 months and a new block will be created and linked to existing chain. The following data is captured in the block.
   a. Date & Place
   b. Lab Information
   c. Blood Test ID
   d. Tests reports – reports are scanned
      i. HIV 1 & 2 result.
      ii. Hep B result
      iii. Hep C result
      iv. Syphilis result
   e. Milk Bank Authority ID (user who authorises the donor)

B. Milk Bank Blockchain

Milk Bank Blockchain involves in capturing data in the following stages.

The above diagram depicts the milk processing stages and data captured in each stage.

1) Raw Milk Collection – Data captured is listed below.
   a) Raw milk ID
   b) Donor Unique ID
   c) Collection Date and Time
   d) Receiving Site
   e) Breast Pump Type
   f) Unit volume
   g) Number of units
   h) Total volume
   i) Unit ID

2) Batch Mixing
   a) Raw Milks : <Array of IDs>
   b) Aliquot Unit ID : <Array of IDs>
   c) Mixing Date
   d) Aliquot Unit volume
   e) Aliquot Unit quantity

3) Pasteurization
   a) Pasteurisation Batch Id
   b) Aliquot ID
   c) Date
   d) Pasteurisation Index
   e) pre pasteurisation result
   f) post pasteurisation result

C. Recipient Blockchain

1. Recipient Registration – Data captured is listed below
9.  REFERENCES

i. HIPPA
   https://en.wikipedia.org/wiki/Health_Insurance_Portability_and_Accountability_Act

8. CONCLUSION

This solution considers all risk factors such as donor test reports, pasteurisation reports, storage of milk information and expiry dates etc., and brings more transparency and integrity by capturing the crucial data in all stages using blockchain. Since the end recipients are infants, quality cannot be compromised. QR code on every dispensed milk bottle can be used to view entire history of the milk, which brings transparency and drives confidence in distribution chain.