Received: 25 March 2022 | Revised: 15 June 2022 | Accepted: 24 June 2022

DOI: 10.1049/ntw2.12041

IET Networks

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH

PharmaChain: A blockchain to ensure counterfeit-free pharmaceutical supply chain

Anand K. Bapatla[1] | **Saraju P. Mohanty**[1] | **Elias Kougianos**[2] | **Deepak Puthal**[3] | **Anusha Bapatla**[4]

[1]Department of Computer Science and Engineering, University of North Texas, Denton, Texas, USA

[2]Department of Electrical Engineering, University of North Texas, Denton, Texas, USA

[3]Electrical Engineering and Computer Science, Khalifa University, Abu Dhabi, UAE

[4]Pharmaceutical Development Sciences, Thermo Fisher Scientific, Greenville, NC, USA

Correspondence

Saraju P. Mohanty, Department of Computer Science and Engineering, University of North Texas, Denton, TX, USA.
Email: saraju.mohanty@unt.edu

Abstract

Globalisation has facilitated different industries to eliminate geographical boundaries and equipped organisations to work collectively to produce goods. Pharmaceuticals is one such industry which has majorly benefited from globalisation to reduce costs and increase profitability at different aspects of research, development, manufacturing of drugs and finally distribution of drugs. Globalisation not only reduces costs and increases profits for the industry, it also makes essential medication available to all individuals even at remote areas of undeveloped, developing and developed countries. As the number of entities or participants grew in the Pharmaceutical Supply Chain (PSC), the complexity and abstractness of the drug delivery from manufacturer to consumer has increased, which raised new concerns like counterfeit medication, incorrect, incomplete or no information about the drug reaching the consumer thus undermining customer confidence and most importantly distribution delays which can cause serious impact on the life of the consumer as well as business growth. Considering all these challenges, there is a dire need for a robust PSC which can eliminate blind parties and provide a transparent chain. A transparent PSC will eliminate data fragmentation between different participating entities by creating a trail of secure, single source of truth for the entire life cycle of a medicine, thereby eliminating the introduction of counterfeits and ensuring the consumer safety. This article proposes a novel Distributed Ledger Technology (DLT) based transparent supply chain for PSC and proof-of-concept is implemented to analyse the scalability and efficiency of the proposed architecture. Distributed Ledger Technology (DLT) is one of the technologies which can provide such transparent PSC and Smart Contracts (SC) are the most commonly used component in such systems to implement business logic and access control mechanisms. The proposed PharmaChain model investigates all the interactions between the main entities in PSC and also addresses smart contract issues such as re-entrance, randomness and lacking trustworthy data feed. Results are compared with other proposed solutions for PSC based on DLT.

1 | INTRODUCTION

The Pharmaceutical Supply Chain (PSC) is a main component of the Healthcare Cyber Physical System (H-CPS) which ensures the access of essential medication with the right amount and quality reach the consumer in need, on time [1]. In this globalised environment, the PSC helps in executing the operations that interact with organisations which are geographically far apart smoothly and redefines the workable boundaries for the pharmaceutical industry in achieving more profitable and less costly strategies [2]. Even though the PSC provides many advantages, it is more complex in terms of goods, information and financial flow due to the multiple entities involved which include manufacturers, the Food and Drug Administration (FDA), retailers, distributors, mail-order pharmacies, brick and mortar pharmacies, Pharmacy Benefit

Managers (PBM), Health Insurance Companies and the consumer. Associations between these entities change from time-to-time based on the need and usage patterns. This entanglement of the PSC has created an opaque view which makes it difficult to trace the drugs within the supply chain and makes entities accountable for the issues raised. One of the most common and immediate threats is introduction of counterfeits into the supply chain by malicious entities. These counterfeits contain from inefficient ingredients to placebos or drugs manufactured in substandard conditions, and even more lethal, repackaging expired drugs and introducing them into the supply chain [3]. Identifying and penalising such malicious players in the PSC is difficult in current systems due to the tangled PSC [4]. Many such incidents have happened in the past. One such recent incident is that of the Canadian wholesaler SB Medical being accused of mishandling medications in substandard condition before shipping them into the US PSC [5]. According to the World Health Organisation (WHO), approximately 30% of the drugs sold across 140 countries are believed to be counterfeit and account for $250 billion per year [6] which does not include non-monetary damage. This clearly shows the immediate attention needed to redefine the architecture of the PSC to make it more transparent and consumer safe.

The traditional architecture on how different entities in the PSC interact can be seen in Figure 1 [7, 8]. This figure does not include back orders and some complex interactions in order to give a simple overview of the pharmaceutical supply chain and the main entities involved. Manufacturers in the PSC are responsible for producing different generic and brand name drugs. Brand name drugs are named by the producing company which has originally developed and discovered them, whereas generic drugs makes use of the same Active Pharmaceutical Ingredient (API) at the same concentrations as the brand name drug but may vary in colour, shape, taste inactive ingredients and packaging. Brand name drug manufacturers invest some of their expenses in research and development (R&D) of new drugs, whereas generic drug manufacturers typically do not have such expenses which makes generic drugs more affordable than brand name drugs. Producing these drugs requires a variety of chemicals along with many natural raw materials like plants, animals/humans, micro-organisms along with minerals and inorganic materials. Manufacturers heavily depend on biological and chemical ingredient suppliers for acquiring the required pharmaceutical supplies. Any drug/medication that needs to be marketed under prescription, and in some cases over the counter drugs, is evaluated for toxicity and efficacy and other properties in clinical trials. Once this data is available, a New Drug Application (NDA) for brand name or an Abbreviated New Drug Application (ANDA) for generic drugs is sent to the FDA for approval. Once approved, the manufacturer can engage in mass production and marketing of the new drug. Produced drugs from the manufacturer will be sent to the distributors/wholesalers for further distribution in the supply chain. Manufacturers typically make contracts with some wholesalers and provide bulk purchasing and immediate payment discounts which vary based on type of agreement between the manufacturer and the wholesaler. Along with discounts, the manufacturers also provide some service fee for managing the inventory and distribution. In

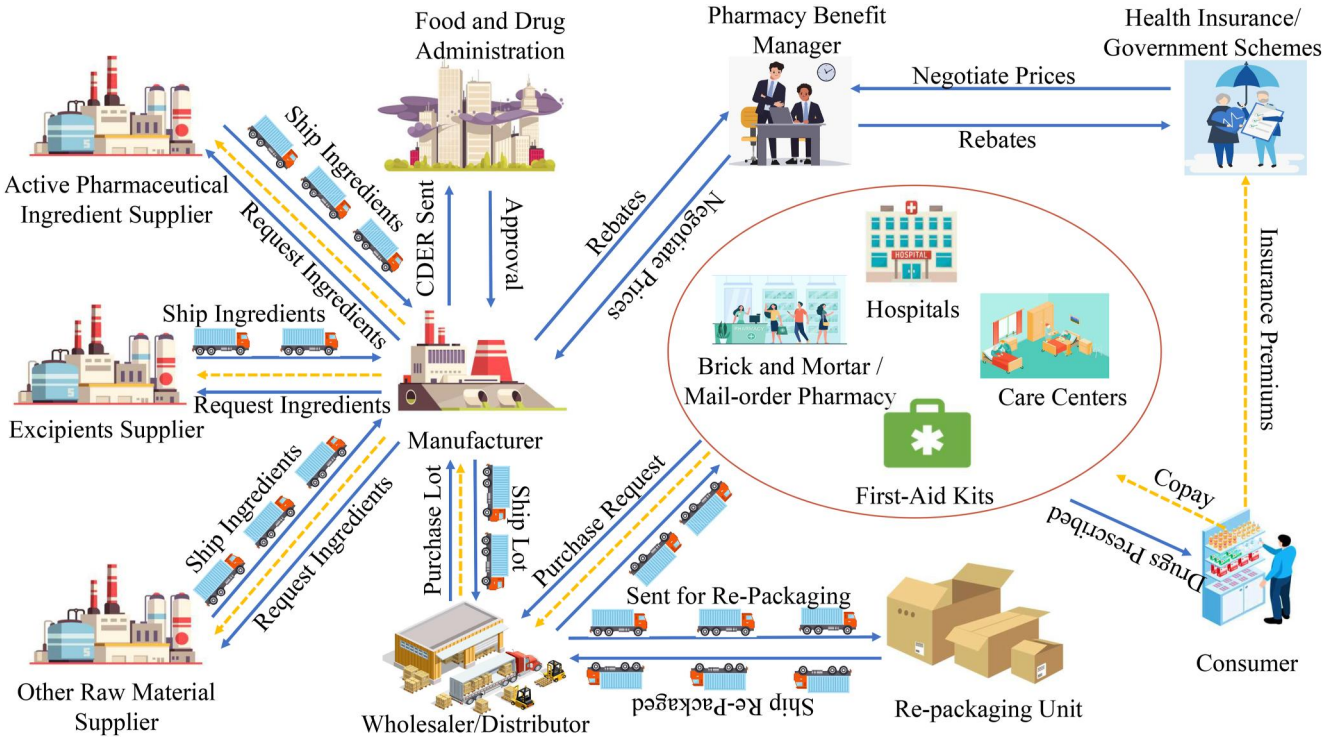


FIGURE 1 Goods and financial transaction flows in the pharmaceutical supply chain

special cases, the manufacturers also ship drugs directly to some healthcare insurance plans, Specialty Pharmacies, and Government subsidised programs like Vaccine For Children (VFC).

The role of wholesalers/distributors in the PSC is to purchase lots of produced medication from manufacturers and stock them until purchased by other actors in the PSC, such as mail-order pharmacies, brick and mortar pharmacies, and local suppliers. This will ease the access of drugs from the manufacturer to different outlets and will also remove the overhead on the drug manufacturers to ship multiple smaller quantities to a large number of consumers. Some of these wholesalers are specialised in specific type of drug distribution, whereas some work in various generic drug distributions. Currently, wholesalers/distributors are also involved in many other initiatives like drug re-packaging, buy-back programs and reimbursement programs. Pharmacies, at the next step of the PSC, work with different medical establishments and distribute drugs directly to consumers based on the need and act as a critical interface between the manufacturers, Pharmacy Benefit Managers (PBM's) and consumers. In a typical scenario, pharmacies purchase drugs from wholesalers at a discounted price determined based on the purchasing power of the pharmacies and negotiations. Pharmacy Benefit Managers (PBM) play a major role as an intermediary to negotiate the prices with the manufacturers and to forward them to different government agencies and insurance companies which will realize those benefits to eligible consumers. Along with negotiating prices, a PBM also provides a wide variety of services like assisting claim processes, provide substitutions for expensive brand drugs, developing the pharmacies and also involve in providing mail-order drug fulfilment etc. The PBMs are responsible for circulating the drugs into the market and increase the manufacturer's share. PBMs are also responsible for creating formulary which is the list of drugs covered under the health plan and also decide the tier which determines the patients' cost.

Pharmacies generally make contracts with PBMs to include into the network of hospitals and for reimbursement of claims at a negotiated rate. Pharmacies usually pay certain amount of fees to access the networks and Point-of-Sale systems provided by the PBMs. Patients at the pharmacies pay for the copayment based on the tier in which the prescribed drug is placed in the healthcare insurance plan and the rest is reimbursed by claim systems managed by the PBMs. Along with these entities, distribution network also relies greatly on logistic management. PSC generally consists of three modes of transport which include air, road and sea. Drugs transported using these distribution networks are more prone to adulteration and damages due to mishandling. Some specialised drugs also need temperature controlled transportation which needs skills, equipment and facilities across the path in which the drugs are moving between the entities. Hence, maintaining and monitoring the quality of the distribution network which can also include third party distribution systems is a major responsibility for the manufacturer to ensure quality products reaching the end consumer.

1.1 | Problems with the traditional supply chain in the pharmaceutical industry

As discussed previously, the PSC is complex and involves multiple different entities which interact with each other in complex ways while drugs are moving in the supply chain. This complexity and abstractness introduced into this supply chain has given rise to multiple problems. Information fragmentation is one of the major problem which will in turn reduce the accountability of entities participating in the PSC [9, 10]. Because of the lack of accountability and the difficulty of penalising the entities with malicious intent, introducing counterfeit medication with degraded to placebos or drugs produced in sub-standard conditions into the PSC in order to monetise them and take benefit of the consumers, is more commonly done. Fragmentation of information is caused by the blinded parties: the entities participating in a transaction are only aware of that transaction instead of having the whole information. This problem is detailed in Figure 2. This lack of whole information at all entities will make it difficult to take unilateral decision in case of conflict and may result in delays of PSC operations. This lack of information can also lead to incorrect prediction of demand and result in unnecessary suffering and potential deaths of consumers, as was seen during the COVID-19 crisis [11, 12]. Hence, having a single source of truth which can keep track of all transactions happening between multiple entities participating in the PSC can give better visibility and can help smooth its operation with low conflicts and faster resolution times. The importance of upgrading traditional systems to increase track and trace of drugs in the PSC is emphasised and mandated by many countries. For example, the United States in its Drug Supply Chain Security Act (DSCSA) introduced in 2013 [13] provides steps to achieve interoperability of different entities and efficient track and trace of drugs at package level. Blockchain technology, in this case, can help in addressing many challenges with traditional supply chain like managing a large number of vendors and manufacturer entities along with complex interactions between them like procurement, manufacturing, reverse logistics, etc. [14].

1.2 | Distributed ledger systems background

Advancements of technologies like the Internet-of-Medical-Things (IoMT), communication protocols, Artificial Intelligence (AI) and Machine Learning (ML), and Distributed Ledger Technologies (DLT) have helped in redefining the PSC and have introduced more efficient mechanisms. DLT is one of the recent technological innovations which have started redefining traditional financial processes [15] and have shown promising impact on other fields like smart cities [16, 17], smart healthcare [18–20], smart farming [21, 22], IoT data security and privacy [23], and also in building efficient and transparent supply chains [24]. Transparent supply chains can help in avoiding the above discussed issues with the PSC, which is the scope of this paper. As different systems have different sets of requirements, DLT

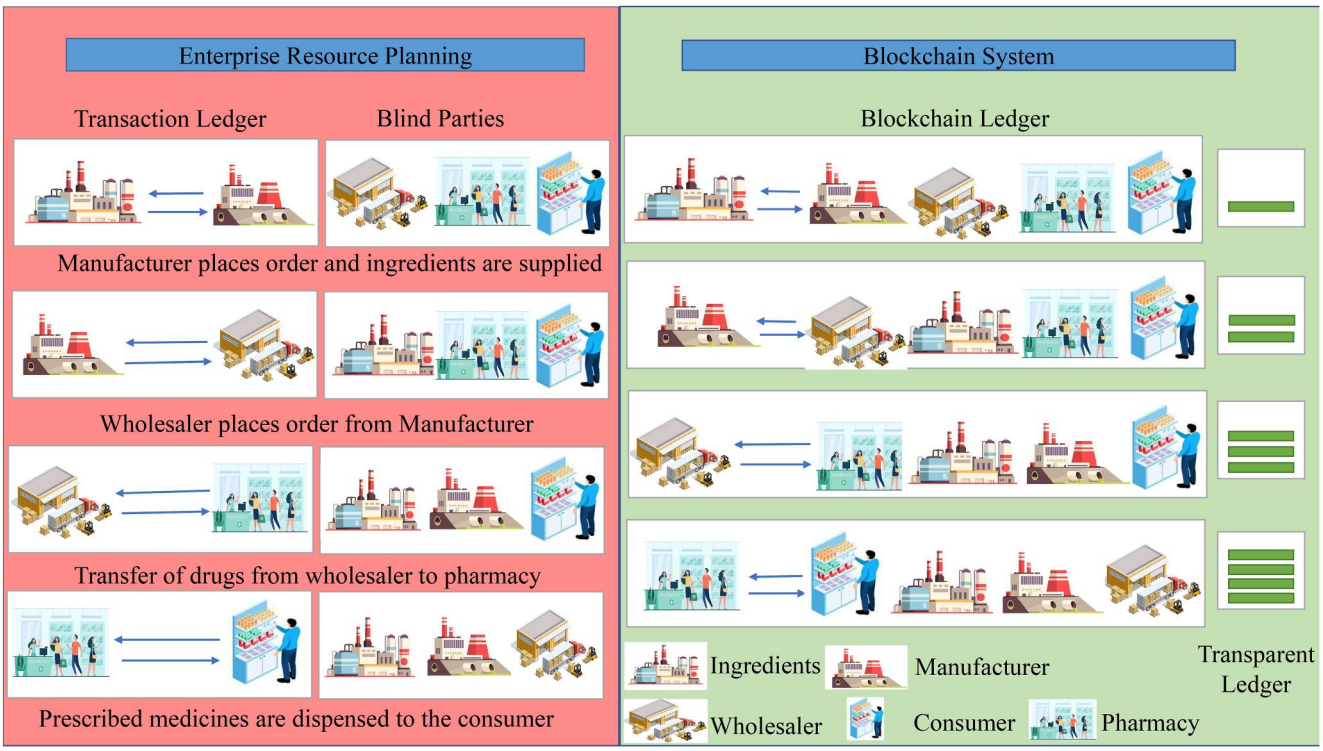


FIGURE 2 Blind parties in conventional record keeping

technology has evolved over time, and can be classified into three generations. First generation DLTs include Bitcoin and Litecoin which are designed for addressing issues with digital assets and have helped in decentralising the traditional banking system with a peer-to-peer verifiable network. As the wide adoption of these first generation DLTs increased, use-cases of this technology in other fields are explored. In the second generation, usage of trust agreements (Smart Contracts) along with maintaining the digital assets is explored which has given rise to the blockchain development platform called Ethereum [25]. This modification has led the wide adoption of DLT technology into more business processes as smart contracts helped in incorporating business logic into blockchains. Even though there are huge benefits from this advancement, it is still facing some issues like scalability, transaction costs, power consumption, transaction speeds, etc. To address these issues, third generation DLTs like IOTA, Hedera Hashgraph, Cardano, etc. came into existence by tweaking the consensus protocols and proposing new structures for the blockchain.

1.2.1 | Consensus mechanism overview

Two important problems with a peer-to-peer network of trustless nodes are Byzantine fault and Sybil attack. Lack of central authority and verification and the probability of having malicious or inactive nodes in the network can create problems by the trustless nodes participating in the network to reach consensus, which is called the Byzantine problem. Similarly, malicious nodes can create an army of zombie nodes and participate in the network to gain power over the network to transmit malicious information within the network, which is called the Sybil attack. In order to address these two issues, a set of predefined rules is used to process the new transactions coming into the network, which is called the consensus mechanism. The consensus mechanism developed for bitcoin consists of two aspects: the longest chain rule and Proof-of-Work (PoW). PoW makes use of the hashcash problem proposed in Ref. [26] to prevent Sybil attacks. To engage nodes in participating as miners, a reward combined with transaction fees is given.

1.2.2 | Smart contract overview

Smart contracts (SC) can be defined as a segment of a self-executing code which is triggered when certain conditions are met. Adaptation of blockchain technology into other fields and the need for implementing business logic resulted in the evolution of smart contracts. These SCs have the capability of executing and interacting with other SCs to work collectively for performing certain business functions. Even though SCs have many advantages in developing Decentralised Applications (DApp), there are some limitations for the usage of SCs which include the inability to call external Application Programing Interfaces (APIs) and interact with real-world data. To solve this problem with smart contracts, Oracles are used in the proposed PSC architecture.

1.2.3 | Oracle overview

The main component of Ethereum blockchain is the Ethereum Virtual Machine (EVM) which is Turing-complete and has the ability to execute the predefined code modules when certain conditions are met (Smart Contracts) and the state of Ethereum is updated based on the result of the execution. EVM execution must be deterministic among all the nodes in the network in order to reach consensus between all the distributed nodes. Along with execution of EVM, the blockchain should be able to replay the transactions and should reach the same state, which is a problem when we use the extrinsic data provided by a data source, as the data source can be mutable and it changes over time [27]. One more problem with using Ethereum-based smart contracts is the introduction of randomness if the smart contract has functions which need to generate a random number or re-entrance of data [28]. Distributed nodes which are geographically apart will resolve different random numbers. For example, node 1 in the network may get a random number of 7 to compute the calculation, whereas node 2 can work with 10 as the random number which leads to difficulties in reaching consensus between the nodes and can lead to blockchain failure. These constraints limit the smart contract capability to interact and fetch data from real-world applications like IoT systems. This in turn invalidates the application of Ethereum-based DApps in most of the real-world applications. Hence, to solve this problem and integrate Ethereum blockchain with external data feeds, Oracles are used. This will help Ethereum smart contracts to create on-chain API and receive the required external data feed into the in-memory of the smart contract. These Oracles provide a decentralised verifiable data sharing system using which the blockchain transactions can be replayed and still reach the consensus state. These Oracles can be classified into different types based on different parameters: Information source, Degree of Centralisation and Direction of Information flow. Based on the type of information source, Oracles can be classified into Software Oracles and Hardware Oracles. Software Oracles make use of online sources like RSS data feeds. Sometimes, smart contracts need to interact directly with hardware resources like in the current scenario where the data from sensors, bar-code scanners, etc. has to be integrated and used in the proposed PharmaChain smart contract. Such Oracles are called hardware Oracles. Based on the degree of centralisation, Oracles can be divided into centralised Oracles and decentralised Oracles. In centralised Oracles, data is provided by a single data source, whereas for decentralised Oracles the data will be coming from multiple information sources and aggregated before using it in the smart contract. Using centralised Oracles is more prone to security threats and any malicious entity at the data source can lead to introduction of malicious data into the smart contract. Based on the direction of information flow, Oracles can be classified as Inbound Oracles and Outbound Oracles. Inbound Oracles will take the data from the external world and then provide it to the smart contracts in-memory so that it can be consumed by different functions within the blockchain. Outbound Oracles help in taking data from the smart contracts and are interfaced to the external world.

The proposed PharmaChain architecture leverages blockchain technologies combined with the IoT to address the issues present in the traditional PSC. The rest of the paper is organised as follows: Section 2 discusses the proposed and existing DLT-based solutions for the PSC. Section 3 clearly describes the problems addressed by the proposed PharmaChain in the traditional PSC, along with an overview of H-CPS systems in the proposed PharmaChain solution. Section 4 presents the architectural overview and methods proposed in PharmaChain. Section 5 presents the implementation details of the proposed Pharmachain with extensive discussion on implemented smart contracts. Section 6 discusses the results of functional testing performed on the implemented PharmaChain prototype. Section 7 analyses the scalability, reliability, security and other aspects of the implemented PharmaChain solution. Section 8 provides conclusions of the paper along with possible future directions for research.

2 | RELATED PRIOR RESEARCH

Due to globalisation which resulted in increasing complexity and abstraction in supply chains, the PSC is now more prone to counterfeits. It enhanced the need for introducing more recent technologies into the supply chain to enhance the security of medications and the safety of consumers. The complexity of the PSC has also led to significant delays in the distribution of medications which was experienced during COVID-19 vaccine distribution. Regulations are placed by many different countries, including the US, like the Drug Supply Chain Security Act (DSCSA) which mandates the usage of technological components to build a robust drug supply chain and which can efficiently track and trace the products moving within the supply chain. This section critically analyses the existing blockchain-based solutions proposed to avoid counterfeits and provide genuine medication at the place of need, at the right time [29].

Crypto Pharmacy is a mobile application integrated with the New Economic Movement (NEM) blockchain solution for authenticating and purchasing medicine securely using the blockchain native token XEM [30]. The mobile application designed uses a smart assets system which is a powerful engine provided by NEM and the manufacturer creates the mosaics which are associated with the quantity of the drug produced and transferable between NEM addresses. This helps in keeping track of all the hand-off between different entities to help verify the authenticity. The proposed model makes use of the Proof-of-Importance (PoI) consensus mechanism which is based on the reputation of nodes with more number of confirmation to select the mining node. It provides a mechanism to authenticate the product but the proposed model is not analysed for scalability, availability and security. Modum.io AG is another application developed based on blockchain [31], which leverages Ethereum-based smart contracts and IoT systems to ensure the safety of the drugs being handled within the supply chain. The proposed mechanism makes use of Bluetooth LE for getting shipment details at every stage and to

communicate them to the blockchain network using mobile devices to generate the transactions. This paper proposes a robust system, however, making use of PoW can limit the transaction confirmation rate and increase network congestion. In addition, manual scanning and inputting of environmental parameters of shipments is needed which could introduce a multitude of errors and false data injection from the front end of the mobile application by malicious entities. Scalability, availability and security analysis are not performed for the proposed model. Another blockchain-based solution is proposed in Ref [32] which makes use of Public Key Infrastructure (PKI) where smart contracts are used to check the authorised privileges to access information. Once privileges are verified, data will be retrieved and encrypted using the participant's public key before sharing the information. This will ensure that data can be accessed by the intended party. This proposed mechanism takes only the interaction between the manufacturer and participants (consumer) to share information securely, whereas the supply chain information should be made public and all the interaction between the entities should be recorded. Also, in the proposed mechanism, the manufacturer will act as the centralised authority and participants have to retrieve data through the manufacturer, unlike the proposed system where they are available directly from the blockchain. Performance metrics are not discussed and IoT systems are not used for automated processing and capturing environmental parameters.

Druggeder is another blockchain-based solution proposed in Ref. [33] which will associate drugs manufactured to the Unspent Transaction (UTXO) of the bitcoin and make use of it to track and trace at multiple stages of the supply chain. Implementing business logic with the limited scripting capabilities available in bitcoin is difficult which in turn limits the recording of the complex interaction in the supply chain network. Transaction speeds are also lower in bitcoin compared to Ethereum which can cause network congestion and backlogging large number of transactions. Efficiency and scalability aspects of the proposed model are not discussed in this work. Another such application with blockchain as solution for counterfeit detection is proposed in Ref. [34] which makes use of the hyperledger fabric platform. The proposed model uses a private permissioned blockchain and leverages chain-codes to implement supply chain functions. Throughput is analysed for the proposed model, however, all the entities in the supply chain are not considered along with the access control mechanisms. A user interface is not provided for the proposed model and IoT systems are not used for continuous shipment tracking to provide entities with real-time decision support tools unlike our current work. IoT-based Blockchain Distributed Network, which makes use of Raft consensus mechanism and bloXroute servers to improve the scalability and provide counterfeit detection mechanism, is proposed in Ref. [35].

The Ethereum-based blockchain solution cryptocargo is proposed in Ref. [36] which leverages IoT systems to provide a smart shipping container capable of monitoring the environmental parameters and use different cloud functions to provide real-time notifications to entities and log violations in the blockchain. Along with that, interactions are recorded using the Ethereum blockchain which makes use of PoW as the consensus mechanism. However, the PoW consensus mechanism will limit the throughput of the network significantly compared to the consensus used in the current work, Proof-of-Authority (PoA). Smart contracts used in the proposed model suffer the major drawback of not being capable of calling APIs to retrieve real-time data from the shipping container, unlike in our work where Chainlink Oracle solutions are used to interact and interface data from the sensor IoT network directly into smart contracts removing human errors and addresses some security vulnerabilities of smart contracts like reliable data feeds. Time-dependent changes of sensor data can cause unpredictable states of the EVM, and generating randomness for business functions is also a problem [37]. A solution, the Town-Crier (TC) for financial data feeds based on Intel's Software Guard Extension (SGX), is proposed in Ref. [38]. In our proposed PharmaChain, a robust Chainlink Oracle mechanism is developed to provide reliable data feed to the smart contracts. A similar approach of using the Ethereum platform-based blockchain leveraging smart contracts for business functions has been implemented in Ref. [39, 40]. A cloud-based solution combining both IoT technology and hyperledger fabric blockchain is proposed in Ref. [41]. Comparison between our proposed PharmaChain solution and existing blockchain-based solutions can be seen in Table 1.

2.1 | Problem statement

As discussed from Table 1, IoT Data integration to blockchain in other implementations is done using cloud functions which will centralise the data source and undermine the benefits of the blockchain. A distributed system needs a distributed data source, hence in the current proposed PharmaChain the Oracle's distributed data sources are used which ensures data fed into the system is more reliable and the transactions of blockchain re-playable. This in turn will increase the reliability and accuracy of the developed blockchain-based PSC. Using such decentralised data sources will equip smart contracts with the capability to interact with external API's and consume IoT data instead of manual Oracle data which can be error prone. Along with addressing these issues, the current PharmaChain also aims to include more interactions between different entities to model accurate real-world PSC and also developing a robust blockchain role-based access control mechanism which can be used to ensure that each entity can be assigned with specific roles and responsibilities.

3 | NOVEL CONTRIBUTIONS

This section discusses the problems faced in traditional PSC and the novel solutions proposed in the proposed blockchain-based PharmaChain architecture. An overview of the proposed architecture can be seen in Figure 3.

TABLE 1 Comparison between our proposed solution and other existing solutions

Parameter	Subramanian et al. [30]	Bocek et al. [31]	Kumar et al. [32]	Huang et al. [33]	Alhoori et al. [36]	Our solution
Blockchain Platform	New Economic Movement (NEM)	Ethereum	-	Bitcoin	Ethereum	Ethereum
Business Functions	Smart Contracts	Smart Contracts	-	UTXO Scripts	Smart Contracts	Smart Contracts
Consensus Mechanism	PoI	PoW	-	PoW	PoW	PoA
Data Integration from IoT	Cloud Functions	Centralised database	[×]	[×]	Cloud Functions	Oracles
Transactions Re-playable	[×]	[×]	[×]	[×]	[×]	[✓]
IoT Integration	[✓]	[✓]	[×]	[×]	[✓]	[✓]
Scalability Analysis	[×]	[×]	[×]	[×]	[✓]	[✓]
Cost Analysis	[×]	[×]	[×]	[×]	[×]	[✓]
Security Analysis	[×]	[×]	[×]	[✓]	[×]	[✓]
User Friendly Interface	[✓]	[×]	[×]	[×]	[✓]	[✓]
Access Control Mechanism	[×]	[×]	[×]	[×]	[✓]	[✓]
Real-time Decision Support Tools	[×]	[×]	[×]	[×]	[✓]	[✓]
Throughput	Highest	Less	-	Least	Less	Higher

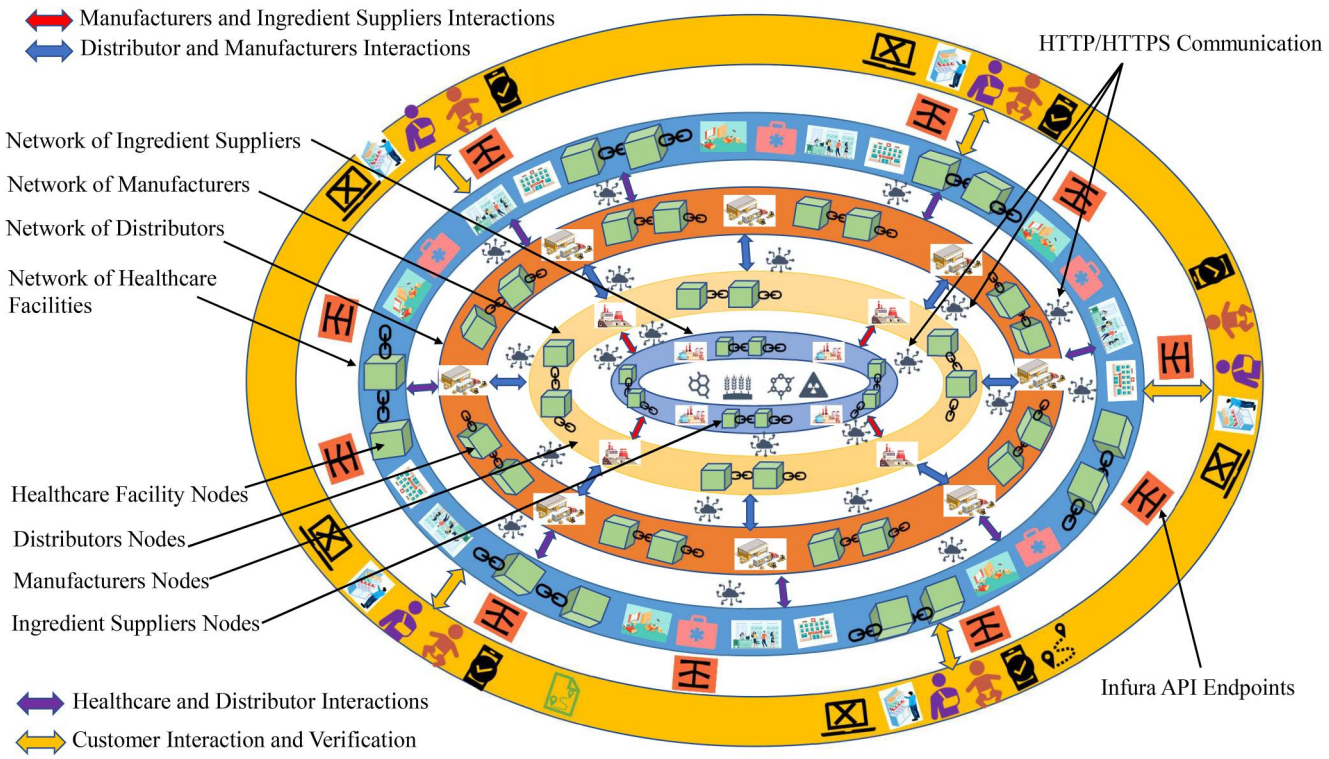


FIGURE 3 H-CPS system with proposed PharmaChain solution

3.1 | Problems addressed in the current paper

- Order processing delays are very common because of complex interactions between distributed entities which may lead to delays in essential medication reaching the end consumer.
- Information fragmentation caused by the distributed nature of the system can lead to difficulties in finding and resolving of issues.

- Lack of accountability of participating entities due to difficult track and trace issues in such a complex environment.
- Blind parties in traditional PSC can cause invisibility and can create severe processing delays both at manufacturing and distributions.
- Detecting and punishing the malicious entitiesthereby introducing counterfeits into the supply chain is hard.
- Major economic losses and wastage during the recalls, and even safety of the consumer is at stake.
- Data recording mechanisms are not automated in the traditional PSC, causing delays and are also error prone.

3.2 | Novel solutions proposed in PharmaChain

- The proposed blockchain-based solution in PharmaChain can help in expediting the order processing by avoiding the communication delays and ensure smooth operation of distributed entities.
- The blockchain ledger generated by the proposed PharmaChain will act as a single source of truth and is accessible by all entities in the network, thereby solving the information fragmentation issue in traditional PSC.
- The proposed PharmaChain creates a transparent supply chain which will ensure the data is accessible by all the entities involved by removing blind parties which can help in collective working and prompt decision making.
- Counterfeiting, which is major problem in traditional PSC is addressed by keeping track of the drug movement in the supply chain from the manufacturer till the Point of Sale (PoS) using a distributed blockchain ledger.
- Transactions recorded in the proposed PharmaChain are cryptographically signed, which makes them non-reputable. This increases the accountability of the participating entities and helps in taking prompt actions against malicious entities in the network.
- Drug recall, which is a common and expensive problem, can be made more economical and efficient as the entire trail of drug movement in the supply chain is being tracked by using an immutable ledger in the proposed PharmaChain.
- The proposed PharmaChain mechanism makes use of the IoT, which ensures that environmental parameter data is accessed with high accuracy and reduces human errors in the data being fed into the network.
- Smart contracts alone cannot interact with real-word data using APIs. Oracles are used in the proposed PharmaChain to address this issue and automate the entire track and tracing mechanism.
- A real-time decision support tool is also built in the proposed PharmaChain to check the status of product within the supply chain and take prompt actions before drugs become unsafe.

3.3 | Novel blockchain (PharmaChain) in H-CPS for counterfeit detection and prevention

The proposed PharmaChain solution seamlessly integrates with the H-CPS system to leverage the benefits of preventing counterfeit medication and making operational procedures more efficient by providing a transparent chain. An overview of the H-CPS system integrated with the proposed PharmaChain solution is shown in Figure 3. As shown in the figure, H-CPS system consists of a large number of entities interacting with each other in many different possible ways. Different ingredient suppliers will form a network and can create their own nodes to participate in the PharmaChain. These nodes store the transactions emitted in the blockchain. Similarly, manufacturers, distributors and medical service providers also form a network and maintain nodes which will act as participants of the PharmaChain network, thereby receiving all the updates on all the medications moving through the entire supply chain. On the other hand, consumers do not need to setup or maintain nodes in order to purchase or verify the authenticity of the medicine received. A consumer will be able to interact with the transparent log in the PharmaChain using the Infura gateway.

4 | OVERVIEW OF THE PROPOSED PHArmaCHAIN

An architectural overview of the proposed PharmaChain is discussed in the following section. PharmaChain can be divided into 5 logical components: (1) IoT sensing nodes placed in transport trucks, (2) a cloud component for off-chain storage of unimportant data, (3) a Chainlink component for interfacing real-time monitoring parameters from the sensing nodes to Ethereum smart contracts, (4) an Ethereum blockchain component for creating an immutable and transparent ledger for all the entities, and (5) a web DApp to interact with the blockchain by entities to perform different functions along with verifying the authenticity of the drugs at any stage of the supply chain. A high-level overview of the PharmaChain application is shown in Figure 4. Sensing nodes in the proposed mechanism are designed with minimal power, scalability and form-factor in consideration as a large number of devices is required to track all the transporting trucks. The sensing nodes are designed to monitor important parameters for pharmaceutical shipment which include temperature and humidity along with GPS co-ordinates of the shipment. Monitoring data from sensing nodes will be processed and formatted into a JSON file before being sent to the cloud component. Sensing nodes make use of the lightweight Message Queuing Telemetry Transport (MQTT) protocol and the data will be published on to an already created topic. The IoT hub will be using event listeners on the created topic and any new data posted on the topic from authorised sensing nodes will be consumed. The communication between the sensing nodes and cloud is protected by using self-signing digital

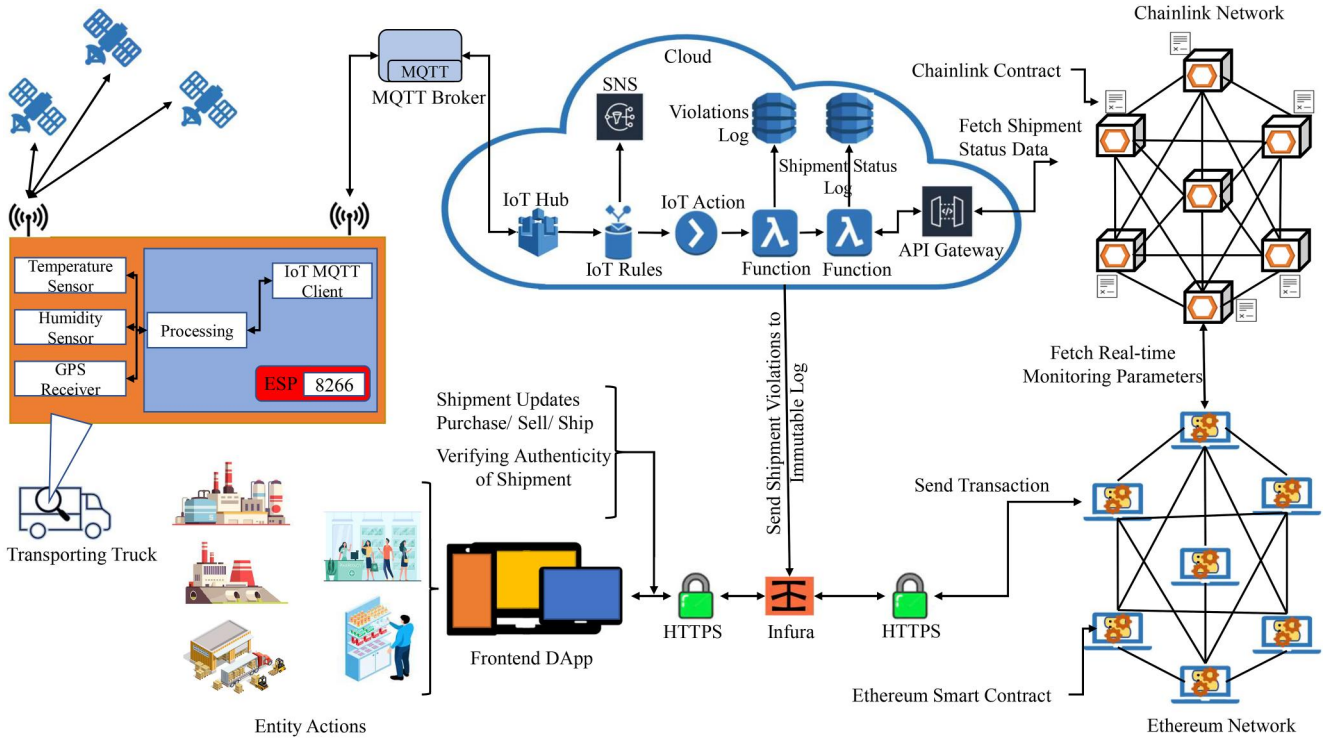


FIGURE 4 Architectural overview of proposed blockchain-based solution PharmaChain

signatures. This ensures the removal of malicious nodes. The communication between the sensing node and cloud component is shown in Figure 5.

The second component cloud architecture is discussed here to give a clear idea on how the data providers can be designed. These data providers can be the ChainLink nodes deployed and maintained by the PSC entities or third party data providers. These data providers are responsible for deploying the sensing nodes and for collecting data and providing real-time updates to the smart contracts. Deploying and maintaining the nodes by third party entities will impose cost overhead on the drugs which can be eliminated in the other case. To create a reliable data source, data from multiple data sources is consumed by the aggregator function of the Oracle and aggregation is performed from all the data source nodes. The most common aggregation function is getting the median of all the responses and providing a single verifiable temperature, humidity or location of the shipment. The cloud component is designed in a server-less configuration for limiting the difficulty of setting up and maintaining the cloud infrastructure. The proposed cloud component consists of an IoT core which is responsible for authenticating the devices and consuming the new data published onto the created MQTT topics by the sensing node. Once the data is consumed, it is verified using IoT custom rules which are designed to track abnormalities in temperature and humidity. A real-time notification using Simple Notification Service (SNS) is sent to the concerned entities to notify them of the abnormalities. This serves as a real-time decision support tool by the entities for real-time monitoring and controlling of the shipment health. Apart from sending the notification to the concerned entities, an immutable blockchain log is also generated by

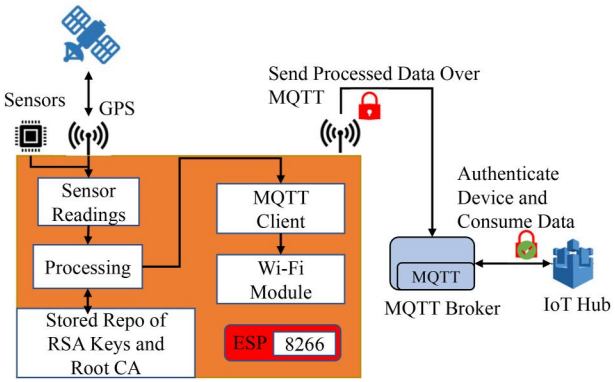


FIGURE 5 Proposed sensing node sharing data securely to cloud

creating a transaction from the cloud. This is achieved by sending the transaction to blockchain via the Infura gateway. This immutable log can help in verifying that the drug shipment has been in a safe environment during the shipping process. Along with creating an immutable log, abnormalities are also logged in the cloud database which is NoSQL for auditing purposes. The other important function of the cloud component in PharmaChain is to act as a data provider for the Chainlink component. Ethereum smart contracts are not capable of interacting with real-world data which in most cases is available via APIs. This will limit the usage of smart contracts in the majority of real-world scenarios, hence a Chainlink component is needed for Ethereum smart contracts. A data provider, which will take the latest data from the sensor network and provide it via an API, is needed for the Chainlink network. The proposed cloud component is designed in such a way as to

use the API gateway and provide the latest shipment status to Chainlink. The proposed architecture in [42] provides a mechanism to keep the data providers engaged by providing incentives by leveraging smart contracts. Various steps performed in the cloud component can be seen in Figure 6.

Reliable and tamper-proof data from on-field sensors is needed to be provided to the blockchain network to make the application reliable and robust. Smart contracts as such cannot interact with Application Programing Interfaces (API's), hence Oracles are used to feed reliable data from sensing nodes to ethereum smart contracts. The PharmaChain model makes use of decentralised inbound hardware Oracle to achieve sending verifiable data from sensing nodes to smart contracts. The steps of execution for integrating Oracles with smart contracts are:

- A request is created by the hybrid smart contract which requires extrinsic data to be used in any of the smart contract functions. This request consists of a set of parameters: job id (which will be different based on the data type of the requesting data), destination address of the data along with the fulfilment function which will perform simple operations on the requested data before sending it to the in-memory of the smart contract.
- The Oracle contract will publish an event in the next step to the entire Oracle network with all the parameter details like job id, destination id, and fulfilment function along with the fee assigned by the requester.
- Once the event reaches all the Oracle network nodes, the node which has the specified job id will take up the task and make use of the set parameters to execute the requested function.
- As the execution is done based on the type of fulfilment function, the given data will be modified before sending it to the hybrid smart contract.
- Once the fulfilment function is executed, the data will be available in the hybrid contracts in-memory for its functions to be used.

A decentralised Oracle structure is implemented in the proposed PharmaChain mechanism in order to enhance security and avoid issues with centralised Oracles. Multiple jobs are executed to fetch the data instead of a single job. The results from these multiple jobs are integrated by using an aggregator function before publishing the data to the hybrid smart contract. Oracle operation can be seen in Figure 7.

The fourth component is the blockchain with hybrid smart contract which is responsible to perform all the business logic in the pharmaceutical supply chain. An Ethereum-based blockchain system is proposed in the current PharmaChain application. Ethereum provides both MAINNET and TESTNET. MAINNET consists of cryptocurrencies which hold monetary value and in each transaction certain ether (currency in Ethereum network) is utilised every time a transaction is made. In order to avoid the real usage of money, TESTNET is used to test the created DApps before deploying it to the MAINNET for real usage. Ethereum provides test networks which are ROPSTEN, which simulates the main network and uses Proof-of-Work (PoW) as consensus mechanism. The PoW consensus mechanism provides greater security in transferring of digital assets but the computational resources needed are large. In order to avoid such need for computational resources and increase scalability, the test net KOVAN which executes based on Proof-of-Authority (PoA) is used in PharmaChain. Apart from the consensus mechanism (PoA), the other reason for selecting a KOVAN testnet among other available test networks is the availability of Chainlink test nodes. As per the chainlink marketplace, KOVAN has 71 active nodes compared to Rinkeby with 17 active nodes and Goerli with 1 active node. This shows that Chainlink nodes are more active in the Ethereum KOVAN testnet which can make it easy to reciprocate the results of the implemented prototype of PharmaChain. Instead of deploying and running a full Ethereum node to connect to the network and performing transactions, a JSON-RPC (JSON Remote Procedure Call)-based Infura gateway can be used to connect to the hosted nodes and

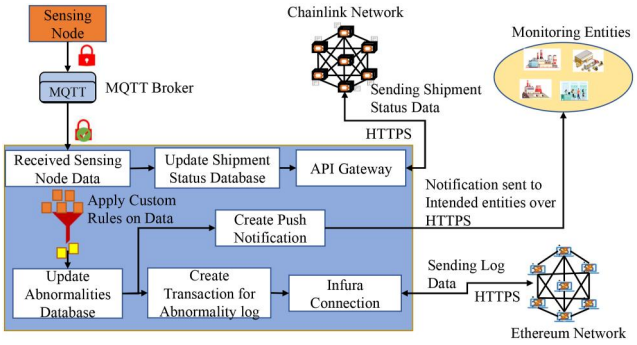


FIGURE 6 Functioning and data flow of cloud component in proposed PharmaChain

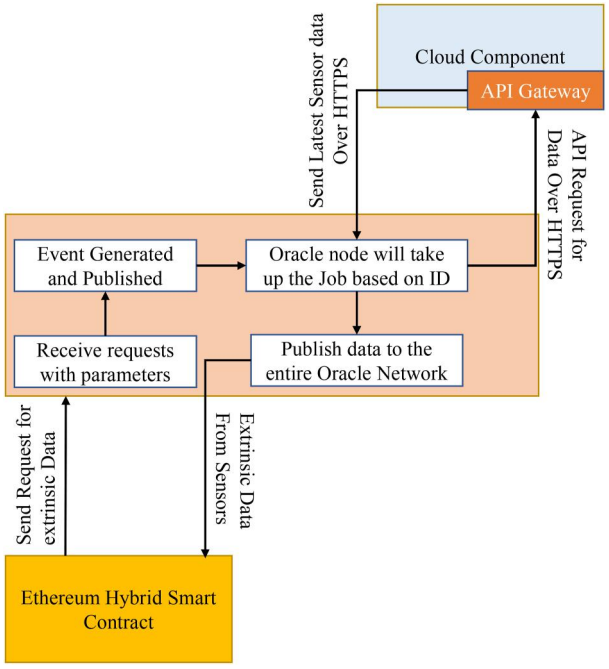


FIGURE 7 Functioning and data flow of Oracle component in proposed PharmaChain

perform transactions in the network. This will reduce the need for deploying and maintaining the nodes to make transactions. It will provide seamless integration of the DApp with the Web3 component and to build easy to use interactive web interfaces for interacting with the blockchain.

Hybrid smart contracts designed in current application can be logically divided into three components. As each entity is responsible for performing certain actions, an access model is needed to be implemented which will authorise the transacting node to perform the intended functions. Hence, a Role-Based Access Control (RBAC) model is proposed in the current application. Fewer important end-to-end entities are considered while designing the RBAC mechanism. Back orders/returns are also considered to remove the complexities of the design while only success paths are considered, where shipment moves from one entity to another without any issues. The manufacturer entity is responsible for operations like producing the lots of drugs and selling them to distributors. Distributors have the authorisation to purchase the lots of drugs from a manufacturer, process and repackage them, sell and ship the repackaged goods to retailers which include mail-order pharmacies and brick-and-mortar pharmacies. These retailers have only one function: selling the drugs to the final consumer. At every stage of the life cycle of the drug, authenticity can be verified to make sure no counterfeits are included in the supply chain. This RBAC model is achieved in PharmaChain by using modifiers of solidity. Modifiers are conditions which verify the authorisation of the transacting Ethereum address if the required privileges to call the function are sanctioned. If privileges are not assigned to the current transacting address, the transaction is discarded at no gas cost. Some of the modifiers used in the RBAC model are onlyManufacturer, onlyDistributor, onlyRetailer, onlyConsumer, etc. An entity activity diagram is shown in Figure 8. The second logical component of the hybrid smart contract is the business functions which are designed to implement the update of shipment status throughout the life journey within the supply chain. This component is useful in including all the business logic related to the PSC and also performs complex interactions between entities. The third logical component of the hybrid smart contract is the Oracle component in which the smart contract will create the request for extrinsic data and publish it to the Oracle network. In the proposed design, reliable nodes are considered and Job IDs are hard coded to send the transactions.

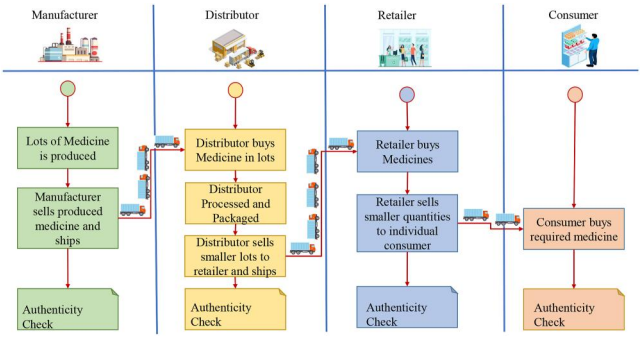


FIGURE 8 Entity activity diagram considered in implemented PharmaChain solution

Responses from the Oracle component will come aggregated and the received data is updated into the in-memory of the smart contract and fetched by different business functions.

The final component of the PharmaChain system is the Web App which can be used by all participating entities to verify and authenticate the drugs received by them. Infura endpoints help in transacting with the deployed smart contracts on Ethereum by providing an API and contract address. Web3, API, HTTP connection, JavaScript, HTML and CSS are the technologies which are used in designing the web app designed for PharmaChain.

5 | IMPLEMENTATION OF THE PROPOSED PHARMACHAIN

This section presents implementation details of the PharmaChain solution for counterfeit detection and avoidance in the PSC. Details regarding the design for all sensing nodes, cloud components and blockchain components are presented.

A prototype with a single sensing node which is capable of tracking and tracing a shipment from end-to-end in the pharmaceutical supply chain is designed. It is designed by using a low form-factor and power efficient NodeMCU ESP8266 module which has integrated Wi-Fi and is capable of connecting to wireless networks. It also comes with different interfacing mechanisms like I2C, and SPI. In the current design, I2C is used to connect the required sensors to the NodeMCU module. For measuring both ambient temperature and relative humidity around the shipment a DHT11 sensor is used. In order to track the shipment during the transportation a NEO-6M GPS module is interfaced to NodeMCU. The designed sensing node module can be seen in Figure 9. The data from the sensing node is sent over a secure channel to the cloud component. To maintain the integrity and authenticity of the data being sent RSA digital signatures are used. The required private key, public key, and Root CA certificate are loaded into the SPI Flash File System (SPIFFS) of the sensing node. The WiFIClient library is utilised to load all the certificate files to establish a secure connection between the sensing node and cloud. Once the secure connection is established, the environmental parameter data from the sensors DHT11 and GPS module along with other shipment information is encoded into a JSON string data. This string data is converted into a character array before

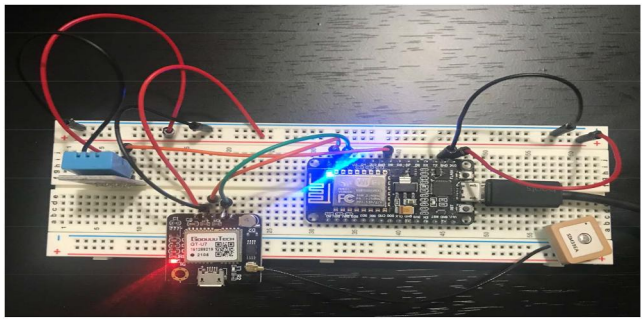


FIGURE 9 Designed sensing node for proposed PharmaChain solution

publishing to the topic using the PubSubClient library. The steps of loading the certificates, establishing the secure connection and sending the data to the cloud can be seen in Algorithm 1.

Algorithm 1 Algorithm for Sensing Node in Proposed PharmaChain Implementation.

Input: Certificate and Keys for IoT thing, Ambient temperature (temp) and relative humidity (hum) from DHT11 Sensor, GPS latitude (lat) and longitude (lng) co-ordinates from GPS Module
Output: Boolean status of message published to topic

```
/* During Setup sensing node tries to setup
   wifi and MQTT connection and establish
   secure channel to communicate between
   sensing node and cloud                    */
Load required libraries for WiFi Connection, MQTT
    Communication, JSON for data manipulations
    /* Sensing node tries to connect to Wi-Fi
    network using given SSID and PASSWORD
    strings if it not connected to network   */
while WiFi.status() == FALSE do
    WiFi.begin(ssid, password)
/* SPI Flash File Systems is mounted
   successfully                              */
while SPIFFS.status() == FALSE do
    SPIFFS.begin() /* Open Certificate and Key
       files in flash memory of sensing node
       */
    SPIFFS.open("Root CA certificate(CA_cert)")
    SPIFFS.open("Public Key (PuK)")
    SPIFFS.open("Private Key (PrK)")
    /* Load certificate and key files        */
    esp8266Client.loadCertificate(PuK)
    esp8266Client.loadPrivateKey(PrK)
    esp8266Client.loadCACert(CA_cert)
/* Attempting MQTT connection                */
while pubClient.connected() == FALSE do
    /* Trying to reconnect                   */
    pubClient.connect()
/* Environmental and Geolocation inputs from
   sensors along with shipment overhead
   information is processed every 60 seconds
   */
for every 60 seconds do
    /* JSON String is genereted including all
       environmental and geolocation data
       from sensing node                     */
    Publishing message MSG ←
     JSON(timestamp,lat,lng,sku,lot,drugName,temp,hum)
    /* Data to be published is converted to
       char array                            */
    MSG+ ← JSON.toCharArray(MSG)
    /* Converted MSG+ is published to topic
       τ_pub                                 */
    pubclient.publish(τ_pub,MSG+)
```

Environmental parameters along with shipment details sent from the sensing node are consumed and processed by the designed cloud architecture. The consumed data has undergone custom rules to filter and alert the entities' monitoring with real-time alerts which will help them in taking prompt actions in securing the shipment. In the designed prototype an over-the-counter medication for which the safe temperature ranges are prescribed as less than 25°C is considered. Custom rules are designed in such a way as to check if there are any temperature abnormalities. In this case, a temperature greater than 25°C is considered as an abnormality and the notification is sent to the monitoring entities. This way the designed cloud architecture acts as a Decision Support Tool (DST) and equips the entities in PSC to verify and take prompt actions. These abnormalities are also recorded in a monitoring abnormalities' dynamo database. Apart from notification, the latest status of the shipment and its environmental parameters are also updated in the database tables and it is made available at the designed API endpoints. These API gateways will act as the data source for the Oracle network and provide real-time data of the shipment to the smart contracts for performing other business functions. Any abnormality detected during the custom rules will also trigger an immutable transaction on the Ethereum blockchain via the Infura gateway. The steps followed during the consumption of sensing data and providing latest information to the Oracles can be seen in Algorithm 2 and the custom rules and functions implemented in the prototype can be seen in Figure 10.

In the prototype designed for PharmaChain, Chainlink is used for the Oracle services and the Ethereum platform is used for designing the blockchain transparent ledger which uses smart contracts. As the smart contracts are not able to interact with extrinsic data, Chainlink Oracle services are used to interface data from the sensing node to the smart contract [43]. Such smart contracts which can perform the API calls to get the external data are coined as Hybrid Smart Contracts. Ethereum is an open-source platform which provides an efficient platform to build and deploy DApps. The main component of the Ethereum platform is the Ethereum Virtual Machine (EVM) which acts as world computer for which the state will change based on different transactions and smart contract executions. It also provides a Turing complete language called solidity to programme the smart contracts. These smart contracts will execute when predefined conditions are met and the executions will lead to the change of state of the EVM. MAINNET of the Ethereum currently works on the Proof-of-Work (PoW) algorithm which is developed on the basis of the Hash Cash mechanism and makes use of computationally hard hash problems to decide the miner node that generates a new block. But the current implementation is designed keeping in mind the scalability and computational ability of IoT systems. Hence, KOVAN Testnet is considered which runs on more resource aware consensus called Proof-of-Authority (PoA). Testnets are designed to simulate similar to MAINNET but use cryptocurrency which is of no value and

Algorithm 2 Algorithm for Data Source API Generation in Proposed PharmChain Implementation.

Input: Input JSON message from Sensing Node (MSG+), HTTP GET API Request from Oracle
Output: Requested Shipment Data

```
/* Data published MSG+ on topic τ_pub from previous algorithm is consumed by the IoT cloud component */
if τ_pub.hasMessage() then
    MSG ← IoTCore.consume(τ_pub,MSG+)
    /* In the example drug taken temperature threshold is 25°C */
    if MSG.temperature > 25) then
        /* Update the audit table for temeprature abnormalities recorded during the supply chain process */
        AbnormalTempAuditTable.update(MSG)
        /* An E-mail notitification will be sent to the registered parties in case the temperature exceeds the threshold */
        SNSNotification.sendEmail(MSG)
    /* Update this table with latest information of the shipment */
    ShipmentLatestUpdateTable.update(MSG)
while HTTP GET Request (req) do
    /* get request path parameters to geenrate the DB query */
    pathParameter param = req.getParam()
    /* Scan the DB Table to retreive requested shipment information and store result */
    result = ShipmentLatestUpdateTable.scan(filtering param)
    /* result from scan is mapped to http body */
    http.mapBody(result)
    /* A http response is generated and sent back to the requested Oracle */
    httpResponse (resp) ← http.generateResponse(httpHeaders,httpBody)
    http.ResponseSend(resp)
```

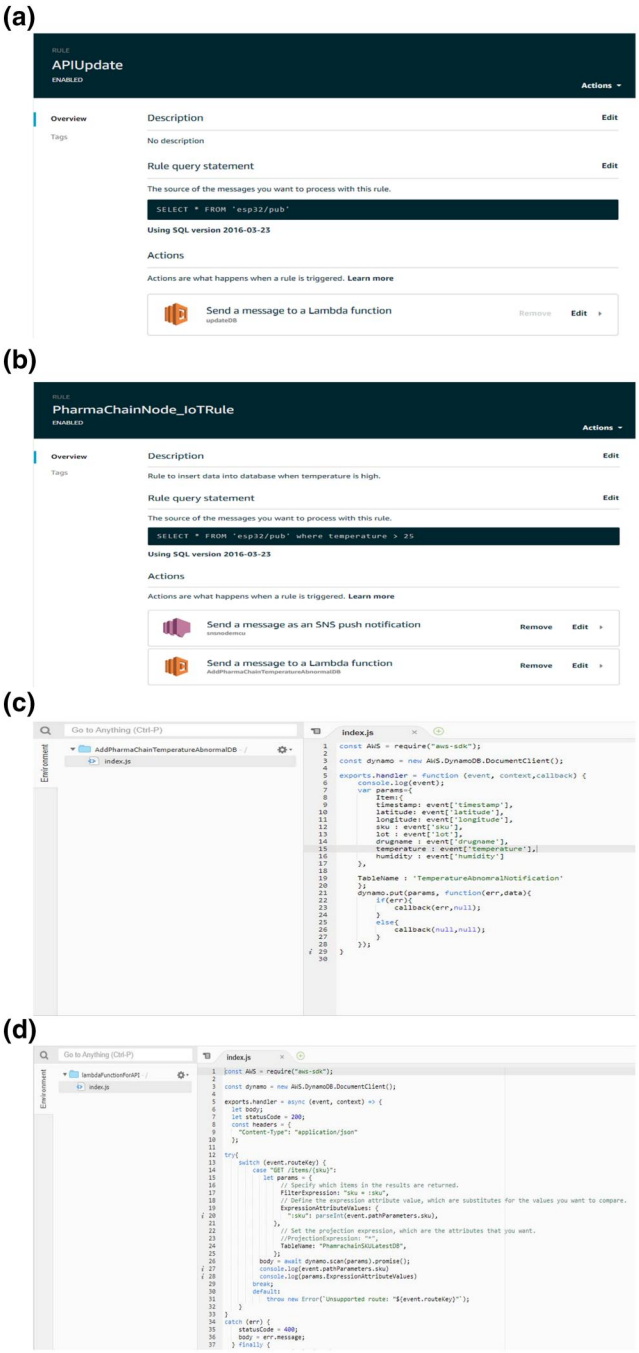


FIGURE 10 Implemented data source rules and functions in proposed PharmaChain

easy to get using faucets free of cost. This helps in developing and testing of the DApp before deploying it to the MAINNET. Chainlink also provides testnet jobs which take the test LINK token for performing transactions. Smart contracts designed in Ethereum for PharmaChain can be logically divided into three components. First component, which is an RBAC mechanism that helps in assigning Ethereum account, addresses of entities to certain roles and authorises them to perform certain functions. This RBAC mechanism is achieved by implementing Modifiers in solidity language. The shipment status is updated by the smart contract from every interaction between entities and this status assumed in different stages of the supply chain can be seen in Table 2. Implemented modifiers for RBAC mechanism and functions for access control are discussed below:

Function addManufacturer(address account) public onlyManufacturer: Helps in assigning a manufacturer role to a given Ethereum address. The Ethereum address of the manufacturer is passed as parameter. The function will check if the manufacturer role is already assigned to the address. If

TABLE 2 Assumed shipment states in proposed PharmaChain and corresponding enumeration

Pharmaceutical shipment state	Description	Enumerated state value
Produced by manufacturer	Acquiring RAW materials from Ingredient Suppliers and drugs are manufactured	0
Update inventory by manufacturer	Updating the inventory and offer drug lots for sale	1
Purchased by distributor	Distributor purchasing the lots from manufacturer	2
Shipped by manufacturer	Product shipping from manufacturer to distributor	3
Received by distributor	Lots received by distributor	4
Processed by distributor	Updating inventory with received products at distributor	5
Packaged by distributor	Received product is processed and sometimes repackaged to smaller packages	6
For sale by distributor	Update inventory for sale by distributor	7
Purchased by retailer	Re-packaged products are purchased by Healthcare network entities	8
Shipped By Distributor	Product shipped by distributor to retailers	9
Received by retailer	Received product at retailer locations	10
For sale by retailer	Updating the inventory and offering for sale	11
Purchased by consumer	Purchase of drugs by consumer from retailers	12

not, the address will be added to the list of authorised manufacturers.

Function renounceManufacturer() public: This function can help in revoking the authorisation of a manufacturer. Transaction sender address can be accessed in solidity using msg.sender and the address from which this function is being called will be removed from the authorised list.

Function isManufacturer(address account) public view returns (bool): This function doesn't emit any events (no gas cost) and is simply a view function to check if the sent address is assigned with a manufacturer role. If assigned, a boolean true will be returned, otherwise false.

Function addDistributor(address account) public onlyDistributor: Helps in assigning the distributor role to a given Ethereum address. The Ethereum address of the distributor is passed as parameter. The function will check if the distributor role is already assigned to the address. If not, the address will be added to the list of authorised distributors.

Function renounceDistributor() public: This function can help in revoking the authorisation of distributors. Transaction sender address can be accessed in solidity using msg.sender and the address from which this function is being called will be removed from the authorised list.

Function isDistributor(address account) public view returns (bool): This function doesn't emit any events (no gas cost) and is simply a view function to check if the sent address is assigned with the distributor role. If assigned, a boolean true will be returned, otherwise false.

Function addRetailer(address account) public onlyRetailer: Helps in assigning the Retailer role to a given Ethereum address. The Ethereum address of the retailer is passed as parameter. The function will check if the retailer role is already assigned to the address. If not, the address will be added to the list of authorised retailers.

Function renounceRetailer() public: This function can help in revoking the authorisation of retailers. Transaction sender address can be accessed in solidity using msg.sender and the address from which this function is being called will be removed from the authorised list.

Function isRetailer(address account) public view returns (bool): This function doesn't emit any events (no gas cost) and is simply a view function to check if the sent address is assigned with the retailer role. If assigned, a boolean true will be returned, otherwise false.

Function addConsumer(address account) public onlyConsumer: Helps in assigning the consumer role to a given Ethereum address. The Ethereum address of the retailer is passed as parameter. The function will check if the consumer role is already assigned to the address. If not, the address will be added to the list of authorised consumers.

Function renounceConsumer() public: This function can help in revoking the authorisation of consumers. Transaction sender address can be accessed in solidity using msg.sender and the address from which this function is being called will be removed from the authorised list.

Function isConsumer(address account) public view returns (bool): This function doesn't emit any events (no gas cost) and is simply a view function to check if the sent address is assigned with the consumer role. If assigned, a boolean true will be returned, otherwise false.

Modifier onlyOwner(): This modifier helps in verifying if the sender of the transaction (msg.sender) is the same as the Ethereum address of who deployed the contract, This helps in functions where the smart contract ownership has to be transferred to another Ethereum address.

Modifier producedByManufacturer(uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, the status of the shipment is checked. If the state matches the enumerated code '0' as shown in Table 2, it will return true otherwise false.

Modifier updateInventoryByManufacturer(uint _upc): Based on the Universal Product Code (UPC) which is sent as

parameter to this function, whether state of the manufactured product is moved to Update Inventory By Manufacturer is checked. If the state matches the enumerated code '1' as shown in Table 2, it will return true otherwise false.

Modifier purchasedByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is purchased by distributor is checked. If the state matches the enumerated code '2' as shown in Table 2, it will return true otherwise false.

Modifier shippedByManufacturer (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is shipped by manufacturer to distributor is checked. If the state matches the enumerated code '3' as shown in Table 2, it will return true otherwise false.

Modifier receivedByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is received by distributor is checked. If the state matches the enumerated code '4' as shown in Table 2, it will return true otherwise false.

Modifier processByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is inventory updated at distributor is checked. If the state matches the enumerated code '5' as shown in Table 2, it will return true otherwise false.

Modifier packagedByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is re-packaged by distributor is checked. If the state matches the enumerated code '6' as shown in Table 2, it will return true otherwise false.

Modifier forSaleByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is for sale by distributor is checked. If the state matches the enumerated code '7' as shown in Table 2, it will return true otherwise false.

Modifier shippedByDistributor (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is shipped by distributor is checked. If the state matches the enumerated code '8′ as shown in Table 2, it will return true otherwise false.

Modifier purchasedByRetailer (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is purchased by retailer is checked. If the state matches the enumerated code '9' as shown in Table 2, it will return true otherwise false.

Modifier receivedByRetailer (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is for received by retailer is checked. If the state matches the enumerated code '10' as shown in Table 2, it will return true otherwise false.

Modifier forSaleByRetailer (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is for sale by retailer is checked. If the state matches the enumerated code '11' as shown in Table 2, it will return true otherwise false.

Modifier purchasedByConsumer (uint _upc): Based on the Universal Product Code (UPC) which is sent as parameter to this function, whether state of the product is purchased by consumer is checked. If the state matches the enumerated code '12' as shown in Table 2, it will return true otherwise false.

Modifier onlyManufacturer(): This modifier helps in checking if the caller address is assigned with the role of manufacturer. This internally will call the isManufacturer(msg. sender) function to return true if authorised and false if not authorised.

Modifier onlyDistributor(): Same as above, this modifier checks the distributor role and returns true if authorised and false if not authorised.

Modifier onlyRetailer(): Same as above, this modifier checks the retailer role and returns true if authorised and false if not authorised.

Modifier onlyConsumer(): Same as above, this modifier checks the consumer role and returns true if authorised and false if not authorised.

Combining these modifiers with the supply chain functions designed in smart contracts creates a RBAC mechanism which will ensure that only authorised entities will be able to perform authorised transactions. Different functions implemented and their attached modifiers can be seen in the Table 3.

RequestTemperatureData(string memory _sku): This operation helps in fetching the ambient temperature of the shipment by interacting with Chainlink Oracle network. Stock Keeping Unit (SKU) is sent as parameter to this function and based on SKU value, the latest shipment ambient temperature is pulled from the API provided by data source designed.

RequestHumidityData(string memory _sku): This operation helps in fetching the relative humidity around the shipment by interacting with Chainlink Oracle network. Stock Keeping Unit (SKU) is sent as parameter to this function and based on SKU value, the latest shipment relative humidity is pulled from the API provided by data source designed.

RequestLatitude(string memory _sku): This operation helps in fetching the current latitude of the shipment by interacting with Chainlink Oracle network. Stock Keeping Unit (SKU) is sent as parameter to this function and based on SKU value, the latest shipments latitude location is pulled from the API provided by data source designed.

RequestLongitude(string memory _sku): This operation helps in fetching the current longitude of the shipment by interacting with Chainlink Oracle network. Stock Keeping Unit (SKU) is sent as parameter to this function and based on SKU value, the latest shipments longitude location is pulled from the API provided by data source designed.

VerifyAuthenticityoFProduct(uint _upc): This is used to verify the authenticity of the product and avoid counterfeits. This function call will verify the origin of the product and different interactions between the entities in the supply chain to ensure no counterfeits are introduced. As everyone in the network should be able to verify the authenticity, no modifier restrictions are applied to call this function.

ProduceItemByManufacturer(string memory _sku, string memory _drugName, uint _upc): When a product is

TABLE 3 PharmaChain smart contract functions and associated access control modifiers and event generations

Pharmaceutical Supply Chain Smart Contract Function	Modifiers	Event Generated
requestTemperatureData(string memory _sku)	Any	None
requestHumidityData(string memory _sku)	Any	None
requestLatitude(string memory _sku)	Any	None
requestLongitude(string memory _sku)	Any	None
verifyAuthenticityoFProduct(uint _upc)	Any	None
produceItemByManufacturer(string memory _sku, string memory _drugName, uint _upc)	onlyManufacturer()	ProducedByManufacturer(uint upc)
sellItemByManufacturer(uint _upc)	onlyManufacturer() producedByManufacturer(_upc) verifyCaller(items [_upc].ownerID)	UpdateInventoryByManufacturer (uint upc)
purchaseItemByDistributor(uint _upc)	onlyDistributor() updateInventoryByManufacturer(_upc)	PurchasedByDistributor(uint upc)
shippedItemByManufacturer(uint _upc)	onlyManufacturer() purchasedByDistributor(_upc) verifyCaller(items [_upc].originManufacturerID)	ShippedByManufacturer(uint upc)
receivedItemByDistributor(uint _upc)	onlyDistributor() shippedByManufacturer(_upc) verifyCaller(items [_upc].ownerID)	ReceivedByDistributor(uint upc)
processedItemByDistributor(uint _upc)	onlyDistributor() receivedByDistributor(_upc) verifyCaller(items [_upc].ownerID)	ProcessedByDistributor(uint upc)
packageItemByDistributor(uint _upc)	onlyDistributor() processByDistributor(_upc) verifyCaller(items[_upc]. ownerID)	PackagedByDistributor(uint upc)
sellItemByDistributor(uint _upc)	onlyDistributor() packagedByDistributor(_upc) verifyCaller(items [_upc].ownerID)	ForSaleByDistributor(uint upc)
purchaseItemByRetailer(uint _upc)	onlyRetailer() forSaleByDistributor(_upc)	PurchasedByRetailer(uint upc)
shippedItemByDistributor(uint _upc)	onlyDistributor() purchasedByRetailer(_upc) verifyCaller(items[_upc]. distributorID)	ShippedByDistributor(uint upc)
receivedItemByRetailer(uint _upc)	onlyRetailer() shippedByDistributor(_upc) verifyCaller(items[_upc]. ownerID)	ReceivedByRetailer(uint upc)
sellItemByRetailer(uint _upc)	onlyRetailer() receivedByRetailer(_upc) verifyCaller(items[_upc]. ownerID)	ForSaleByRetailer(uint upc)
purchaseItemByConsumer(uint _upc)	onlyConsumer()	PurchasedByConsumer(uint upc)

manufactured by the manufacturer, this function is called with SKU, UPC and Drug name as the parameters which will create a new product and assign the status of the product to enumerated value '0' and also generates an immutable log event "ProducedByManufacturer".

SellItemByManufacturer(uint _upc): This function is designed for manufacturer functions to offer the produced lots of drugs to sell. Calling this function will also verify if the manufacturer is the real owner of the produced lot before changing the status of the product to enumerated value '1' and emitting event for the immutable log.

PurchaseItemByDistributor(uint _upc): This operation is designed for distributors to buy the products offered by the manufacturer to sell. Calling this function will check if the caller is assigned with distributor privileges and also if the lot is available for sale by the manufacturer. Once both the conditions meet, the purchase will be done and the status of the shipment is changed to enumerated state '2'.

ShippedItemByManufacturer(uint _upc): Once purchase of lot by the distributor is successful, the manufacturer will ship the lot to the distributor. This function is called by the manufacturer who is the owner of the lot. An event is generated and recorded as an immutable log in blockchain.

ReceivedItemByDistributor(uint _upc): Helps in updating the inventory of the distributor once the shipment is received by the distributor from the manufacturer. The status of the product is changed to enumerated code '4' and a received shipment event will be generated to create an immutable log entry.

ProcessedItemByDistributor(uint _upc): Inventory updating and processing of incoming shipments is done by the distributor using this function. This ensures the distributor role and ownership to the lot before changing the status.

PackageItemByDistributor(uint _upc): Repackaging to smaller shipments is mostly done at the retailer. This function helps in recording the log for this operation and also update the status of the shipments.

SellItemByDistributor(uint _upc): Once inventory is updated with re-packaged goods, the product is offered for sale by the distributor owning. Only the distributor owning the

product should be able to call the function and an immutable log is generated for this event.

PurchaseItemByRetailer(uint _upc): Demand analysis is performed by the retailers and according to the need the products are purchased from the distributor. Once the purchase is successful, the status of the product and ownership will change and event for immutable log is generated.

ShippedItemByDistributor(uint _upc): Purchased product is moved from distributor to the retailer. This function call ensures the right distributor is sending the transaction and also ensures the shipment going to the right retailer. A shipping event is also generated to be recorded on to the immutable ledger.

ReceivedItemByRetailer(uint _upc): This function ensures the shipment reached the retailer and can be executed by the retailer owning the product. This function helps in recording the handover from the distributor to the retailer while creating an immutable log of event in the blockchain ledger.

SellItemByRetailer(uint _upc): Inventory of the retailer is updated with the incoming product and the products are offered for sale by the owning retailer. This function ensures that the sell offer is recorded and the product is available for purchase by any consumer.

PurchaseItemByConsumer(uint _upc): At the end of the supply chain, consumer purchases are handled by using this function. Once this function is executed, the ownership of the product will change from the retailer to the consumer and the log is maintained to ensure the product went to the right customer.

Deployed smart contract details along with the test accounts used for testing in the KOVAN Testnet are given in Table 4.

The sequence of steps performed in a life cycle of the drug in the implemented PharmaChain are:

- A manufacturer creates a new product and a unique identification is assigned by combination of SKU and UPC.
- After the manufacturing process, the inventory of the manufacturer is updated with newly created products and the drug lots are offered for sale.
- A distributor will purchase the lots of medication from the manufacturer, based on requirements.
- After a successful purchase the product from the manufacturer will be shipped to the distributor.
- The received product is processed and repackaged and inventory is updated by the distributor. Updated inventory is offered for sale by the distributor.
- Based on the demand patterns, a retailer will buy the drug shipments from the distributor.
- Successfully purchased goods will be shipped by the distributor to the retailer location.
- Retailer acknowledges the received shipments and the inventory is updated.
- Updated inventory is offered for sale by the retailer and made available to consumers.
- The consumer purchases the prescribed drugs from the retailer and should be able to follow the authenticity of the medication by looking up the trail of logs created during the journey of the drug.

The implemented PharmaChain is provided with a user-friendly web application interface. This web interface is built using JavaScript, Web3 JS, and Infura end points API to interact with the deployed smart contract in KOVAN Testnet. As discussed previously, different entities are assigned different roles and have some authorised functions. The web application is designed in such a way that based on the account from metamask it will enable the authorised function for the user to perform. The manufacturer has only access to "Produce Item By Manufacturer", "Sell Item By Manufacturer", and "Ship Item By Manufacturer". Similarly a distributor has access to "Purchase Item By Distributor", "Received Item By Distributor", "Processed Item By Distributor", "Package Item By Distributor", "Sell Item By Distributor" and "Shipped Item By Distributor". A retailer has access to "Purchase Item

TABLE 4 Deployed PharmaChain smart contract details

Parameters	Value
Deployed Contract Address	0x9409739e0D68Cb459a41892a48F3E62A2ceb7eeF
Contract Owner Address	0xBc6251e4f6389117a8f08b9B465a0bd44bA15Ab4
Manufacturer Address	0x3eDe97Ea0DFF3EcD7320b1822E33f4a2764E8ed4
Distributor Address	0xE8FDEa8272393Ad05f004d2BF583D784b509f9D3
Retailer Address	0x81F66b24db9EA43f454636750922c4ef12c26b48
Consumer Address	0xD46c558431c9CA642A85885509FBf6dA4Ba60435
Contract Creation Transaction Hash	0x0c99850a32d2f836023a8ada0ec3feab2ae34be2d16969c2d38537e51e77db7a
Oracle Address for Unsigned Integer data	0xc57B33452b4F7BB189bB5AfaE9cc4aBa1f7a4FD8
Job ID for Unsigned Integer data	d5270d1c311941d0b08bead21fea7747
Oracle Address for Signed Integer data	0x9C0383DE842A3A0f403b0021F6F85756524d5599
Job ID for Signed Integer data	ba1d5d5070a247eaa7070f838a42bb03

By Retailer", "Received Item By Retailer", and "Sell Item By Retailer". A consumer has access to "Purchase Item By Consumer". All the roles have access to "Fetch Item Details" and "Verify Authenticity of Product". Implemented web user interface can be seen in Figure 11.

6 | RESULTS OF THE IMPLEMENTED PHARMACHAIN SOLUTION

The sensing node which is responsible for capturing and communicating environmental parameters uses different sensors like DHT11 for both ambient temperature and relative humidity, and a GPS module for location of the shipment. Initially, the sensing node will connect to the Internet and establishe a secure MQTT channel using private keys and root certificates. A char buffer is used to populate the sensor data to form a JSON object before publishing the created topic for cloud IoT Hub to consume. The published JSON object will also include drug details like drug name, SKU and timestamp at which the data is published. The publishing data format with the establishment of a secure connection can be seen in Figure 12.

The alerting system designed in PharmaChain will equip the entities with real-time decision support tools (DST) for verifying shipment health and take prompt decisions to protect drugs from environmental fluctuations. A simple notification system which is provided by cloud as Function-as-a-Service (FaaS) is used to design this alerting system. MQTT communication model can also be used to send data which can be used to control environmental parameters by activating/deactivating actuators remotely. This will give a complete control and visibility of drugs moving throughout the supply chain. Simple notifications sent to the entities include drug details along with location and environmental parameters which can be seen in Figure 13.

A user-friendly interface is designed with the RBAC mechanism for PhamaChain. Ethereum addresses assigned for each entity are used to identify the role and functions which can be performed by the entity. All entities have access to the function using which product authenticity can be verified based on UPC. Drug specific details like UPC, SKU, Lot number can be accessed from QR codes attached. A QR scanner can be used to fetch these details and automatically populate the UPC to verify function parameter. Verifying the authenticity of the product can be seen in Figure 14.

7 | VALIDATION OF THE PROPOSED PHARMACHAIN

7.1 | Scalability and availability of proposed PharmaChain solution

Cost analysis is performed on the PharmaChain prototype to determine the usability of the proposed system in real-world applications. Whenever a transaction is submitted to the public Ethereum blockchain, transaction fee is collected by the miners to perform the consensus tasks along with the rewards generated from coinbase. This transaction cost consumed is

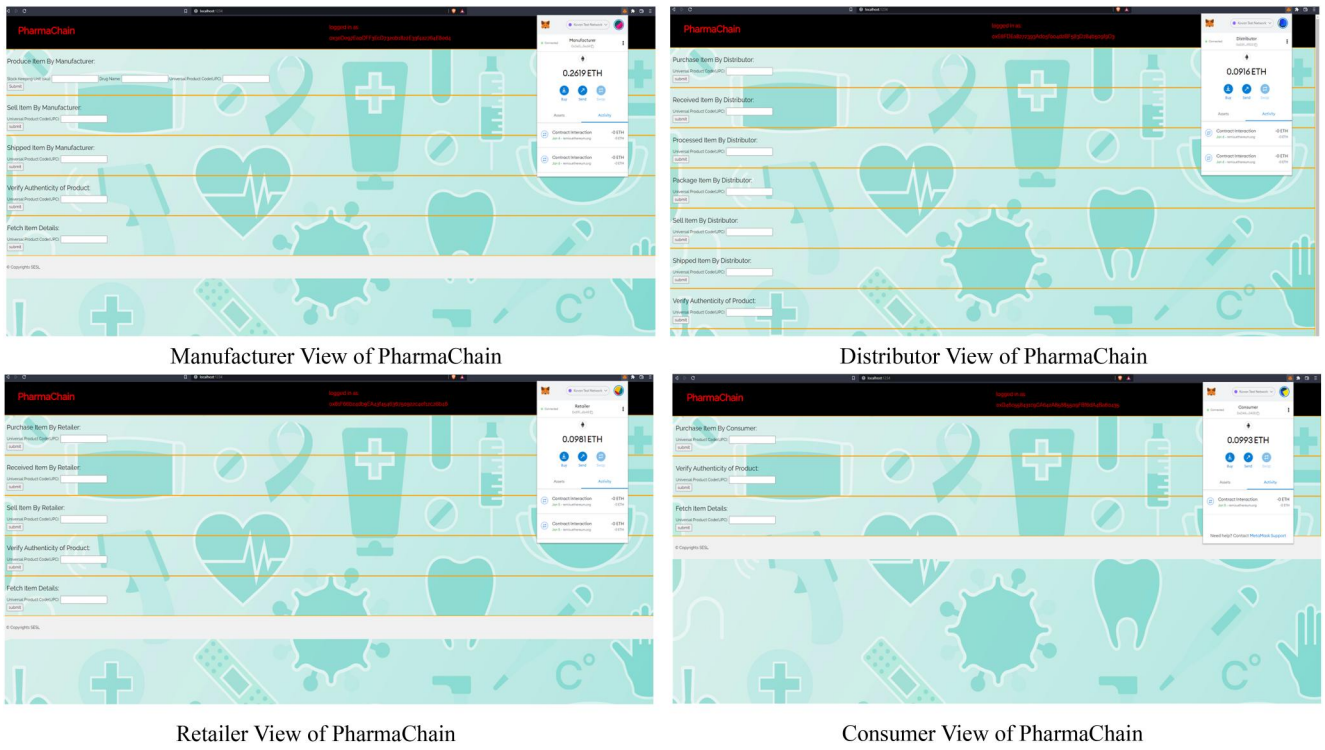


FIGURE 11 User interface for implemented PharmaChain based on different roles

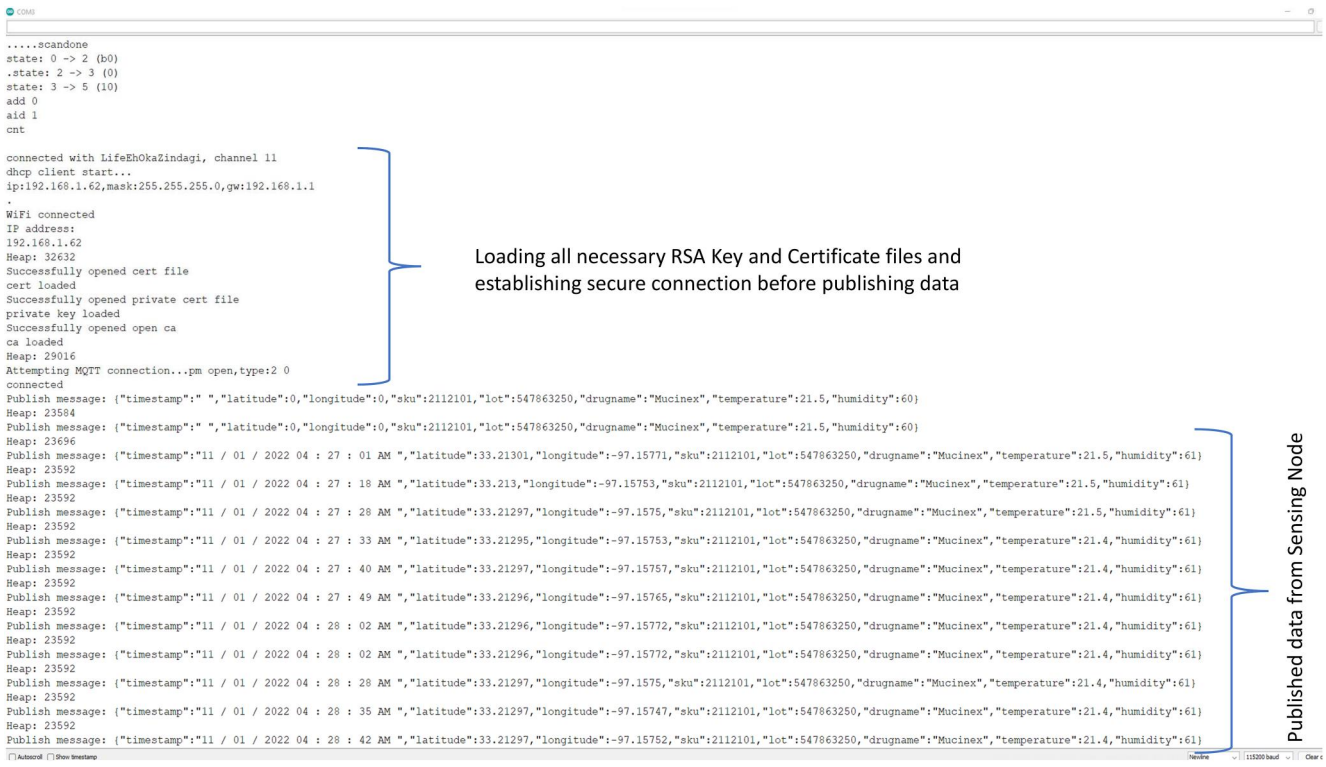


FIGURE 12 Data published from implemented sensing node for PharmaChain

termed as gas cost and is typically measured in GWEI. Conversion between one ether which is a native token to Ethereum is equivalent to 10^9 GWEI. Cost analysis is performed for each function including contract deployments and creating and assigning roles. As transaction costs vary based on the number of miners in the network, network congestion, cost of ether and some other factors, each transaction is executed 10 times and the average transaction costs are shown in Table 5. As the Ethereum network is currently working on Proof-of-Work (PoW), transaction costs are soaring with increase in network usage. These costs will go down as the price of Ethereum stabilizes and also with Ethereum 2.0 upgrade where the consensus used is moving from PoW to PoS. Another alternative to these transaction costs is to create a private Ethereum network of nodes from all the entities to form a P2P network which can significantly reduce the transaction costs used for both the Ethereum network and Chainlink. The average cost of the transactions from the developed prototype is 0.0000906 ETH which is 0.22$/transaction at an Ethereum price of 2290.89$ on May 10th, 2022. Along with this, Oracle nodes also collect incentives for providing real-time data for smart contracts. Average Oracle transaction fee computed from the prototype is 0.415 Link which is 3.726$ at Chainlink price of 8.98$ on May 10th, 2022. As we can see from the analysis, the cost of the drugs increases significantly if public Ethereum is used. Hence, a private Ethereum blockchain is a feasible solution for PSC where nodes from PSC entities can act as both miners and data providers which will eliminate the requirement of the transaction fee and reduce the cost overhead.

The amount of time taken for a transaction to be added to a block is considered as the transaction time in Ethereum. Transaction times are one of the important parameters analysed for scalability of the proposed DApps. Similar analysis is performed on the proposed PharmaChain solution. There are two components involved in transaction times of the currently proposed system: Time taken for mining and including the block and Chainlink interaction times. Time taken for the mining processes depends on multitude of parameters which include the availability of the miners, transaction fees offered, network congestion, price of ether at the time of transaction, etc. As shown in Table 5 the average transaction time for the implemented PharmaChain prototype in KOVAN Testnet is 5.6 s. The Chainlink interaction component significantly depends on the performance and availability of the data source designed. Load tests are performed on the data source of implemented PharmaChain application, and JMeter is used to run the test plan which executes 100 threads (users) with a loop count of 10 sent to the data source API provided by cloud. This test plan will send 1000 requests of data to the cloud within a short span of 2 s. Results from the load test can be seen in Table 6. The number of failed transactions among the 1000 requests sent is 0, which shows the scalability of the cloud data source provided. Average response time for the data source is 285 ms, which is not significant and is acceptable in solutions like the proposed PharmaChain. Response times of the 1000 requests sent are plotted in Figure 15 which shows that most of the data requests are served with response time less than 500 ms and only a small percentile of the requests

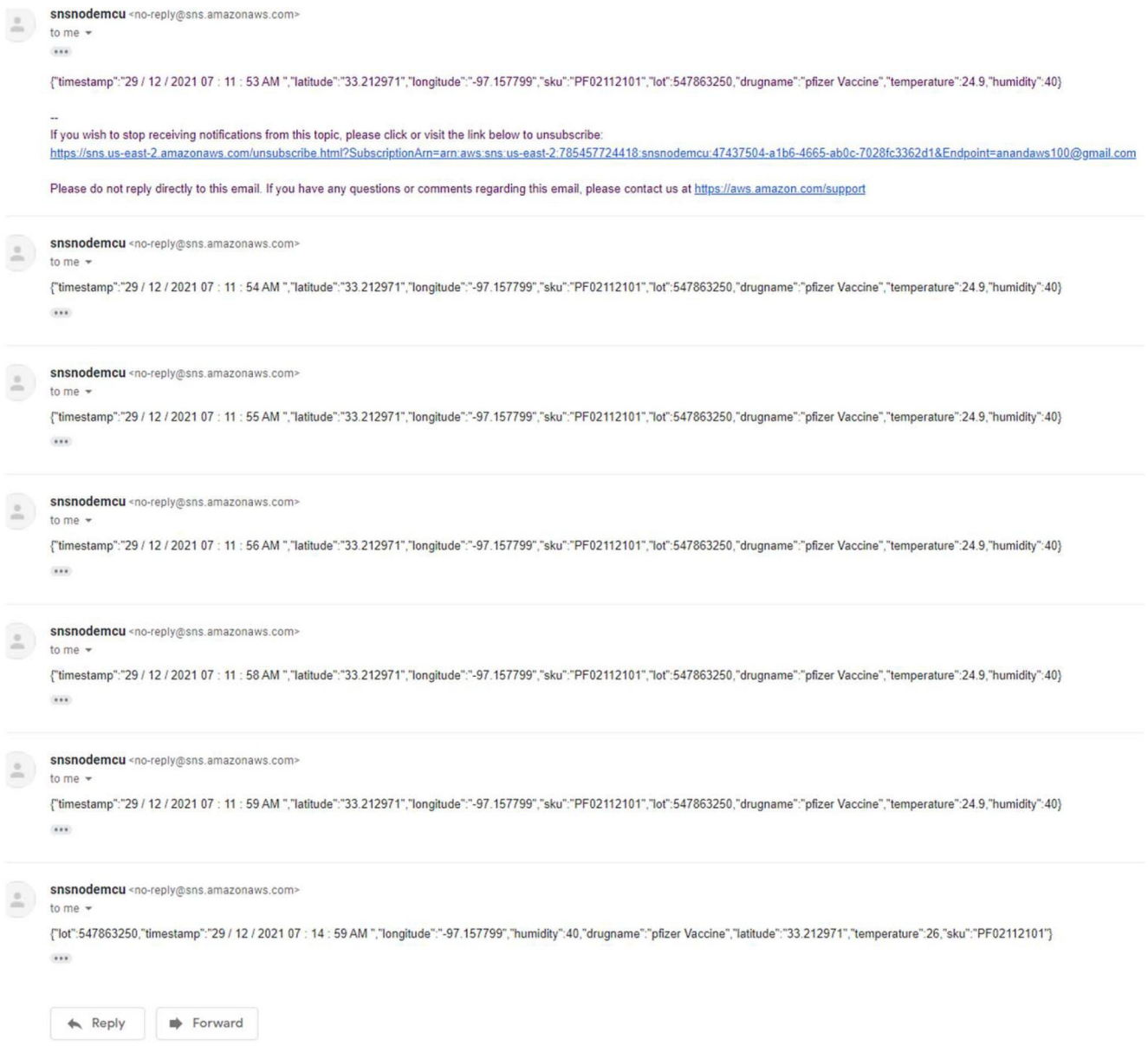
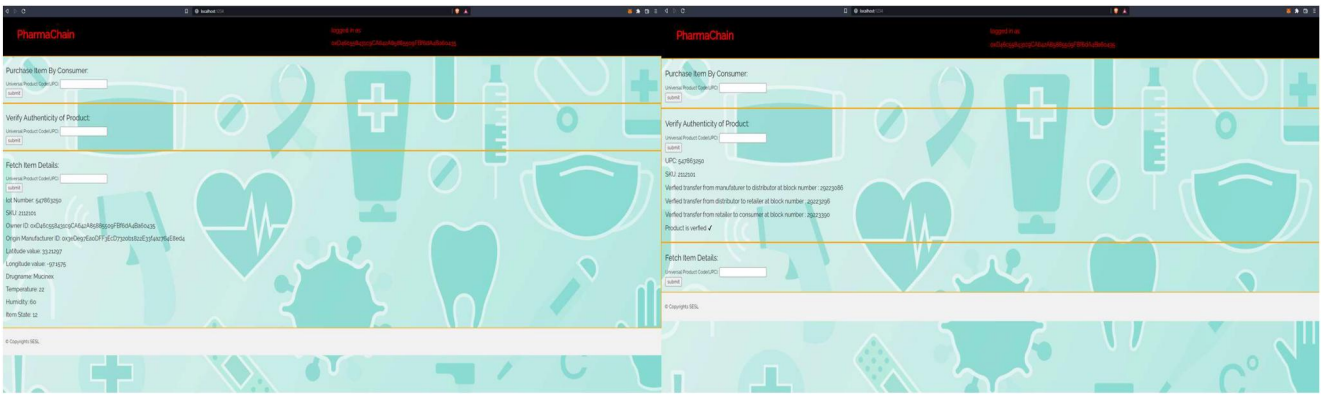


FIGURE 13 Alert notification sent in implemented PharmaChain proto-type with test data for temperature violation



Fetching Product History by Consumer

Verifying Authenticity of Product by Consumer

FIGURE 14 Consumer verifying product details and authenticity in proposed PharmaChain

TABLE 5 Cost and transaction time analysis of proposed PharmaChain solution

Action	Tx fees for Blockchain	Tx hash	Tx fees for Oracle	Block Time (s)
Contract Deployment	0.00898 ETH	0x0c99850a32d2f836023a8ada0ec3feab2a e34be2d16969c2d38537e51e77db7a	0	8
Add Manufacturer	0.00011 ETH	0xeae4963ac8b822d35f18bba2c20d5d01ee3ad4eb2451c6372c3d3f93e7b11b47	0	4
Add Distributor	0.00011 ETH	0x23c223c177185e78e4e7ab90b8bcf217930ebc57ebdcbce3431bfa366cd51bf9	0	4
Add Retailer	0.00011 ETH	0x28aa318991ea876d5bd2bd7238577068 adc690131af94ee06d9e61b01e41391d	0	8
Add Consumer	0.00011 ETH	0xd6d91da84aa9f4d9346a7ab7ea088816d bde750d566e8eeeb7f23bdf4489b442	0	4
Produce Item By Manufacturer	0.00151 ETH	0x28169fde8509c6f2996763eba72280b2b151337d74143b5579e91a3ca8655aa7	0.5 LINK	4
Sell Item By Manufacturer	0.00143 ETH	0x79048f114ea5c0ed737753f973f69c575a4 cdbd206d2cd33c69db97c5098f19a	0.5 LINK	8
Purchase Item By Distributor	0.00118 ETH	0xeab6ade50d16201b681a95a645499b0ae189f2d4738c670ab081cb2a39f34d42	0.4 LINK	4
Shipped Item By Manufacturer	0.00106 ETH	0xa52f4a9629c69c40c6550c00f903d1e631338596184e4552fa27841f8a9860a3	0.4 LINK	4
Received Item By Distributor	0.00106 ETH	0x97eeb340ff9fb6b87cb006f4d4005043d7 a809f0479de85ad84024ace520ce43	0.4 LINK	8
Processed Item By Distributor	0.00106 ETH	0x6ef2229a3947e4dde0dbe4f800158f78ff1fe9ede1300274f5c74b537218165a	0.4 LINK	4
Packaged Item By Distributor	0.00106 ETH	0xb11966bd5087d70c0baa6ead1e76e6b331b9d708aa43b8d8d82a997a2c185c3b	0.4 LINK	8
Sell Item By Distributor	0.00107 ETH	0xf7c0c2cac67f765a4fadaac4f935bd088b907d1cff4be0008328e0c1a68b6918	0.4 LINK	4
Purchase Item By Retailer	0.00118 ETH	0x25961c30875f3d9387c27dcf38b570fbcc fd7cdfc371404dc2f226c3d2d7d2fc	0.4 LINK	4
Ship Item By Distributor	0.00106 ETH	0xb6d6fc63e6d5d21e53468f0d91193175f623c66d649e1f4df028d5769c0c536e	0.4 LINK	4
Receive Item By Retailer	0.00106 ETH	0xd122e0c86688659a386a83a0a87c34aeb f6db1a8a820676ef14413ae03b886cb	0.4 LINK	8
Sell Item By Retailer	0.00106 ETH	0x13f1f0c73363b685445e54956d0f5667a7c0987aa1290bd1f26ef342d68ba275	0.4 LINK	4
Purchase Item By Consumer	0.00118 ETH	0x3e522790ae54650b3880f5bfcbdf9c437 f44b12a8298147fc34f8de417460094	0.4 LINK	4

took more than 500 ms response time to provide data to the Chainlink component. This analysis shows the scalability and availability of the proposed network to be implemented as a real-world application.

Current Enterprise Resource Planning Systems (ERP's) can process a large number of transactions and have higher throughput compared to blockchain solutions because of the Blockchain Trilemma which considers the three fundamental principles of blockchain: Decentralisation, Security and Scalability. If we have to maximise the scalability of a blockchain system, then compromise should be made on the other two principles. Research is currently being conducted to solve the blockchain trilemma which could include proposing new consensus mechanisms with faster ways to achieve consensus by not impacting decentralisation and security [44–46]. The proposed PharmaChain can be further improved in terms of scalability by adapting such faster consensus protocols to reach the baseline of the current ERP systems.

7.2 | Security analysis of the proposed PharmaChain solution

Security analysis is performed for the proposed PharmaChain solution on different security threats. This section discusses how the proposed system provides immunity towards such attacks.

- **Data Privacy:** Unauthorized access to patient information is a major problem in current healthcare systems which could lead to revealing patient health conditions and location data. The proposed blockchain based system makes use of RSA keys in place of identity, thereby providing complete anonymity of the consumer. This can help in preventing data privacy attacks to reveal the health conditions of the patients who access the PharmaChain solution.
- **Data Security:** Unauthorized modifications or introduction of malicious data into the network can cause data security issues. This is avoided as the digital signatures and RSA keys are used to perform the transactions. Along with that, data logged in the blockchain is hash linked and distributed which will provide immutability. This will prevent any modifications to the data which is already added to the blocks and also prevent malicious data to be entered into the network.
- **Accountability:** As each and every transaction performed by the entities is recorded on the blockchain along with the performing entity's Ethereum address, this will increase the accountability in case of intentional/unintentional errors performed in the network. This will help in identifying malicious nodes and prevent counterfeit medication to be introduced into the supply chain.
- **Access Control:** A robust role based access control mechanism using smart contracts is designed in the current proposed PharmaChain solution which will make use of modifiers to check the privileges of the user before calling

TABLE 6 Load testing results in proposed PharmaChain data source

Parameters	Value
Number of oracle requests sent	1000
Load duration	2 s
Failed requests	0
Percentage of error	0%
Average response time(ms)	285.196 ms
Minimum response time(ms)	78 ms
Maximum response time(ms)	1960 ms
Throughput (Requests/Sec)	16.66

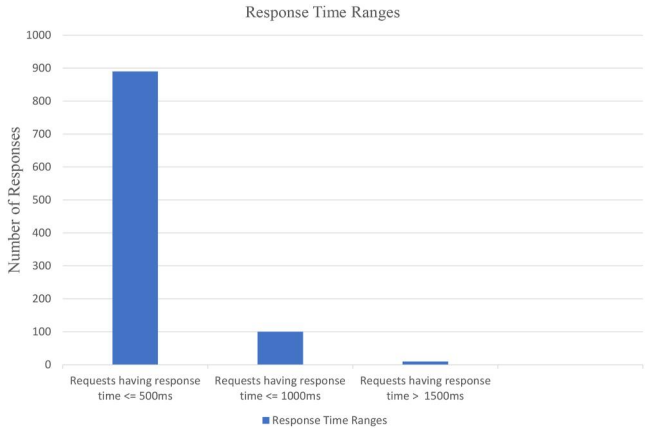


FIGURE 15 Response time overview of load test performed on data source implemented in PharmaChain

the smart contract functions. This will prevent both data security and privacy issues from the network.

- **Sybil Attack:** Sybil attack is one of the main threats of the P2P network in which a malicious node can create a large number of pseudonym identities and perform transaction in order to attack the reputation system. This is avoided by the RBAC mechanism developed as part of the PharmaChain which will make use of the Ethereum addresses of the entities to ensure that transaction is coming from only authorised nodes with privileges attached.

8 | CONCLUSION AND FUTURE RESEARCH

The proposed PharmaChain leverages the blockchain combined with smart contracts to provide a transparent ledger and helps in enhancing smooth operation and efficient communication between distributed entities. This also increases the accountability of participating entities thereby making it easy to track and remove malicious entities from the network. This will reduce the counterfeits being introduced into the supply chain. PharmaChain also provides consumers with tools to verify the authenticity of drugs before consumption to ensure the safety and build a consumer-oriented track and trace system. The proposed system is also integrated with the IoT network to monitor the real-time environmental parameters of the drug shipment and generate notifications to entities. This provides powerful decision support tools for entities to have control over the shipment when in the supply chain and take prompt actions to secure the integrity of the shipment before reaching to consumer. False data injection is a major problem when retrieving and using data from the sensing IoT network into blockchains which could lead to presenting wrong information to the consumer and other entities and tamper shipment integrity. This is avoided in the proposed PharmaChain by making use of Oracles which is a distributed data source mechanism based on the blockchain which will also solve problems with smart contracts to interact with real-world sensor data along with making the blockchain transactions re-playable to verify. A robust Role-Based Access Control Mechanism (RBAC) is implemented using smart contracts which will ensure that only authorised entities will be able to perform privileged functions.

A Proof-of-Concept prototype of the proposed PharmaChain solution is implemented in a test network and different performance metrics which include cost, security and scalability are analysed for adaptability of the proposed system into real-world applications. Result analysis has shown an average block time of 5.6 s which is acceptable in pharmaceutical supply chains. Along with blockchain performance metrics, cloud data source designed for Oracles is also analysed for scalability. Results from this analysis have shown that response times are 285 ms for Oracle data requests along with a success rate of 100% in the load test performed. This clearly shows the efficiency of the data source mechanism proposed in the PharmaChain. Hence, PharmaChain provides a reliable and scalable solution to detect and avoid counterfeit medication in Pharmaceutical Supply Chains.

The current work can be further extended by including the hardware security mechanism in order to authenticate the data coming from sensing nodes by using Physical Unclonable Functions (PUF) [47]. This hardware authentication scheme can help in reducing the power consumption by removing RSA encryption and digital signatures at sensing nodes thereby making it more scalable. The initial idea and workings of the PUFchain are proposed in Ref. [48]. More automation is needed in handling drugs in PSC in order to remove the data integrity issues and remove human errors that can lead to delivery of unsafe medicine. More research towards improving such systems that benefit smart healthcare [49] and smart cities [50] will be included in our future works in this direction.

All source codes implemented are made available for public access on GitHub [51]. A preliminary version of this work has been archived at Ref. [52].

DATA AVAILABILITY STATEMENT

Bapatla, A.K. 'Pharmachain source code'. Accessed March 2022. https://github.com/anandkumarbapatla/PharmaChain. Bapatla, A.K, Mohanty, S.P, Kougianos, E. 'Pharmachain: A blockchain to ensure counterfeit free pharmaceutical supply chain'. Available from: https://arxiv.org/abs/2202.02592.

ORCID

Saraju P. Mohanty https://orcid.org/0000-0003-2959-6541

REFERENCES

1. Jaberidoost, M., et al.: Pharmaceutical supply chain risks: a systematic review. DARU J. Pharma. Sci. 21(1), 1–7 (2013). https://doi.org/10.1186/2008-2231-21-69
2. Xu, L.D.: Enterprise systems: state-of-the-art and future trends. IEEE Trans. Ind. Inf. 7(4), 630–640 (2011). https://doi.org/10.1109/tii.2011.2167156
3. Musamih, A., et al.: A Blockchain-Based Approach for Drug Traceability in Healthcare Supply Chain, vol. 9, pp. 9728–9743. IEEE (2021)
4. Blackstone, E.A., Fuhr, J.P., Pociask, S.: The health and economic effects of counterfeit drugs. Am. Health & Drug Benefits. 7, 216–224 (2014)
5. Medicines, S.: A Conspiracy of Warm Boxes: The Story of Canadian Drug Wholesaler SB Medical and Their Disregard for Patient Safety (2022). https://rb.gy/ck8xws
6. UNODC.: Counterfeit Goods: A Bargain or a Costly Mistake (2022). https://www.unodc.org/toc/en/crimes/counterfeit-goods.html
7. Shafique, M.N., et al.: The role of wearable technologies in supply chain collaboration: a case of pharmaceutical industry. IEEE Access. 7, 49014–49026 (2019). https://doi.org/10.1109/access.2019.2909400
8. Shafique, M.N., et al.: The role of big data predictive analytics and radio frequency identification in the pharmaceutical industry. IEEE Access. 7, 9013–9021 (2019). https://doi.org/10.1109/access.2018.2890551
9. Breen, L.: A preliminary examination of risk in the pharmaceutical supply chain (PSC) in the national health service (NHS). J. Serv. Sci. Manag. 01(02), 193–199 (2008). https://doi.org/10.4236/jssm.2008.12020
10. Shamsuzzoha, A., Ndzibah, E., Kettunen, K.: Data-driven sustainable supply chain through centralized logistics network: case study in a Finnish pharmaceutical distributor company. Curr. Res. Environ. Sustain. 2, 100013 (2020). https://doi.org/10.1016/j.crsust.2020.100013
11. Schmoldt, A., Benthe, H.F., Haberland, G.: Digitoxin metabolism by rat liver microsomes. Biochem. Pharmacol. 24(17), 1639–1641 (1975). https://doi.org/10.1016/0006-2952(75)90094-5
12. Sharma, A., Gupta, P., Jha, R.: COVID-19: Impact on health supply chain and lessons to be learnt. J. Health Manag. 22(2), 248–261 (2020). https://doi.org/10.1177/0972063420935653
13. FDA.: Drug Supply Chain Security Act (Dscsa). FDA (2022). https://www.fda.gov/drugs/drug-supply-chain-integrity/drug-supply-chain-security-act-dscsa
14. Badhotiya, G.K., et al.: Investigation and assessment of blockchain technology adoption in the pharmaceutical supply chain. Mater. Today Proc. 46, 10776–10780 (2021). https://doi.org/10.1016/j.matpr.2021.01.673
15. Kuhn, R., Yaga, D., Voas, J.: Rethinking distributed ledger technology. Computer. 52(2), 68–72 (2019). https://doi.org/10.1109/mc.2019.2898162
16. Rivera, R., et al.: How digital identity on blockchain can contribute in a smart city environment. In: International Smart Cities Conference (ISC2), (2017)
17. Xie, J., et al.: A survey of blockchain technology applied to smart cities: research issues and challenges. IEEE Communications Surveys & Tutorials. 21(3), (2019). https://doi.org/10.1109/comst.2019.2899617
18. Rachakonda, L., et al.: SaYoPillow: blockchain-integrated privacy-assured IoMT framework for stress management considering sleeping habits. IEEE Trans. Consum. Electron. 67(1), 20–29 (2021). https://doi.org/10.1109/tce.2020.3043683
19. Azaria, A., et al.: MedRec: using blockchain for medical data access and permission management. In: 2nd International Conference on Open and Big Data (OBD), pp. 1–2 (2016)
20. Vangipuram, S.L.T., Mohanty, S.P., Kougianos, E.: CoviChain: a blockchain based framework for nonrepudiable contact tracing in healthcare cyber-physical systems during pandemic outbreaks. SN Comput. Sci. 2(5), 346 (2021). https://doi.org/10.1007/s42979-021-00746-x
21. Bapatla, A.K., Mohanty, S.P., Kougianos, E.: sFarm: a distributed ledger based remote crop monitoring system for smart farming. In: Internet of Things. Technology and Applications, pp. 13–31. Springer International Publishing (2022)
22. Caro, M.P., et al.: Blockchain-based traceability in agri-food supply chain management: a practical implementation. In: IoT Vertical and Topical Summit on Agriculture, (2018)
23. Tian, H., et al.: Research on distributed blockchain-based privacy-preserving and data security framework in IoT. IET Commun. 14(13), 2038–2047 (2020). https://doi.org/10.1049/iet-com.2019.0485
24. Shahid, A., et al.: Blockchain-based agri-food supply chain: a complete solution. IEEE Access. 8, 69230–69243 (2020). https://doi.org/10.1109/access.2020.2986257
25. Buterin, V.: A Next-Generation Smart Contract and Decentralized Application Platform (2015). https://blockchainlab.com/pdf/Ethereum_white_paper-a_next_generation_smart_contract_and_decentralized_application_platform-vitalik-buterin.pdf
26. Back, A.: Hashcash - a Denial of Service Counter-measure (2022). http://www.hashcash.org/hashcash.pdf
27. Zheng, Z., et al.: An overview on smart contracts: challenges, advances and platforms. Future Generat. Comput. Syst. 105, 475–491 (2020). https://doi.org/10.1016/j.future.2019.12.019
28. Bamakan, S.M.H., Moghaddam, S.G., Manshadi, S.D.: Blockchain-enabled pharmaceutical cold chain: applications, key challenges, and future trends. J. Clean. Prod. 302, 127021 (2021). https://doi.org/10.1016/j.jclepro.2021.127021
29. Wang, G., Huang, S.H., Dismukes, J.P.: Manufacturing supply chain design and evaluation. Int. J. Adv. Manuf. Technol. 25(1-2), 93–100 (2004). https://doi.org/10.1007/s00170-003-1791-y
30. Subramanian, G., et al.: Crypto pharmacy – digital medicine: a mobile application integrated with hybrid blockchain to tackle the issues in pharma supply chain. IEEE Open J. Comput. Soci. 2, 26–37 (2021). https://doi.org/10.1109/ojcs.2021.3049330
31. Bocek, T., et al.: Blockchains everywhere - a use-case of blockchains in the pharma supply-chain. In: IFIP/IEEE Symposium on Integrated Network and Service Management, (2017)
32. Kumar, R., Tripathi, R.: Traceability of counterfeit medicine supply chain through blockchain. In: International Conference on Communication Systems & Networks (COMSNETS), (2019)
33. Huang, Y., Wu, J., Long, C.: Druggeder A practical blockchain system for drug traceability and regulation. In: International Conference on Internet of Things (iThings) and Green Computing and Communications (GreenCom) and Cyber, Physical and Social Computing (CPSCom) and Smart Data, SmartData (2018)
34. Kumar, A., et al.: Combating counterfeit drugs: a quantitative analysis on cracking down the fake drug industry by using blockchain technology. In: 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence), (2019)
35. Singh, R., Dwivedi, A.D., Srivastava, G.: Internet of things based blockchain for temperature monitoring and counterfeit pharmaceutical prevention. Sensors. 20(14), 3951 (2020). https://doi.org/10.3390/s20143951
36. Alkhoori, O., et al.: Design and implementation of CryptoCargo: a blockchain-powered smart shipping container for vaccine distribution. IEEE Access. 9, 53786–53803 (2021). https://doi.org/10.1109/access.2021.3070911
37. Atzei, N., Bartoletti, M., Cimoli, T.: A survey of attacks on ethereum smart contracts (SoK). In: Lecture Notes in Computer Science, pp. 164–186. Springer Berlin Heidelberg (2017)
38. Zhang, F., et al.: Town crier. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, (2016)
39. Pham, H.L., Tran, T.H., Nakashima, Y.: Practical anti-counterfeit medicine management system based on blockchain technology. In: 4th Technology Innovation Management and Engineering Science International Conference, (2019)
40. Jangir, S., et al.: A novel framework for pharmaceutical supply chain management using distributed ledger and smart contracts. In: 10th International Conference on Computing, Communication and Networking Technologies, (2019)

41. Celiz, R.C., Cruz, Y.E.D.L., Sanchez, D.M.: Cloud model for purchase management in health sector of Peru based on IoT and blockchain. In: IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference, (2018)
42. Suliman, A., et al.: Monetization of IoT data using smart contracts. IET Netw. 8(1), 32–37 (2019). https://doi.org/10.1049/iet-net.2018.5026
43. Breidenbach, L., et al.: Chainlink 2.0: Next Steps in the Evolution of Decentralized Oracle Networks. https://rb.gy/tc38po
44. Puthal, D., Mohanty, S.P.: Proof of authentication: IoT-friendly blockchains. IEEE Potentials. 38(1), 26–29 (2019). https://doi.org/10.1109/mpot.2018.2850541
45. Jalalzai, M.M., Busch, C., Richard, G.G.: Proteus A scalable BFT consensus protocol for blockchains. In: 2019 IEEE International Conference on Blockchain (Blockchain), (2019)
46. Guo, H., Li, W., Nejad, M.: A location-based and hierarchical framework for fast consensus in blockchain networks. In: 2021 4th International Conference on Hot Information-Centric Networking (HotICN), (2021)
47. Yanambaka, V.P., Mohanty, S.P., Kougianos, E.: Making use of manufacturing process variations: a dopingless transistor based-PUF for hardware-assisted security. IEEE Trans. Semicond. Manuf. 31(2), 285–294 (2018). https://doi.org/10.1109/tsm.2018.2818180
48. Mohanty, S.P., et al.: PUFchain: a hardware-assisted blockchain for sustainable simultaneous device and data security in the Internet of everything (IoE). IEEE Consumer Electronics Magazine. 9(2), 8–16 (2020). https://doi.org/10.1109/mce.2019.2953758
49. Sayeed, M.A., et al.: eSeiz: an edge-device for accurate seizure detection for smart healthcare. IEEE Trans. Consum. Electron. 65(3), 379–387 (2019). https://doi.org/10.1109/tce.2019.2920068
50. Ram, S.K., et al.: Energy perspectives in IoT driven smart villages and smart cities. IEEE Consumer Electr. Mag. 10(3), 19–28 (2021). https://doi.org/10.1109/mce.2020.3023293
51. Bapatla, A.K.: Pharmachain Source Code. https://github.com/anandkumarbapatla/PharmaChain
52. Bapatla, A.K., Mohanty, S.P., Kougianos, E.: Pharmachain A Blockchain to Ensure Counterfeit Free Pharmaceutical Supply Chain. https://arxiv.org/abs/2202.02592

How to cite this article: Bapatla, A.K., et al.: PharmaChain: a blockchain to ensure counterfeit-free pharmaceutical supply chain. IET Netw. 12(2), 53–76 (2023). https://doi.org/10.1049/ntw2.12041